МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«САМАРСКИЙ ГОСУДАРСТВЕННЫЙ АЭРОКОСМИЧЕСКИЙ
УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)» (СГАУ)

Информатика

Электронный образовательный контент
в системе дистанционного обучения MOODLE

САМАРА
2012

Автор-составитель: **Савченко Ольга Геннадьевна, Шляпугин Алексей Геннадьевич**

Информатика [Электронный ресурс] : электрон. образоват. контент в системе дистанц. обучения MOODLE / Минобрнауки России, Самар. гос. аэрокосм. ун-т им. С. П. Королева (нац. исслед. ун-т); авт.-сост. О. Г. Савченко, А. Г. Шляпугин.- Электрон. текстовые и граф. дан. (576Мбайт). - Самара, 2012. – 1 эл. опт. диск (CD-ROM).

В состав учебно-методического комплекса входят:

1.Курс лекций.
2.Дидактический материал.
3.Примеры заданий к лабораторным работам.
4. Тесты для контроля знаний.

Электронный образовательный контент предназначен для студентов инженерно-технологического факультета, обучающихся по направлению подготовки бакалавров 150400.62 «Металлургия», изучающих дисциплину «Информатика» и 150700.62 «Машиностроение», изучающих дисциплину «Информационные технологии» в 1 семестре.

Подготовлено на кафедре обработки металлов давлением СГАУ.

Язык программирования Паскаль.
Знакомство со средой программирования Турбо Паскаль. Основные понятия.
Первая программа.

Паскаль - язык профессионального программирования, который назван в честь французского математика и философа Блеза Паскаля (1623-1662) и разработан в 1968-1971 гг. Никлаусом Виртом. Первоначально был разработан для обучения, но вскоре стал использоваться для разработки программных средств в профессиональном программировании.

Паскаль популярен среди программистов по следующим причинам:

1.Прост для обучения.

2.Отражает фундаментальные идеи алгоритмов в легко воспринимаемой форме, что предоставляет программисту средства, помогающие проектировать программы.
3.Позволяет четко реализовать идеи структурного программирования и структурной организации данных.
4.Использование простых и гибких структур управления: ветвлений, циклов.
5.Надежность разрабатываемых программ.

Турбо Паскаль - это система программирования, созданная для повышения качества и скорости разработки программ (80-е гг.). Слово Турбо в названии системы программирования - это отражение торговой марки фирмы-разработчика Borland International (США).

Систему программирования Турбо Паскаль называют интегрированной (integration - объединение отдельных элементов в единое целое) средой программирования, т.к. она включает в себя редактор, компилятор, отладчик, имеет сервисные возможности.

Основные файлы Турбо Паскаля:

Turbo.exe - исполняемый файл интегрированной среды программирования;
Turbo.hlp - файл, содержащий данные для помощи;
Turbo.tp - файл конфигурации системы;
Turbo.tpl - библиотека стандартных модулей, в которых содержатся встроенные процедуры и функции (SYSTEM, CRT, DOS, PRINTER, GRAPH, TURBO3, GRAPH3).

Запуск интегрированной среды программирования.

Для запуска интегрированной среды программирования нужно установить текущим каталог с Турбо Паскалем (TP7\BIN) и ввести команду: turbo.exe.

Запустить среду программирования и рассмотрите экран. Перед вами полоса меню, область окна и строка статуса. Нажмите клавишу F10 - теперь вам доступны все опции меню. С помощью клавиш перемещения курсора рассмотрите меню. С командами меню мы будем знакомиться постепенно. Нажмите клавишу Esc (вы вышли из меню). Перемещая курсор в окне следите за строкой статуса.Почти все что вы видите и делаете в среде Турбо Паскаль происходит в окнах.

Окно - это область экрана, которую можно перемещать, изменять в размере, перекрывать, закрывать и открывать.

Интегрированная среда программирования Турбо Паскаль позволяет иметь любое количество открытых окон, но в любой момент времени активным может быть только одно.

Активное окно - это окно с которым вы в настоящий момент работаете.

Общие горячие клавиши:

F1 - выводит окно подсказки;
F2 - сохраняет файл активного окна;
F3 - появление диалогового окна и возможность открыть файл;
F4 - запускает программу до строки, на которой стоит курсор;
F5 - масштабирует диалоговое окно;
F6 - переходит к следующему открытому окну;
F7 - запускает программу в режиме отладки с заходом внутрь процедур;
F8 - запускает программу в режиме отладки, минуя вызов процедур;
F9 - компилирование программы в текущем окне;
F10 - возвращение в меню.

Войти в меню возможно:

1.с помощью "мышки";

2.с помощью клавиши F10;

3..с помощью комбинации Alt+<выделенная буква>. О том, что мы в меню свидетельствует **курсор** - прямоугольник зеленого цвета.

С помощью клавиш управления курсором подсветите слово FILE и нажмите клавишу "Enter". Появилась вертикальная таблица со списком команд, называемая выпадающим меню.

Open-F3 - открыть существующий файл (при активизации этой опции появляется окно со списком файлов, где можно выбрать необходимый),

New - создать новый файл (очищает память редактора и переводит в режим создания нового файла, которому присваивается имя Noname.pas; имя можно изменить при записи файла на диск),

Save-F2 - сохранить файл (переписывает файл из памяти редактора на диск),

Save as - сохранить с новым именем,

Save all - сохранить все в окнах (записывает содержимое всех окон редактора в соответствующие файлы),

Change dir - смена каталога (позволяет изменить установленный по умолчанию диск или каталог),

Print - печать файла,

Get info - выдача информации о текущем состоянии программы и используемой памяти,

DOS Shell - выход в DOS без выгрузки из памяти (для возврата ввести команду exit),

Exit - выход и выгрузка из памяти.

Программы на языке Паскаль имеют блочную структуру:

1. Блок типа PROGRAM - имеет имя, состоящее только из латинских букв и цифр. Его присутствие не обязательно, но рекомендуется записывать для быстрого распознавания нужной программы среди других листингов.

2. Программный блок, состоящий в общем случае из 7 разделов:

раздел описания модулей (uses);

раздел описания меток (label);

раздел описания констант (const);
раздел описания типов данных (type);
раздел описания переменных (var);
раздел описания процедур и функций;
раздел описания операторов.

Общая структура программы на языке Паскаль следующая:

```
Program ИМЯ..; {заголовок программы}

Uses ...;          {раздел описания модулей}

Var ..;            {раздел объявления переменных}

 ...

Begin              {начало исполнительной части программы}

 ...               {последовательность операторов}

End.               {конец программы}
```

Составим программу, которая складывает два числа и выводит сумму на экран.

Откройте файл, в который Вы запишите эту программу. Для этого нажмите клавишу F10, чтобы выйти в главное меню, затем клавишами перемещения курсора выберите опцию File, а в выпавшем меню команду New.

Обратите внимание на оформление текста программы.

```
Program Summa;          {Вычислить сумму двух чисел и вывести на
экран.}
Var
n1, n2,rez: integer;          {переменная для хранения первого и второго
числа,        для хранения результата вычисления, описаны , как целые}
Begin                         {начало программы}
  n1 := 3;                    {присваиваем переменной n1 значение 3}
  n2 := 4;                    {присваиваем переменной n2 значение 4}
  {складываем значения переменных n1 и n2 и результат присваиваем
переменной rez }
  rezt := n1 + n2;
Write (n1, '+', n2,'=',rez);  {вывод примера на экран}
End.                          {признак конца программы}
```

Имя этой программы Summa. Заметим, что требования к имени выполняются: оно отражает содержание программы, а также не содержит недопустимых символов.

Далее идет специально выделенный комментарий в {}. Комментарии в программе можно не писать , если программа читаема. Транслятор комментарии не видит

Из разделов описаний имеется лишь один - раздел переменных. Он начинается со служебного слова Var. Описаны три переменные: n1, n2, rez. Все они переменные целого типа. Поэтому мы перечислили их через запятую, поставили двоеточие и указали тип переменных. Подобные объявления разделяются между собой точкой с запятой.

После описательной части идет раздел операторов, начинающийся со служебного слова Begin, после которого идут операторы языка.

Недостатком этой программы является то, что значения переменных постоянны.

Оператор присваивания.

Арифметические выражения.

Типы данных.

Оператор присваивания - основной оператор любого языка программирования.

Общая форма записи оператора: имя величины := выражение

Например,

V:=A

V:=A+1

При помощи оператора присваивания переменной могут присваиваться константы и выражения, значения переменных любого типа.

Как только в программе встречается переменная, для неё в памяти отводится место. Оператор присваивания помещает значение переменной или значение выражения в отведённое место.

Если в процессе выполнения программы встречается переприсваивание (т.е. та же самая переменная принимает другое значение), то старое значение переменной стирается, на свободное место записывается новое значение. Команда присваивания позволяет лучше понять смысл слова переменная (т.е. меняющая своё значение по ходу программы).

Выражение может быть арифметическим, логическим или литерным. Важно, чтобы тип величины был согласован с видом выражения.

Арифметические выражения должны быть записаны в так называемой линейной записи согласно следующим правилам:

-выражение должно быть записано в виде линейной цепочки символов;

-используемые операции приведены в таблице:

НАЗВАНИЕ ОПЕРАЦИИ	ФОРМА ЗАПИСИ
сложение	x + y
вычитание	x - y
умножение	x * y

деление x / y

-нельзя опускать знаки операций, например писать 5b. Для записи произведения чисел 5 и b надо писать 5*b;

-аргументы функций (sin, cos и др.) как и аргументы вспомогательных алгоритмов, записываются в круглых скобках, например sin(x), cos(4*x).

Порядок выполнения операций

Порядок выполнения операций при вычислении арифметических выражений можно регулировать при помощи скобок по обычным правилам. Там, где скобки отсутствуют, ЭВМ выполняет операции в следующем порядке:

-вычисляет значение всех алгоритмов-функций и стандартных функций;

-выполняет справа налево все операции возведения в степень;

-выполняет слева направо все операции умножения и деления;

-выполняет слева направо все операции сложения и вычитания.

В нашем примере, нахождение суммы двух чисел, сначала переменной n1 присваивается значение равное 3 и переменной n2 присваивается значение равное 4, затем вычисляется значение выражения (n1 + n2) и оно присваивается переменной rez. Сумма чисел посчитана.

Теперь надо вывести ее значение на экран. Для этого используют оператор Write - записать (вывести) на экран значение переменной, записанной в скобках. В нашем случае значение переменной n1, затем символ + , далее значение переменной n2, символ = и, наконец, значение результата rez.

И, наконец, в конце раздела операторов стоит служебное слово End, после которого стоит точка.

Имя программы должно соответствовать ее содержанию.

Имя файла должно быть таким же, как и имя программы,

Файлы, содержащие программы, относящиеся к одной теме, должны находиться в одном каталоге, название этого каталога должно отражать его содержание.

Для сохранения файла на диске, нужно из меню F10-File выберите команду Save и в предложенной строке наберите путь C:\USER_D~1\4\PRIM2

C-диск с именем C;

USER_D~1- адрес папки «Мои документы»

PRIM2 имя файла, по умолчанию с расширением PAS.

Основные определения.

Типы данных.

Алгоритм - четкая последовательность действий, необходимая для решения задачи.

Программа - алгоритм, записанный на языке программирования.

Алфавит языка - набор элементарных символов, используемый для составления программ. Алфавит языка Паскаль содержит:

-52 буквы латинского алфавита (строчные и заглавные);

-арабские цифры (0-9);

-специальные символы;

-знаки математических действий (+ - * /);

-знаки пунктуации (. : , ; " `);

-скобки ([ ] ( ) { });

-знак пробела;

-знаки отношений (< > =).

Идентификатор (имя) - имя какого-либо элемента программы, которое должно удовлетворять следующим требованиям:

-длина имени не должна превышать 63 символа,

-первым символом не может быть цифра,

-переменная не может содержать пробел;

-имя не должно совпадать с зарезервированным (служебным) словом,

-прописные и строчные буквы воспринимаются одинаково.

Зарезервированные (служебные) слова - это слова, использующиеся только по своему прямому назначению. Их нельзя использовать в качестве переменных, так как они выполняют определенную смысловую нагрузку.

Примеры зарезервированных слов: AND, GOTO, PROGRAM, ELSE, IF, RECORD, NOT, ARRAY, REPEAT, UNTIL, BEGIN, IN, SET, END, CASE, CONST, USES, INTERFACE, STRING, LABEL, THEN, OF, DIV, TO, VAR, DO, TYPE, WHILE, DOWNTO, FILE, FUNCTION, PROCEDURE и другие.

Переменные (Var) - вид данных, который может изменять свое значение в ходе программы, описывают переменные после зарезервированного слова Var.

Константы (Const) - вид данных, который является постоянным на всем протяжении выполнения программы, описывают константы после зарезервированного слова Const.

Комментарии - некоторая запись, служащая для пояснения программы, которая записывается в фигурных скобках.

Типы данных

Для временного хранения информации в операторах памяти машины в языке Паскаль используются константы и переменные. Они могут быть различных типов:

-целых чисел (integer, longint);

-действительных чисел (real);

-символьный тип (char);

-строковый (string);

-логический (boolean);

-сложные (комбинированный (record), множественный (set) и другие).

Целые типы:

Название	Длина в байтах	Диапазон значений
Byte	1	0 ... 255
ShortInt	1	-128 ... 127
Word	2	0 ... 65535

Integer	2	-32768 ... 32767
LongInt	4	-2147483648 ... 2147483647

Над целыми типами определены такие операции:

1. "+" - сложение;
2. " * " - умножение;
3. " - " вычитание;
4. div - целочисленное деление;
5. mod - получение остатка от целочисленного деления.

Вещественные типы:

Вещественные типы представляются с некоторой точностью, которая зависит от компьютера. Вам необходимо знать, что вещественный тип разделяется на несколько типов, но использовать мы будем вещественные данные только типа Real, которые занимают 6 байт, имеют диапазон возможных значений модуля от 2.9E-39 до 1.7E+38 и точность представления данных - 11...12 значащих цифр.

Примечание. Несмотря на то, что в Turbo Pascal имеется широкий выбор вещественных типов, доступ к некоторым из них (single, double, extended) возможен при особых режимах компиляции. Особое положение в Turbo Pascal занимает тип comp, трактующийся как вещественное число без экспоненциальной и дробной частей. Он сохраняет 19 - 20 значащих цифр и знак числа. В то же время comp полностью совместим с любыми другими вещественными типами.

В языке Паскаль числа могут быть представлены в двух видах: с фиксированной точкой и плавающей запятой.

Числа с фиксированной точкой изображаются десятичным числом с дробной частью, которая может быть и нулевой. Например, 27.9, 5.00

Такие большие числа как 137.000.000 можно записать в виде чисел с десятичным порядком 1.37•108. Такие числа имеют вид mEp. Здесь m - мантисса; E - признак записи числа с десятичным порядком; p - степень числа 10. Получится 1.37E+8. Такие числа, представленные с десятичным порядком и называются числами с плавающей точкой. Например,

Математическая запись	Запись на Паскале
4•10-4	4E -4
0,62•105	0.62E+5
-10,88•1012	-10.88E12

Компьютер, по умолчанию, представляет действительные числа в виде чисел с плавающей точкой. Такое представление чисел не очень нравится пользователям. Поэтому мы будем “заставлять” компьютер выдавать действительные числа в более привычном варианте следующим образом:

R:m:n

R - действительное число;

m - количество позиций, отводимых для целой части;

n - количество позиций, отводимых для дробной части.

Например, если мы хотим вывести на экран число X с фиксированной точкой, причем знаем, что для вывода целой части этого числа достаточно 7 мест, а вывод дробной части ограничим сотыми, то мы запишем вывод так:

Write (X:7:2)

Символьный тип (char):

Значениями данного типа является множество всех символов компьютера: русская или латинская большая или маленькая буква, цифра, знак препинания, специальный знак (например, "+", "-", "*", "/", "", "=" и др.) или пробел " ". Каждый из символов имеет уникальный номер от 0 до 255, т. е. внутренний код, который возвращает функция ORD. Символьная константа или символьная переменная - любой символ языка, заключённый в апострофы.

Например,

Var Simvol : char;

Строковый тип (string):

Значением строковой величины является строка переменной длины (быть может пустая). Строковая константа или строковая переменная представляет

собой произвольную последовательность символов, заключенную в апострофы.

Например,

Var Stroka : string;

Логический тип (boolean):

Логический тип данных часто называют булевым по имени английского математика Д. Буля, создателя математической логики. В языке Паскаль имеются две логические константы TRUE и FALSE. Логическая переменная принимает одно из этих значений и имеет тип Boolean.

Для сравнения данных предусмотрены следующие операции отношений: <, <=, =, <>, >, >=. А также существуют специфичные для этого типа логические операции OR - или; AND - и; NOT - не.

При проверке некоторых условий результат операции может быть истинным или ложным. Например, 3>5 ложь.

Более подробно этот тип данных мы рассмотрим при изучении условного оператора.

Сложные типы:

К сложным или структурированным типам относятся массивы, записи, множества, которые требуют специального изучения и здесь рассматриваться не будут.

Создадим программу, в которой опишем несколько переменных разного типа, введем в них значения и выведем на экран.

```
Program TipDann;

Uses Crt;

Var

  X : Integer;

  У : Real;

  Simvol : Char;

  Stroka : String;
```

```
  Logika : Boolean;
Begin
  ClrScr;
  X:=12;
  У:=X*2;
  У:=У/5;
  Simvol:='d';
  Stroka:='Строчка';
  Logika:= X> У;
  WriteLn ('Вывод значений:');
  WriteLn ('Значение переменной X : ',X);
  WriteLn ('Значение переменной У : ',У:5:2);
  WriteLn ('Значение переменной Simvol : ',Simvol);
  WriteLn ('Значение переменной Stroka : ',Stroka);
  WriteLn ('Значение переменной Logika : ',Logika);
End.
```

Внимательно рассмотрите каждую строчку программы. Обратите особое внимание на описание переменных: X - переменная целого типа, У - действительного, Simvol - символьного, Stroka - строкового, Logika - логического. Далее в основной программе идет присвоение переменной X целого числа 12, переменной У - целого числа 24. Обратим внимание, что переменной действительного типа У присвоено целое число; никакой ошибки нет, т. к. множество целых чисел является подмножеством множества действительных чисел. Следующая строчка еще более интересная: переменной У присваивается значение той же переменной, только деленной на 5.

Такое присваивание используют в программах, если предыдущее значение этой переменной уже не понадобится и для более рационального использования описанных переменных. Для того чтобы переменной символьного типа присвоить какой-либо символ (например, d), надо этот

символ записать в апострофах (знак «’»). Аналогично поступают с переменными строкового типа (смотри следующую строку программы). А про переменные логического типа мы знаем, что им можно присваивать только два значения: True и False. В этой программе мы присвоим значение результата сравнения двух переменных, здесь оно будет равно True. А теперь выведем присвоенные значения на экран используя оператор WriteLn.

Ввод - вывод.

Операторы Read (Readln), Write (Writeln).

Простейшие линейные программы.

Решим задачу, прокомментировав каждое свое действие в фигурных скобках. Напомним, что комментарий не воспринимается компьютером, а нам он нужен для того, чтобы лучше понять как работает программа.

Задача. Напишите программу, которая бы очищала экран и вычисляла произведение двух чисел, вводимых пользователем.

```
Program Proizv2;
Uses Crt;                                   {Подключаем модуль Crt}
Var
  n1,                 {переменная, в которой будет содержаться первое число}
  n2,                 {переменная, в которой будет содержаться второе число}
  rez                    {переменная, в которой будет содержаться результат}
  : integer;
Begin
  ClrScr;                {Используем процедуру очистки экрана из модуля Crt}
  Write ('Введите первое число ');
                  {Выводим на экран символы, записанные между апострофами}
  Readln (n1);
                   {Введенное пользователем число считываем в переменную n1}
  Write ('Введите второе число ');
                  {Выводим на экран символы, записанные между апострофами}
  Readln (n2);
                   {Введенное пользователем число считываем в переменную n2}
  rez := n1 * n2;
     {Находим произведение введенных чисел и присваиваем переменной rez}
```

```
  Write ('Произведение чисел ', n1, ' и ', n2, ' равно ', rez);
                                {Выводим на экран строчку, содержащую ответ задачи}
  Readln;{Процедура задержки экрана}
End.
```

Операторы Write и WriteLn

Write (англ. писать) - оператор, который используется для вывода информации на экран.

Оператор WriteLn выполняет то же самое действие, но так как у него есть еще окончание Ln (line - англ. линия, строка), то после вывода на экран нужного сообщения, он дополнительно переводит курсор на следующую строчку.

Общий вид:

```
  Write (список выражений)
  WriteLn (список выражений)
```

Например:

```
WriteLn('При x=',x,' Значение  y=',y);
WriteLn(' При X=',X:7:2,' Значение Y=',Y:8:4);
```

Процедуры Write и WriteLn используются не только для вывода результата, но и для вывода различных сообщений или запросов. Это позволяет вести диалог с пользователем, сообщать ему, когда ему нужно ввести значения , когда он получает результат, когда он ошибся и др.

Например, при выполнении процедуры WriteLn('Найденное число ',a), будет напечатана строчка, заключенная в апострофы, а затем выведено значение переменной a.

Оператор WriteLn можно применить и без параметров. В этом случае напечатается строка, состоящая из пробелов, и курсор будет переведен на другую строку. Это иногда нам нужно для лучшего восприятия ввода данных.

Операторы Read и ReadLn

Вспомним, что основное назначение ЭВМ - сэкономить человеческий труд. Поэтому необходимо обеспечить возможность, однажды написав программу, многократно ее использовать, вводя каждый раз другие данные. Такая гибкость в языке обеспечивается операторами Read и ReadLn. Этими операторами вводится информация с клавиатуры.

Общий вид:

Read(переменная, переменная...)

ReadLn(переменная, переменная...)

При выполнении процедуры Read ожидается ввод перечисленных в скобках значений. Вводимые данные нужно отделить друг от друга пробелами. Присваивание значений идет по очереди.

Например, если вводятся значения 53 и X, то при выполнении оператора Read(a,b) переменной a будет присвоено число 53, а переменной X - буква X. Причем, отметим, чтобы не было аварийной ситуации, нужно правильно определить тип данных в разделе Var; в нашем случае a:integer, a b:char.

Особых различий при чтении и записи в использовании операторов Read и ReadLn нет. Часто процедуру ReadLn без параметров применяют в конце программы для задержки: до нажатия на клавишу <Enter> результат выполнения программы остается на экране. Это очень полезно делать для анализа результатов.

Когда Вы ставите задержку экрана, обратите внимание на предыдущий ввод. Если данные запрашивались процедурой Read задержки не будет.

Решим задачу, в которой рассмотрим все возможные употребления этих процедур.

Найти среднее значение трех чисел.

Чтобы найти среднее значение нескольких чисел, нужно сложить эти числа и сумму разделить на количество этих чисел.

```
Program Srednee;

Uses  Crt;

Var A, B, C : integer;

      Sum : real;
```

```
Begin
  ClrScr;
  Write (‘Введите первое число ‘);
  ReadLn(A);
  Write (‘Введите второе и третье числа через пробел ‘);
  ReadLn(B, C);
  Sum := A + B + C;
  Sum := Sum/3;
  Write (‘Среднее значение ‘, A, ‘, ‘,B, ‘ и ‘, B, ‘ равно ‘, Sum:5:2);
  ReadLn;
End.
```

Имя программы Srednee отражает содержание задачи. Имя программы и имя файла, котоый содержит эту программу, не должны совпадать. Далее идет подключение модуля Crt. В разделе Var описаны A, B, C как переменные целого типа, а Sum - действительного типа. Раздел операторов начинается со стандартной процедуры очистки экрана ClrScr (Clear Screen), которая находится в модуле Crt. Далее оператором Write мы выводим на экран сообщение ‘Введите первое число ‘, получив которое пользователь должен ввести число.

Теперь компьютер должен считать введенные символы и занести их в переменную A, это произойдет при выполнении следующего оператора ReadLn(A). Затем с помощью оператора Write запрашиваем значения еще двух чисел и считываем их в переменные B и C. Затем вычисляем их сумму и присваиваем полученное число переменной Sum. Чтобы найти среднее, нужно теперь полученное число разделить на 3 и сохранить результат в какой-либо переменной.

Совсем не обязательно описывать еще одну переменную для сохранения результата. Можно, как в нашей программе, значение переменной Sum разделить на 3 и результат опять присвоить той же переменной Sum. Теперь можно вывести результат вычислений на экран с помощью процедуры Write. И, наконец, последняя процедура ReadLn задержит наш вывод на экране до нажатия на клавишу.

Нажмите клавиши <Ctrl>+<F9>. Введите значения переменных 5, 7 и 12, на экране увидите следующее:

Среднее значение 5, 7 и 12 равно 8.00

Операторы условия и выбора.

Разветвляющиеся алгоритмы.

Оператор условия If.

Разветвляющимся называется такой алгоритм, в котором выбирается один из нескольких возможных вариантов вычислительного процесса. Каждый подобный путь называется ветвью алгоритма.

Признаком разветвляющегося алгоритма является наличие операций проверки условия. Различают два вида условий - простые и составные.

Простым условием (отношением) называется выражение, составленное из двух арифметических выражений или двух текстовых величин (иначе их еще называют операндами), связанных одним из знаков:

< - меньше, чем...

> - больше, чем...

<= - меньше, чем... или равно

>= - больше, чем... или равно

<> - не равно

= - равно

Например, простыми отношениями являются следующие:

x-y>10; k<=sqr(c)+abs(a+b); 9<>11; ‘мама’<>‘папа’.

Выражения, при подстановке в которые некоторых значений переменных, о нем можно сказать истинно (верно) оно или ложно (неверно) называются булевыми (логическими) выражениями.

Название “булевые” произошло от имени математика Джорджа Буля, разработавшего в XIX веке булевую логику и алгебру логики.

Переменная, которая может принимать одно из двух значений: True (правда) или False (ложь), называется булевой (логической) переменной.

Например,

K:=True;

Flag:=False;

Second:=a+sqr(x)>t

Вычислить значение модуля и квадратного корня из выражения (x-y).

Для решения этой задачи нужны уже знакомые нам стандартные функции нахождения квадратного корня - Sqr и модуля - Abs. Поэтому Вы уже можете записать следующие операторы присваивания:

Koren:=Sqrt(x-y);

Modul:=Abs(x-y).

В этом случае программа будет иметь вид:

```
Program Znachenia;

Uses Crt;

Var

  x, y : integer;

  Koren, Modul : real;

Begin

  ClrScr;

  write (‘Введите значения переменных x и y через пробел ‘);

  readln (x, y);

  Koren:=Sqrt(x-y);

  Modul:=Abs(x-y).

  write (‘Значение квадратного корня из выражения (x-y) равно ‘);

  write (‘Значение модуля выражения (x-y) равно ‘);

  readln;

End.
```

Казалось бы задача решена. Но мы не учли области допустимых значений для нахождения квадратного корня и модуля. Из курса математики Вы должны знать, что можно найти модуль любого числа, а вот значение

подкоренного выражения должно быть неотрицательно (больше или равно нулю).

Поэтому наша программа имеет свою допустимую область исходных данных. Найдем эту область. Для этого запишем неравенство x-y>=0 и решив его получим x>=y. Значит, если пользователем нашей программы будут введены такие числа, что при подстановке значение этого неравенства будет равно True, то квадратный корень из выражения (x-y) извлечь можно. А если значение неравенства будет равно False, то выполнение программы закончится аварийно.

Каждая программа, насколько это возможно, должна осуществлять контроль за допустимостью величин, участвующих в вычислениях. Здесь мы сталкиваемся с разветвлением нашего алгоритма в зависимости от условия. Для реализации таких условных переходов в языке Паскаль используют операторы If и Else, а также оператор безусловного перехода Goto.

Оператор If.

IF <ЛОГИЧЕСКОЕ ВЫРАЖЕНИЕ> THEN <ОПЕРАТОР 1>

ELSE <ОПЕРАТОР 2>;

если УСЛОВИЕ истинно или выполнимо, ,то выполнить ОПЕРАТОР 1,

иначе выполнить ОПЕРАТОР2.

В зависимости от введенных значений переменных x и y, условия могут выполняться или не выполняться.

Условный оператор работает по следующему алгоритму.

Сначала вычисляется значение логического выражения, расположенного за служебным словом IF. Если его результат истина, выполняется <оператор 1>, расположенный после слова THEN, а действия после ELSE пропускаются; если результат ложь, то, наоборот, действия после слова THEN пропускаются, а после ELSE выполняется <оператор 2>.

Управляющая структура if может показаться негибкой, так как выполняемые действия могут быть описаны только одним оператором. Иногда может потребоваться выполнение последовательности операторов. В этом случае хотелось бы заключить всю последовательность в воображаемые скобки. В Паскале предусмотрен этот случай.

Если в качестве оператора должна выполниться серия операторов, то они заключаются в операторные скобки begin-end. Конструкция Begin ... End называется составным оператором.

```
if <логическое выражение> then   begin
                                        оператор 1;
                                        оператор 2;
                                 end
                          else
                                 begin
                                        оператор 1;
                                        оператор 2;
    ..                           end;
```

Составной оператор - объединение нескольких операторов в одну группу. Группа операторов внутри составного оператора заключается в операторные скобки (begin-end).

```
begin
  оператор 1;
  оператор 2;
end;
```

С учетом полученных знаний преобразуем нашу программу.

Составным оператором является и такой оператор

```
begin
  S:=0;
end.
```

Символ “;” в данном случае разделяет оператор присваивания S:=0 и пустой оператор.

Пустой оператор не влечет никаких действий и в записи программы никак не обозначается.

Например, составной оператор

begin

end.

включает лишь один пустой оператор.

Если Вы обратили внимание, программа на языке Паскаль всегда содержит один составной оператор - раздел операторов программы.

Перед служебным словом Else разделитель (точка с запятой) не ставится.

Отметим, что большинство операторов в программах на языке Паскаль заканчиваются точкой с запятой, но после некоторых операторов точка с запятой не ставится. Сформулируем общие правила употребления точки с запятой:

-Каждое описание переменной и определение константы заканчиваются точкой с запятой.

-Каждый оператор в теле программы завершается точкой с запятой, если сразу за ним не следуют зарезервированные слова End, Else, Until.

-После определенных зарезервированных слов, таких, как Then, Else, Var, Const, Begin, никогда не ставится точка с запятой.

Вывести на экран большее из двух данных чисел.

```
Program Example1;
Var
  x, y : integer; {вводимые числа}
Begin
  writeln(‘Введите 2 числа ‘); {вводим два целых числа через пробел}
  readln(x,y);
  if x>y
    then
```

```
    writeln (x) {если x больше y, то выводим x}
  else
    writeln (y) {иначе выводим y}
  readln;
End.
```

Можно также использовать и сокращенную (неполную) форму записи условного оператора. Эта форма используется тогда, когда в случае невыполнения условия ничего делать не надо.

Неполная форма условного оператора имеет следующий вид.

if <логическое выражение> then <оператор>;

Тогда если выражение, расположенное за служебным словом IF. в результате дает истину, выполняются действия после слова THEN, в противном случае эти действия пропускаются.

Составить программу, которая, если введенное число отрицательное меняет его на противоположное.

```
Program Chisla;
Var
 x : integer; {вводимое число}
Begin
 writeln('Введите число '); {вводим целое число}
 readln(x);
 if x<0
   then
    x:=-x;
    writeln (x);
    readln;
End.
```

Логический тип данных.

Логические операции not, and, or.

Нахождение значений логических выражений.

Переменные логического типа описываются посредством идентификатора Boolean. Они могут принимать только два значения - False (ложь) и True (истина). Описываются они также в разделе описания переменных.

```
Var
  Flag : Boolean;
```

Переменные логического типа обычно получают значения в результате выполнения операций сравнения и математических операций (рассматривались в предыдущем занятии), а также с помощью специфических булевых операций.

В языке Турбо Паскаль имеются логические операции, применяемые к переменным логического типа. Это операции not, and, or и xor. В этой теме Вы рассмотрите три логические операции. Обозначения и результаты этих операций приведены в таблице. Рассмотрите ее.

Операция not (не) имеет один операнд и образует его логическое отрицание. Результат операции not есть False, если операнд истинен, и True, если операнд имеет значение ложь. Так,

not True False (неправда есть ложь)

not False True (неложь есть правда)

Результат операции and (и) есть истина, только если оба ее операнда истинны, и ложь во всех других случаях.

Результат операции or (или) есть истина, если какой-либо из ее операндов истинен, и ложен только тогда, когда оба операнда ложны.

Логические операции, операции отношения и арифметические операции часто встречаются в одном выражении. При этом отношения, стоящие слева

и справа от знака логической операции, должны быть заключены в скобки, поскольку логические операции имеют более высокий приоритет. Вообще принят следующий приоритет операций:

- not
- and, *, /, div, mod
- or, +, -
- операции отношения..

Логическую операцию and еще называют логическим умножением, а логическую операцию or - логическим сложением.

Кроме того, порядок выполнения операций может изменяться скобками. Например, в логическом выражении расставим порядок действийй

<="" p="">

A or B and not (A or B)

Сначала выполняется заключенная в скобки операция or, а затем операции not, and, or. Если подставить вместо переменных A и B значения True и False, то, используя уже рассмотренный порядок действий, получим значение всего выражения равное True.

Вычислите значения выражений при a=10, b=20, c=true, d=false:

- (a>5) and (b>5) and (a<20) and (b<30);
- not (a<15) or not (b<30);
- c or d and (b=20);

В языке Паскаль нет возможности ввода логических данных с помощью оператора read. Однако предусмотрен вывод значений переменных логического типа с помощью оператора write.

Например, после выполнения оператора write (5>2) на экран будет выведено True.

Вложенные условные операторы.

При решении задач часто приходится рассматривать не два, а большее количество вариантов. Это можно реализовать, используя несколько

условных операторов. В этом случае после служебных слов Then и Else записывается новый условный оператор. Рассмотрим пример.

Вычислить значение функции:

Для решения этой задачи рассмотрим координатную прямую, на которой отметим промежутки, на которые разбиваются все значения переменной x.

Начнем записывать условный оператор:

```
если x>0
 то
   вычислить y по формуле y=x-12
 иначе
```

Что же должно выполниться в случае иначе? На эту ветку оператора попадают все не положительные значения x. Если бы для этих чисел нужно было бы выполнить один и тот же оператор (или группу операторов), то проблемы бы не стояло. Но нам нужно этот промежуток разделить еще на две части (отрицательные и ноль), и для части выполнить свой оператор. Поэтому ветка Иначе будет содержать еще один условный оператор и наш вложенный условный оператор будет иметь вид:

```
если x>0
 то
   y вычислить по формуле y=x-12
 иначе
   если x=0
     то
       y вычислить по формуле y=5
     иначе
       y вычислить по формуле y=sqr(x);
```

Тогда фрагмент программы для решения этой задачи будет выглядеть так:

```
if x>0
  then
    y := x-12
  else
    if x=0
      then
        y := 5
      else
        y := sqr(x);
```

Итак, когда оператор if появляется внутри другого оператора if, они считаются вложенными. Такое вложение используется для уменьшения числа необходимых проверок. Этот метод часто обеспечивает большую эффективность, однако одновременно он уменьшает наглядность программы. Не рекомендуется использовать более одного-двух уровней вложения if. За вторым уровнем вложения становится трудно восстановить последовательность проверки условий каждым условным оператором.

Если часть else используется во вложенных if, то каждое else соответствует тому if, которое ему непосредственно предшествует. Таким образом, при определении последовательности выполнения фрагментов нет двусмысленности.

Рассмотрите еще один пример.

Даны целые числа a, b, c. Если $a \le b \le c$, то все числа заменить их квадратами, усли a>b>c, то каждое число заменить наибольшим из них, в противном случае сменить знак каждого числа.

Для решения этой задачи перепишем условие задачи следующим образом:

a:=a2, b:=b2, c:=c2, если $a \le b \le c$

b:=a, c:=a, если a>b>c

a:=-a, b:=-b, c:=-c, в остальных случаях

Программа для решения этой задачи представлена ниже.

```
Program Example3;
Var
  a, b, c : integer;
Begin
  writeln('Введите числа a, b, c');
  readln(a,b,c);
  if (a<=b) and (b<=c)
    then
      begin
        a:=sqr(a);
        b:=sqr(b);
        c:=sqr(c);
      end
    else
      if (a>b) and (b>c)
        then
          begin
            b:=a;
             c:=a;
          end
        else
          begin
            a:=-a;
            b:=-b;
```

```
            c:=-c;
          end
  writeln(a,b,c);
  readln;
End.
```

Оператор выбора CASE

Оператор If позволяет программе выполнять переходы на ту или иную ветвь по значению булева условия. Используя несколько операторов If, можно производить ветвление по последовательности условий. В приведенном фрагменте показано, как при помощи ряда операторов If можно преобразовать целое число (в диапазоне 0-9) к его словесному представлению:

```
if Ziphra = 0
  then
    write (‘Нуль‘);
     if Ziphra = 1
       then
         write (‘Единица‘);
          if Ziphra = 2
            then
             write (‘Два‘);
```

и т.д.

Язык Паскаль предоставляет для этих целей другую управляющую структуру (оператор выбора case), которая позволяет построить ветвление по ряду условий в форме, более удобной для чтения программ.

Оператор выбора позволяет выбрать одно из нескольких возможных продолжений программы. Параметром, по которому осуществляется выбор, служит так называемый ключ выбора (или селектор) - выражение любого типа (кроме типов REAL и STRING).

Общая форма записи следующая:

```
CASE выражение OF
     значение1 : оператор (группа операторов);
     значение2 : оператор (группа операторов);
. . . . . .
```

значениеN : оператор (группа операторов)

ELSE оператор (группа операторов);

END;

Оператор выбора работает следующим образом. Сначала вычисляется значение выражения, стоящее после зарезервированного слова case, а затем выполняется оператор (или составной оператор), соответствующий результату вычисления выражения.

Может случиться, что в списке выбора не окажется константы равной вычисленному значению ключа. В этом случае управление передается оператору, стоящему за словом ELSE.

Например,

```
case NUMBER mod 2 of
  0 : writeln (NUMBER, '- число четное')
else : writeln (NUMBER, '- число нечетное');
end;
```

Если один оператор выполняется при нескольких значениях, то их можно перечислить через запятую.

```
case MONTH of
  1, 2, 3 : writeln ('Первый квартал');
  4, 5, 6 : writeln ('Второй квартал');
  7, 8, 9 : writeln ('Третий квартал');
  10, 11, 12 : writeln ('Четвёртый квартал');
end;
```

Оператором может являться не только простой оператор, но также составной и пустой операторы.

```
case CODE of
  1 : for i := 1 to 5 do
        writeln ('*******');
```

```
  2 : begin {составной оператор}
        x:=sqr(y-1);
        writeln (x);
      end;
  3 : {пустой оператор}
      end;
```

Любому заданному значению селектора соответствует лишь один вход в списке операторов. Константы должны принадлежать тому же типу, что и селектор. Если селектор принимает значение, которому не соответствует ни один вход, то будет выполняться оператор, следующий за словом else. Если же этого оператора нет, то никакие альтернативы не будут выполняться.

Если оператор должен выполняться при нескольких значениях селектора следующих друг за другом, образуя некоторый промежуток, то это можно записать в более сжатой форме. Например,

```
case Chislo of
  0..9 : write (‘Это число является цифрой‘);
```

Посмотрите, в каких вариантах еще можно использовать оператор выбора при решении задачи.

Написать программу преобразования цифр в слова.

```
Program Number1;
Var
  a, b, c : integer;
Begin
  writeln(‘Введите цифру ‘);
  readln(a);
  if (a<0) or (a>9)
    then
      writeln (‘Это число не является цифрой‘)
```

```
    else
      case a of
        0 : writeln (‘ноль‘);
        1 : writeln (‘один‘);
        2 : writeln (‘два‘);
        3 : writeln (‘три‘);
        4 : writeln (‘четыре‘);
        5 : writeln (‘пять‘);
        6 : writeln (‘шесть‘);
        7 : writeln (‘семь‘);
        8 : writeln (‘восемь‘);
        9 : writeln (‘девять‘);
      end;
  readln;
End.


Program Number2;
Var
  a, b, c : integer;
Begin
  writeln(‘Введите цифру ‘);
  readln(a);
  case a of
    0 : writeln (‘ноль‘);
    1 : writeln (‘один‘);
```

```
  2 : writeln (‘два‘);
  3 : writeln (‘три‘);
  4 : writeln (‘четыре‘);
  5 : writeln (‘пять‘);
  6 : writeln (‘шесть‘);
  7 : writeln (‘семь‘);
  8 : writeln (‘восемь‘);
  9 : writeln (‘девять‘)
 else writeln (‘Это число не является цифрой‘);
 end;
 readln;
End.
```

Понятие массива.

Одномерные массивы

Для решения многих задач удобно сначала упорядочить данные по определенному признаку, так можно ускорить поиск некоторого объекта. Например, в преферансе игроки раскладывают карты по мастям и по значению. Так легче определить, каких карт не хватает. Или возьмем любой энциклопедический словарь - статьи в нем упорядочены в алфавитном порядке.

Перегруппирование заданного множества объектов в определенном порядке называют сортировкой.

Почему сортировке уделяется большое внимание? Вы это поймете, прочитав цитаты двух великих людей.

"Даже если бы сортировка была почти бесполезна, нашлась бы масса причин заняться ею! Изобретательные методы сортировки говорят о том, что она и сама по себе интересна как объект исследования." /Д. Кнут/

"Создается впечатление, что можно построить целый курс программирования, выбирая примеры только из задач сортировки." /Н. Вирт/

Отличительной особенностью сортировки является то обстоятельство, что эффективность алгоритмов, реализующих ее, прямо пропорциональна сложности понимания этого алгоритма. Другими словами, чем легче для понимания метод сортировки массива, тем ниже его эффективность.

Сегодня существует множество методов сортировки, но для понимания сути сортировки рассмотрим некоторые из них.

Но прежде чем перейти к рассмотрению конкретного алгоритма той или иной сортировки немного вспомним материал, который пригодится нам в дальнейшем.

Составить фрагмент программы поиска максимального числа из трех введенных с клавиатуры чисел.

Пусть a, b, c - вводимые с клавиатуры числа, Max - максимальное из их значений. На первом шаге предположим , что a - максимальное из чисел и поэтому Max:=a. Затем сравним значение предполагаемого максимума со значениями переменных b и c. Если значение m окажется меньше, чем значение очередной переменной, то переопределим значение максимума.

```
. . .
m:=a;
if m<b
  then
    m:=b;
if m<c
  then
    m:=c;
. . .
```

Дан массив a, состоящий из 10 элементов. Составить программу поиска максимального элемента массива.

Используем идею предыдущей задачи. Перед началом поиска выберем условно в качестве максимального первый элемент массива Max:=a[1]. Затем по очереди каждый элемент массива сравним со значением переменной m. Если он окажется больше, то изменим значение Max. После анализа всех элементов массива переменная Max содержит значение максимального элемента массива.

```
. . .
Max:=a[1];
for i := 2 to 10 do
  if Max<a[i]
    then
      Max := a[i];
. . .
```

До написания программ Вам необходимо выполнить некоторую подготовительную работу - написать шаблон программы следующего содержания:

```
Program Sorting;
```

```
Const
  n = ... ; {количество элементов в массиве}
Type
  TArray = array [1..n] of integer;

Procedure FillArray (Var a: TArray);
Var
  i: integer;
Begin
  for i: = 1 to n do
    a [i] := Random(100);
End; {конец процедуры}

Procedure PrintArray (a: TArray);
Var  i: integer;
Begin
  for i: = 1 to n do
    write (a [i]: 3, ' ');
    writeln;
End;

Begin {Главная программа}
  writeln('Сортировка МЕТОДОМ . . .');
  writeln('Заполняем исходный массив: ');
  FillArray (a);
```

```
  PrintArray (a);
  AnySort (a, b);{имя процедуры, реализующей данный метод}
  writeln('Отсортированный массив: ');
  PrintArray (b);
End.
```

Доступ к элементам массива.

Заменить отрицательные элементы на противоположные по знаку.

Для этого опишем процедуру. Ей будем передавать один параметр - массив, который будет результатом ее выполнения, так как некоторые элементы могут быть заменены.

```
Procedure Zamena (Var m : MyArray; n:integer);
Var i : integer;
Begin
  for i := 1 to n do
    if m[i] < 0  then
       m[i] := -1*m[i];
End;
```

Найти и вывести на экран номера четных элементов.

Для решения задачи необходимо просмотреть весь массив, и если просматриваемый элемент является четным, то выводить его номер. Опишем процедуру, которой передается данный массив и выводятся нужные номера.

```
Procedure PoiskChet(m : MyArray; n:integer);
Var i : integer;
Begin
  for i := 1 to n do
    if m[i] mod 2 =0  then Write(i:5);
```

```
End;
```

Найти количество положительных и отрицательных элементов в данном массиве.

Опишем процедуру, которой будем отправлять три параметра - массив и два счетчика, один для элементов, больших нуля, а второй - для отрицательных элементов.

```
Procedure OtrPol(m : MyArray; ; n:integer; Var k1,k2 : Integer);

Var  i : integer;

Begin

  k1 :=0;

  k2 :=0;

  for i := 1 to n do

    if m[i] > 0  then  Inc(k1)

                  else  if m[i] < 0 then  Inc(k2);

End;
```

Есть ли отрицательный элемент в массиве?

Начинаем с первого элемента (i=1). пока не просмотрен последний элемент (i<=n) и не найден отрицательный (m[i]>=0), будем переходить к следующему (Inc(i)). Таким образом, мы закончим просмотр массива в одном из двух случаев: первый - просмотрели все элементы и не нашли отрицательный, тогда i>n, второй - нашли нужный, при этом i<=n. Опишем функцию, значение которой истина (True), если такой элемент есть, и ложь (False), если его нет.

```
Function Control (m : MyArray; n:integer) : Boolean;

Var  i : integer;

Begin

  i := 1;

  while (i<=n) and (m[i]>0) do

    Inc(i);
```

```
  Control := (i<=n);
End;
```

Удалить из массива максимальный элемент, если все элементы разные.

Для того, чтобы решить задачу нужно:

-найти номер максимального элемента k;

-сдвинуть все элементы, начиная с k-го, на один элемент влево;

-последнему элементу присвоить значение 0;

-уменьшить количество элементов массива на единицу.

Рассмотрим задачу на конкретном примере. Пусть дан одномерный массив из целых чисел, состоящий из 10 элементов:

6, 3, 4, 7, 11, 2, 13, 8, 1, 5.

Номер максимального элемента равен 7 (k=7), то есть, начиная с 7-го элемента, будем сдвигать элементы на один влево: 7-му присвоим значение 8-го, 8-му присвоим значение 9-го, 9-му присвоим значение 10-го, на этом сдвиг заканчивается. Таким образом, сдвиг начинается с k-го элемента и идет по (n-1)-й (где n - количество элементов в массиве). После этого последнему элементу присвоим значение, равное 0, и тогда массив будет следующим:

6, 3, 4, 7, 11, 2, 8, 1, 5, 0.

При удалении элемента размерность массива не изменяется.

Составим программу, удаляющую максимальный элемент из одномерного массива. В программе опустим уже знакомые Вам процедуры заполнения массива и вывода элементов массива на экран. Чтобы последний элемент не выводился, модифицируйте соответствующую процедуру таким образом, чтобы ей передавать не только массив, но и количество элементов, которые надо вывести, начиная с первого.

```
Program DeleteK;
Const  n=30; dd=51;
Type  MyArray = Array [1..n] of Integer;
Var A : MyArray;
```

```
      k : Integer;

Procedure InsertMas1(Var m : MyArray; n : integer);
  . . .

Procedure InsertMas2(Var m : MyArray; n : integer);
  . . .

Procedure PrintMas(m : MyArray; n : integer);
  . . .

Function Maximum (m : MyArray; n:integer) : Integer;
Var  i, max, maxi : integer;
Begin
  max:=-maxint; {начальным значением переменной будет наименьшее
значение данного типа}
  for i := 1 to n do         {просматриваем все элементы массива}
    if m[i] > max       {если найден элемент больше, чем мы считаем максимал}
                then  begin
                        max:=A[i]; {то запомним найденное значение}
                         maxi:=i; {а также место, на котором он стоит в массиве}
                         end;
     Maximum := maxi; {имени функции присвоим найденный результат}
End;

Procedure Delete(Var m : MyArray; Var n:integer; k1 : integer);
```

```
Var  i : integer;
Begin
  for i := 1 to n-1 do
    m[i] := m[i+1];
    m[n]:=0;
    Dec(n);
End;


Begin
  . . .
  k:=Maximum(A,m);
  Delete(A,m,k);
  . . .
End.
```

Вставка элементов в одномерный массив.

Вставка одного элемента

Вставлять элемент можно до или после данного элемента, номер этого элемента можно вводить с клавиатуры или искать при определенных условиях.

Пусть k - это номер элемента, после которого мы должны вставить элемент x. Тогда вставка осуществляется следующим образом:

-первые k элементов массива остаются без изменения,

-все элементы, начиная с (k+1)-го, необходимо сдвинуть на один назад,

-на место (k+1)-го элемента записываем значение x;

-увеличить количество элементов в массиве на единицу.

Вставить число 100 после пятого элемента массива.

Рассмотрим на конкретном примере. Пусть задан следующий одномерный массив из N (N=10) элементов:

3, -12, 5, 14, 27, -6, 1, -34, 10, -15.

Надо вставить 100 после пятого элемента массива, т. е. должен получиться следующий массив:

3, -12, 5, 14, 27, 100, -6, 1, -34, 10, -15.

Таким образом, в массиве стало 11 элементов, то есть массив надо определять на N+1 элемент:

```
Type  MyArray = array[1..n+1] of integer
```

Кроме того, в программе необходимо выводить массив два раза, сначала первые N элементов массива, а затем все N+1 элементы.

Рассмотрите процедуру вставки Insert1(m, n, Mesto, Element), которой передаются:

m - массив, в котором делаем преобразования;

n - количество элементов в массиве.

Mesto - номер элемента, после которого надо вставить данный,

Element - число, которое вставляем,

Кроме того, сдвиг элементов будем начинать с последнего элемента.

```
Program Vstavka1;

Const

  n=10; dd=51;

Type

  MyArray = array [1..n+1] of integer;

Var

  A : MyArray;

  k, x : Integer;
```

```
Procedure InsertMas1(Var m : MyArray; n : integer);
  . . .

Procedure InsertMas2(Var m : MyArray; n : integer);
  . . .

Procedure PrintMas(m : MyArray; n : integer);
  . . .

Procedure Insert1(Var m : MyArray; Var n : integer; Mesto, Element : integer);
Var  i : integer;
Begin
  for i := n downto Mesto+1 do
    m[i+1] := m[i];
    m[Mesto+1]:= Element;
    Inc(n);
End;

Begin
  . . .
  Writeln('Номер элемента, после которого вставлять > ');
  Readln(k);
  Writeln('Вставляемое число > ');
  Readln(x);
  Insert1(A, n, k, x);
```

```
  . . .
End.
```

Вставить число после всех элементов массива, кратных трем.

На сколько элементов может увеличиться массив? Максимальное количество элементов, после которых будет вставлен новый элемент, совпадает с количеством элементов массива, так как может случиться, что все элементы массива отвечают заданному свойству. Поэтому массив может увеличиться в два раза, а значит, соответствующее ему описание будет следующим:

```
Type  MyArray[1..2*n] of Integer;
```

Если мы будем просматривать массив с начала и вставлять новый после элемента с заданным свойством, то номер последнего элемента каждый раз может меняться, кроме того, будет просматриваться и новый (вставленный) элемент и его необходимо будет пропускать, поэтому решение будет не очень эффективным. Лучше всего просматривать массив, начиная с конца, тогда вставляемый элемент мешать не будет. Кроме того, номер последнего элемента можно будет знать (если знать, сколько элементов вставлено на данный момент), при этом просмотр будет последовательным от N-го до 1-го.

```
Program VstavkaN;

Const  n=10; dd=51;

Type MyArray = Array [1..2*n] of Integer;

Var A : MyArray;

      k, x, i : Integer;


Procedure InsertMas1(Var m : MyArray; n : integer);

  . . .


Procedure InsertMas2(Var m : MyArray; n : integer);

  . . .
```

```
Procedure PrintMas(m : MyArray; n : integer);
  . . .

Procedure InsertN(Var m : MyArray; Var n : integer; Mesto, Element : Integer;);
Var  i : Integer;
Begin
  for i := n downto Mesto+1 do
    m[i+1] := m[i];
    m[Mesto+1]:= Element;
    Inc[n];
End;

Begin
  . . .
  Writeln('Вставляемое число > ');
  Readln(x);
  k:=0;
  for i:=n downto 1 do
    if A[i] mod 3=0  then  InsertN(A, n, i, x);
  . . .
End.
```

Поменять местами два элемента массива с номерами k1 и k2.

Рассмотрите процедуру, с помощью которой эта задача легко решается.

```
Procedure Obmen2(Var m : MyArray; n, k1, k2 : integer;);
Var  x : integer;
```

```
Begin
  x:=m[k1];
  m[k1] := m[k2];
  m[k2] := x;
End;
```

Перестановка части массива

Дан одномерный массив A, состоящий из 2n элементов. Поменять местами первую и вторую его половины. Оформите решение этой задачи, применив процедуру обмена значений Obmen2, рассмотренную выше.

Заметим лишь, что должны поменять местами элементы с номерами 1 и n+1, 2 и n+2 и т.д., последняя пара - n и 2n, а значит, обмен происходит по правилу: элемент с номером i меняется местами с элементом с номером n+i. Эту закономерность следует применить в организации обращения к процедуре обмена. Например, так

```
for i := 1 to n do
  Obmen2(A, 2*n, i, i+n,);
```

Найти скалярное произведение двух массивов.

Скалярным произведением двух массивов одинаковой размерности называется сумма произведений соответствующих элементов. Это можно записать так:

a[1]*b[1] + a[2]*b[2] + ... + a[n-1]*b[n-1] + a[n]*b[n],

где n - это количество элементов в массивах (размерность).

Тогда можно составить следующую функцию:

```
Function Sp (a, b : MyArray; n ; integer) : LongInt;
Var  i : Integer;
       s : LongInt;
Begin
  s:= 0;
```

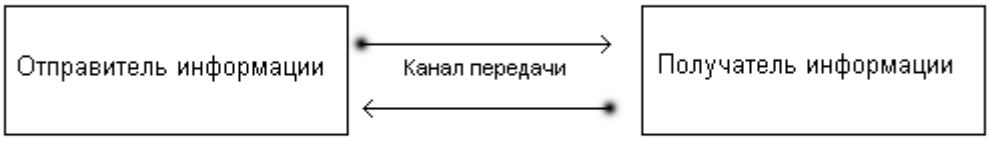
```
  for i := 1 to n do
    s := s+a[i]*b[i];
    Sp := s;
End;
```

Коммуникационные технологии

1.Основные понятия.

В основе коммуникационных технологий лежит обмен информации. Обмен информацией производится по каналам передачи информации. Каналы передачи информации могут использовать различные физические принципы. Так, при непосредственном общении людей информация передается с помощью звуковых волн, а при разговоре по телефону - с помощью электрических сигналов. Компьютеры могут обмениваться информацией с использованием каналов связи различной физической природы: кабельных, оптоволоконных, радиоканалов и др.

Общая схема передачи информации включает в себя отправителя информации, канал передачи информации и получателя информации.



Если производится двусторонний обмен информацией, то отправитель и получатель информации могут меняться ролями.

Основной характеристикой каналов передачи информации является их пропускная способность (скорость передачи информации). Пропускная способность канала равна количеству информации, которое может передаваться по нему в единицу времени.

Обычно пропускная способность измеряется в битах в секунду (бит/с), и кратных единицах Кбит/с, Мбит/с. Однако, иногда в качестве единицы измерения используется байт в секунду (байт/с) и кратные ему единицы Кбайт/с и Мбайт/с.
Соотношения между единицами пропускной способности канала передачи информации такие же, как между единицами измерения количества информации:

1 байт/с = 8 бит/с;
1 Кбит/с = 1024 бит/с;
1 Мбит/с = 1024 Кбит/с;
1 Гбит/с = 1024 Мбит/с;

Основные составляющие коммуникационных технологий:

1.Локальные компьютерные сети

2.Глобальная компьютерная сеть Интернет

3.Протокол передачи данных TCP/IP

4.Электронная почта

5.Телеконференции

2.Локальные компьютерные сети

2.1.Основыне понятия

Создание компьютерных сетей вызвано практической потребностью совместного использования информации пользователями, работающими на удаленных друг от друга компьютерах. Сети предоставляют пользователям возможность не только быстрого обмена информацией, но и совместного использования принтеров и других периферийных устройств и даже одновременной работы с документами.

Локальная сеть - объединяет несколько компьютеров и дает возможность пользователям совместно использовать ресурсы компьютеров, а также подключенных к сети периферийных устройств (принтеров, плоттеров, дисков, модемов и др.).

В небольших локальных сетях все компьютеры обычно равноправны, т.е. пользователь самостоятельно решает, какие ресурсы своего компьютера (диски, каталоги, файлы) сделать общедоступными по сети. Такие сети называются ***одноранговыми***.

Если к локальной сети подключено более 10 компьютеров, одноранговая сеть может оказаться недостаточно производительной. Для увеличения производительности, а так же в целях обеспечения большей надежности при хранении информации в сети некоторые компьютеры специально выделяются для хранения файлов и программных приложений.

Такие компьютеры называются ***серверами***, а локальная сеть - сетью на основе сервера.

2.2.Аппаратное обеспечение сети.

Каждый компьютер, подключенный к локальной сети, должен иметь специальную плату(сетевой адаптер).

Основной функцией ***сетевого адаптера*** является передача и прием информации из сети. В настоящее время наиболее часто используются сетевые адаптеры типа EtherNet, которые могут объединять в сеть компьютеры различных аппаратных и программных платформ (IBM-совместимых, Macintosh, Unix- компьютеры).

Соединение компьютеров (сетевых адаптеров) между собой производится с помощью кабелей различных типов (коаксиального, витой пары, оптоволоконного). Для подключения к локальной сети портативных компьютеров часто используется беспроводное подключение, при котором передача данных осуществляется с помощью электромагнитных волн. Важнейшей характеристикой локальных сетей, которая определяется типом используемых сетевых адаптеров и кабелей, является скорость передачи информации по сети. Скорость передачи информации по локальной сети обычно находится в диапазоне от 10 до 100 Мбит/с.

2.3.Топология сети.

Общая схема соединения компьютеров в локальной сети называется ***топологией сети.***

Существуют следующие топологии:

-локальная сеть «Линейная шина»;

-локальная сеть типа «Звезда»;

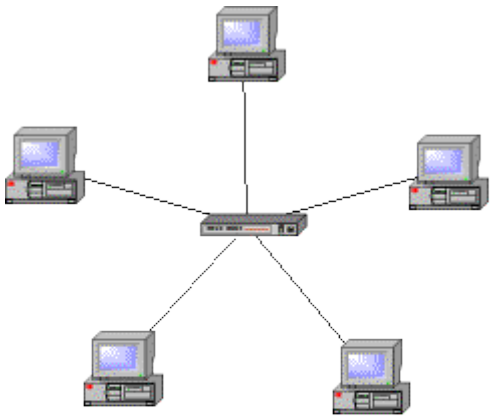
-локальная сеть типа «Кольцо».

Локальная сеть типа "Линейная шина"

Топология при которой кабель проходит от одного компьютера к другому, последовательно соединяя компьютеры и периферийные устройства между собой, называется ***линейной шиной***.

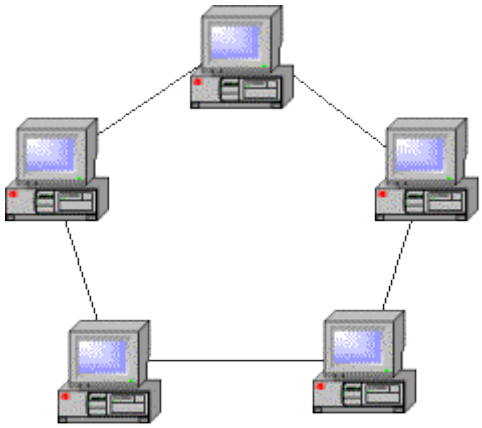
Локальная сеть типа "Звезда"

Топология при которой к каждому компьютеру подходит отдельный кабель из одного центрального узла, называется ***локальной сетью типа "Звезда"***. Обычно при такой схеме соединения центральным узлом является более мощный компьютер.



Преимущество локальной сети типа "звезда" перед локальной сетью типа "линейная шина" состоит в том, что при выходе из строя сетевого кабеля у одного компьютера локальная сеть в целом продолжает нормально функционировать.

Локальная сеть типа «Кольцо»



3.Глобальная компьютерная сеть Интернет.

Локальные сети обычно объединяют несколько десятков компьютеров, размещенных в одном здании, однако они не позволяют обеспечить совместный доступ к информации пользователям, находящимся, например, в различных частях города. В этом случае создаются региональные сети, объединяющие компьютеры в пределах одного региона (города, страны, континента).

Многие организации, заинтересованные в защите информации от несанкционированного доступа (например, военные, банковские и пр.), создают собственные, так называемые корпоративные сети. Корпоративная сеть может объединять тысячи и десятки тысяч компьютеров, размещенных в различных странах и городах.

Потребности формирования единого мирового информационного пространства привели к созданию глобальной компьютерной сети Интернет.

Интернет - это глобальная компьютерная сеть, объединяющая многие локальные, региональные и корпоративные сети и включающие сотни миллионов компьютеров.

В настоящее время на более чем 150 миллионах компьютеров, подключенных к Интернету, хранится громадный объем информации. Глобальная сеть Интернет привлекает пользователей своими информационными ресурсами и сервисами, которыми пользуются около миллиарда человек во всех странах мира.

В каждой локальной или корпоративной сети обычно имеется, по крайней мере, один компьютер, который имеет постоянное подключение к Интернету с помощью линии связи с высокой пропускной способностью (сервер Интернета). В качестве таких "магистральных" линий связи обычно используются оптоволоконные линии с пропускной способностью до 20 Гбит/с или более.

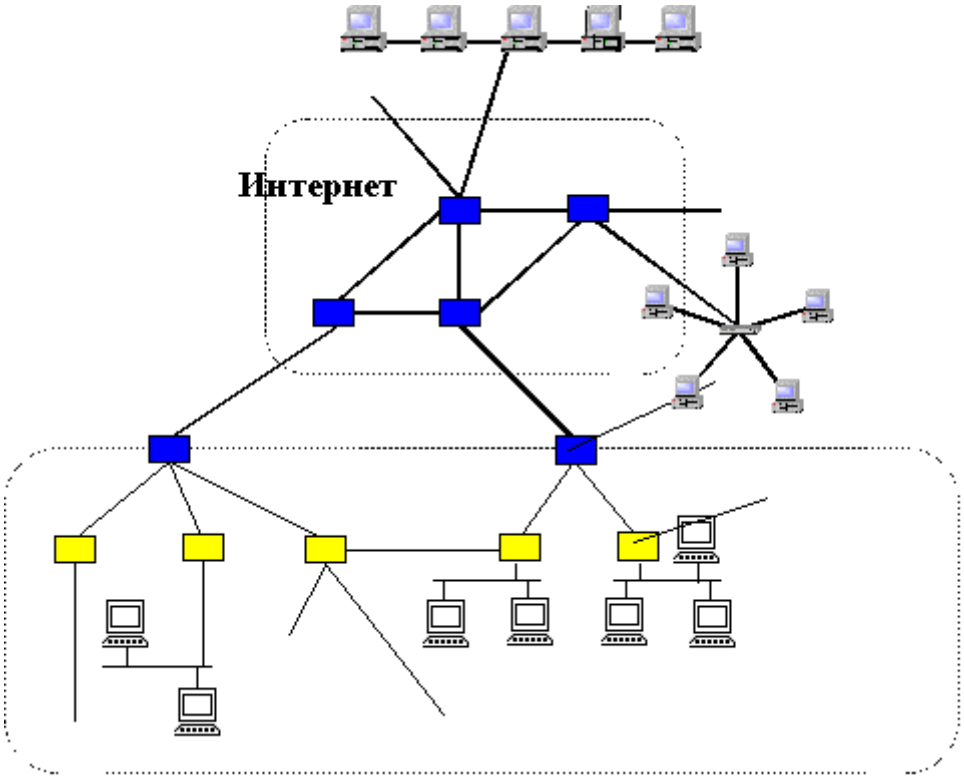
Надежность функционирования глобальной сети обеспечивает большое количество линий связи между региональными сегментами сети. Например, российский региональный сегмент Интернета имеет несколько магистральных линий связи, соединяющих его с североамериканским, европейским и японским сегментами.

3.1.Структура глобальной сети Интернет.

Основу Интернета составляют более 150 миллионов серверов, постоянно подключенных к сети, из которых в России насчитывается около 400 тысяч (на начало 2002 г.).

К серверам Интернета могут подключаться с помощью локальных сетей или коммутируемых телефонных линий сотни миллионов пользователей Интернета.

Структура Интернет



4.Протокол передачи данных TCP/IP

5.4.1. IP-адрес.

Для того чтобы в процессе обмена информацией компьютеры могли найти друг друга, в Интернете существует единая система адресации, основанная на использовании IP-адреса.

Каждый компьютер, подключенный к Интернету, имеет свой уникальный 32-битный (в двоичной системе) ***IP-адрес***.

Система IP-адресации учитывает структуру Интернета, то есть то, что Интернет является сетью сетей, а не объединением отдельных компьютеров. IP-адрес содержит адрес сети и адрес компьютера в данной сети.

Для обеспечения максимальной гибкости в процессе распределения IP-адресов, в зависимости от количества компьютеров в сети, адреса разделяются на 3 класса A, B, C. Первые биты адреса отводятся для идентификации класса, а остальные разделяются на адрес сети и адрес компьютера.

Например, адрес сети класса A имеет только 7 битов для адреса сети и 24 бита для адреса компьютера, т.е. может существовать лишь 128 сетей этого класса, зато в каждой сети может содержаться 2^24 = 16 777 216 компьютеров.

В десятичной записи IP-адрес состоит из 4 чисел, разделенных точками, каждое из которых лежит в диапазоне от 0 до 255. Достаточно просто определить по первому числу IP-адреса компьютера его принадлежность к сети того или иного класса:

-адреса класса A - число от 0 до 127;

-адреса класса B - число от 128 до 191;

-адреса класса C - число от 192 до 223;

Например, IP-адрес сервера компании МТУ-Интел записывается как 195.34.32.11. Он относится к сети класса C, адрес которой 195, а адрес компьютера в сети 34.32.11.

Провайдеры часто предоставляют пользователям доступ в Интернет не с постоянным, а с динамическим IP-адресом, который может меняться при каждом подключении к сети.

4.2. Доменная система имен.

Компьютеры легко могут найти друг друга по числовому IP-адресу, однако человеку запомнить числовой адрес нелегко, и для удобства была введена ***Доменная Система Имен*** (DNS- Domain Name System).

Доменная система имен ставит в соответствие числовому IP-адресу компьютера уникальное доменное имя.

Доменные имена и IP-адреса распределяются международным координационным центром доменных имен и IP-адресов, в который входят по 5 представителей от каждого континента.

Доменная система имен имеет иерархическую структуру: домены верхнего уровня - домены второго уровня и т.д. Домены верхнего уровня бывают 2-х типов: географические (двухбуквенные - каждой стране соответствует двухбуквенный код) и административные (трехбуквенные).

Некоторые имена доменов верхнего уровня

Административные	Тип организации	Географические	Страна
Com	Коммерческая	ca	Канада
Edu	Образовательная	de	Германия
Gov	Правительственная США	jp	Япония
Int	Международная	ru	Россия
Net	Компьютерная сеть	uk	Англия/Ирландия

5. Протокол TCP/IP

Сеть Интернет, являющаяся сетью сетей и объединяющая громадное количество различных локальных, региональных и корпоративных сетей, функционирует и развивается благодаря использованию единого протокола передачи данных TCP/IP. Термин TCP/IP включает название двух протоколов:

- **Transmission Control Protocol** – транспортный протокол;
- **Internet Protocol** – протокол маршрутизации.

5.1. Протокол маршрутизации.

Протокол IP обеспечивает передачу информации между компьютерами сети. Передаваемая по сети информация "упаковывается в конверт", на котором "пишутся" IP-адреса компьютеров получателя и отправителя.

Содержимое пакета на компьютерном языке называется ***IP-пакетом*** и представляет собой набор байтов.

IP-пакеты на пути к компьютеру-получателю проходят промежуточные серверы Интернета, на которых производится операция ***маршрутизации***. В результате маршрутизации IP-пакеты направляются от одного сервера Интернета к другому, постепенно приближаясь к компьютеру получателю.

Internet Protocol (IP) обеспечивает маршрутизацию IP-пакетов, то есть доставку информации от компьютера-отправителя к компьютеру-получателю.

5.2. Транспортный протокол.

Теперь представим, что нам необходимо переслать по почте многостраничную рукопись, а почта бандероли и посылки не принимает. Идея проста: если рукопись не помещается в обычный почтовый конверт, её надо разобрать на листы и переслать их в нескольких конвертах. При этом листы рукописи необходимо обязательно пронумеровать, чтобы получатель знал, в какой последовательности потом эти листы соединить.

В Интернете часто случается аналогичная ситуация, когда компьютеры обмениваются большими по объему файлами. Если послать такой файл целиком, то он может надолго "закупорить" канал связи, сделать его недоступным для пересылки других сообщений.

Для того чтобы этого не происходило, на компьютере-отправителе необходимо разбить большой файл на мелкие части, пронумеровать их и транспортировать в отдельных IP-пакетах до компьютера-получателя. На компьютере-получателе необходимо собрать исходный файл из отдельных частей в правильной последовательности.

Transmission Control Protocol (TCP), то есть транспортный протокол, обеспечивает разбиение файлов на IP-пакеты в процессе передачи и сборку файлов в процессе получения.

Интересно, что для IP-протокола, ответственного за маршрутизацию, эти пакеты совершенно никак не связаны между собой. Поэтому последний IP-пакет вполне может по пути обогнать первый IP-пакет. Может сложиться так, что даже маршруты доставки этих пакетов окажутся совершенно разными. Однако протокол TCP дождется первого IP-пакета и соберет исходный файл в правильной последовательности.

6.Электронная почта

Электронная почта – наиболее распространённый сервис интернета, т.к. является исторически первой информационной услугой компьютерных сетей и не требует обязательного наличия высокоскоростных и качественных линий связи.

Широкую популярность электронная почта завоевала потому, что имеет несколько серьезных преимуществ перед обычной почтой. Наиболее важное из них - это скорость пересылки сообщений. Если письмо по обычной почте может идти до адресата дни и недели, то письмо, посланное по электронной почте, сокращает время передачи до нескольких десятков секунд или, в худшем случае, до нескольких часов.

Другое преимущество состоит в том, что электронное письмо может содержать не только текстовое сообщение, но и вложенные файлы (программы, графику, звук и пр.). Однако не рекомендуется пересылать по почте слишком большие файлы, так как это замедляет работу сети. Для того чтобы этого не происходило, на некоторых почтовых серверах вводятся ограничения на размер пересылаемых сообщений.

Электронная почта позволяет:
-- посылать сообщение сразу нескольким абонентам;
-- пересылать письма на другие адреса;
-- включить автоответчик, на все приходящие письма будет автоматически отсылаться ответ.
-- создавать правила для выполнения определенных действий с однотипными сообщениями (например, удалять рекламные сообщения -- включить автоответчик, на все приходящие письма будет автоматически отсылаться ответ. приходящие от определенных адресов) и т.д.

6.1. Адрес электронной почты.

Для того чтобы электронное письмо дошло до адресата, оно, кроме самого сообщения, обязательно должно содержать адрес электронной почты получателя письма.

Первая часть почтового адреса имеет произвольный характер и задается самим пользователем при регистрации почтового ящика. Вторая часть является доменным именем почтового сервера, на котором пользователь зарегистрировал свой почтовый ящик.

Адрес электронной почты записывается по определенной форме и состоит из 2-х частей, разделенных символом @:
user_name@server_name

Адрес электронной почты записывается только латинскими буквами и не должен содержать пробелов. Например, почтовый сервер компании МТУ-Интел имеет имя mtu-net.ru. Соответственно имена почтовых ящиков пользователей будут иметь вид:

user_name@mtu-net.ru

6.2.Функционирование электронной почты

Любой пользователь Интернета может зарегистрировать почтовый ящик на одном из серверов Интернета (обычно на почтовом сервере провайдера), в котором будут накапливаться передаваемые и получаемые электронные письма.

Для работы с электронной почтой необходимы специальные почтовые программы, причем для любой компьютерной платформы существует большое количество почтовых программ. Почтовые программы входят в состав широко распространенных коммуникационных пакетов: Outlook Express входит в Microsoft Internet Explorer, Netscape Messenger - в Netscape Communicator.

С помощью почтовой программы создается почтовое сообщение на локальном компьютере. На этом этапе кроме написания текста сообщения необходимо указать адрес получателя сообщения, тему сообщения и вложить в сообщение при необходимости файлы.

Процесс передачи сообщения начинается с подключения к Интернету и доставки сообщения в свой почтовый ящик на удаленном почтовом сервере. Почтовый сервер сразу же отправит это сообщение через систему почтовых серверов Интернета на почтовый сервер получателя в его почтовый ящик.

Адресат для получения письма должен соединиться с Интернетом и доставить почту из своего почтового ящика на удаленном почтовом сервере на свой локальный компьютер.

Почтовые программы обычно предоставляют пользователю также многочисленные дополнительные сервисы по работе с почтой (выбор адресов из адресной книги, автоматическую рассылку сообщений по указанным адресам и др.).

7.Телеконференции

В Интернете существуют десятки тысяч конференций или групп новостей, каждая из которых посвящена обсуждению какой-либо проблемы. Каждой конференции выделяется свой почтовый ящик на серверах Интернета, которые поддерживают работу этой телеконференции.

Пользователи могут посылать свои сообщения на любой из этих серверов. Сервера периодически синхронизируются, то есть обмениваются содержимым почтовых ящиков телеконференций, поэтому материалы конференций в полном объеме доступны пользователю на любом таком сервере.

Принцип работы в телеконференциях мало чем отличается от принципа работы с электронной почтой. Пользователь может посылать свои сообщения в любую телеконференцию и читать сообщения, посланные другими участниками.

Для работы в телеконференциях используют обычно те же самые почтовые программы, что и при работе с электронной почтой, например Outlook Express.

МЕХАНИЧЕСКИЙ ЭТАП РАЗВИТИЯ ЭВМ

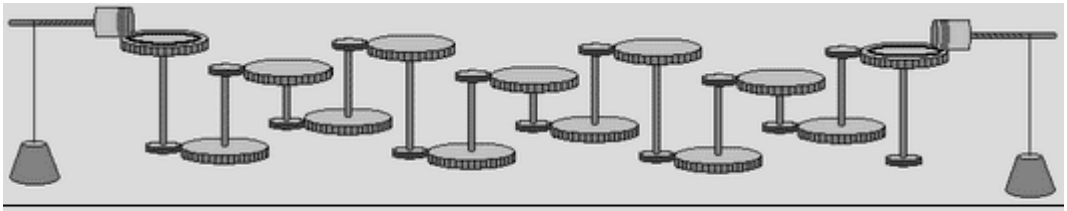
Счетная машина Леонардо да Винчи


Леонардо да Винчи

Историю механического этапа развития вычислительной техники можно начать вести с 1492 года, когда Леонардо да Винчи (1452-1519) разработал чертеж счетной машины и описал его в своих дневниках, ныне известных, как двухтомник «Мадридский Кодекс». Долгое время эти дневники пролежали в безызвестности в национальной Библиотеке Испании, пока 13-го февраля 1967 года не были найдены американскими исследователями.

Среди чертежей первого тома «Мадридского кодекса», почти полностью посвященного прикладной механике, ученые обнаружили эскиз 13-разрядного суммирующего устройства с десятизубцовыми кольцами.

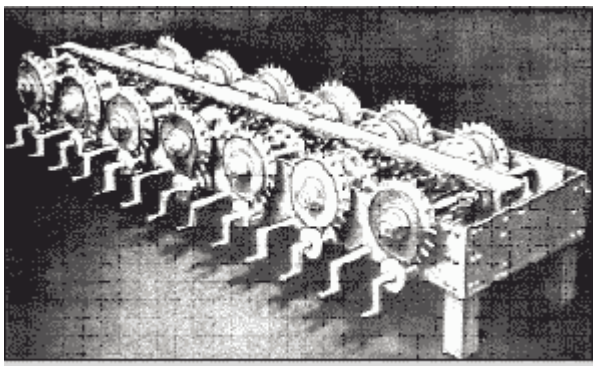
Основу счетной машины составляли стержни с двумя зубчатыми колесами, большое - с одной стороны и маленькое - с другой. Как видно из эскиза Леонардо да Винчи, эти стержни располагались так, чтобы маленькое колесо на одном стержне входило в сцепление с большим колесом на соседнем стержне. Таким образом десять оборотов первого стержня приводили к одному полному обороту второго стержня, а десять оборотов второго - к одному полному обороту третьего стержня и так далее. Вся система состояла из тринадцати стержней и приводилась в движение набором грузов.


Эскиз счетной машины Леонардо да Винчи

Вероятно, при жизни Леонардо да Винчи счетная машина не была создана. Однако, в 1967 году доктор Роберто Гуателли, известный эксперт по Леонардо да Винчи, работающий по приглашению фирмы IBM с 1951 года над воссозданием машин великого мастера, исследуя эскизы счетной машины в «Мадридском кодексе», вспомнил, что видел подобный рисунок в "Атлантическом Кодексе".

Изучив оба рисунка, доктор Гуателли создал в 1968 году копию счетной машины. Модель поддерживала постоянное отношение десяти к одному в каждом из его 13 цифровых колес. После полного оборота первой ручки, колесо единиц немного поворачивалось, чтобы отметить новую цифру в пределах от ноля до девяти.

В соответствии с пропорцией десять к одному, десятый оборот первой ручки заставляет колесо единиц совершить полный оборот и стать на ноль, который в свою очередь сдвигает колесо десятков с ноля на единицу. Каждое последующее колесо, отмечающее сотни, тысячи и т.д., действует подобным же образом.


Модель счетной машины

По сравнению с оригинальным эскизом Леонардо были внесены небольшие улучшения, чтобы дать зрителю более ясную картину того, как каждое из этих 13 колес может двигаться независимо и все же поддерживать пропорцию десять к одному.

Однако, в течение года относительно точности воспроизведения счетной машины появлялись возражения, и для установки подлинности механизма в университете Штата Массачусетс были проведены Академические испытания.

Оппоненты считали, что на эскизах Леонарда да Винчи изображен механизм пропорционирования, а не счетная машина, и один оборот первой оси вызывает 10 оборотов второй, 100 оборотов третьей и 10 в 13 - ой степени оборотов последней оси. Работа такого механизма, по мнению противников доктора Гуателли, не могла осуществляться из-за огромной силы трения, которую необходимо преодолевать для оборота всех стержней.

Голоса сторонников и оппонентов разделились поровну, но, тем не менее, IBM решила удалить спорную модель из коллекции.

Вычисляющие часы.

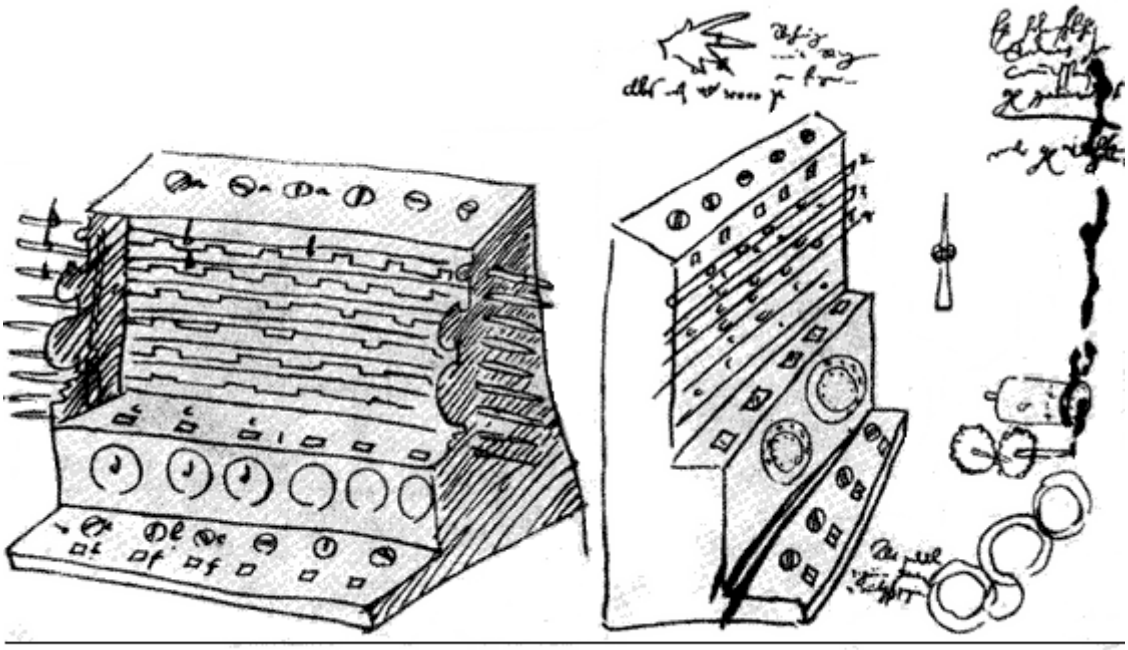

Вильгельм Шикард

Спустя почти 150 лет со дня изобретения счетной машины Леонардо да Винчи, в 1623 году в письме Иоганну Кеплеру немецкий профессор математики и астрономии Вильгельм Шикард (1592-1635) написал о машине, которая способна вычитать и складывать, а с помощью особых приспособлений на корпусе - еще и умножать, и приложил эскиз устройства. Это был шести разрядный механический калькулятор, получивший название «Вычисляющие часы». Устройство было названо часами, потому что его принцип работы основывался на использовании звёздочек и шестерёнок, как и в настоящих часах, а когда результат превышал резервы памяти, раздавался звон колокольчика.

Вычисляющие часы – первое механическая счетная машина, позволяющая складывать, вычитать, делить и умножать числа. Однако, она была известна довольно узкому кругу лиц, и поэтому долгое время (почти 300 лет со дня ее изобретения) первой счетной машиной считалось изобретение Блеза Паскаля (Пасклин).

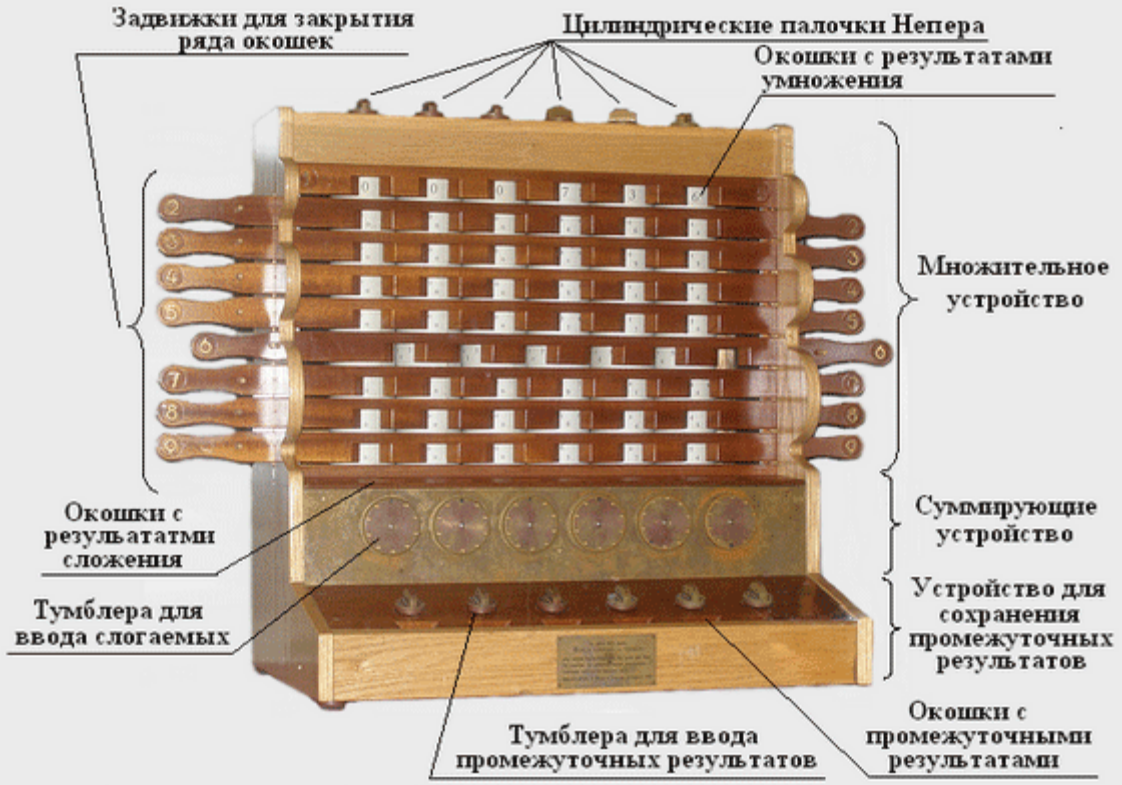
История «вычисляющих часов» трагична. Два изготовленных экземпляра машины, один из которых предназначался Кеплеру, сгорели во время пожара. О самом проекте забыли на долгие годы, и чертежи устройства были утеряны из-за бушующей в тот период Тридцатилетней войны (1618-1648 гг), и только в 1935 году они были найдены. Найдены только для того, чтобы быть потерянными снова по причине второй мировой войны (1941-1945 гг).

И только спустя 21 год, в 1956 году в городской библиотеке Штутгарта была найдена фотокопия эскиза «вычисляющих часов», и в 1960 группа энтузиастов, на основе этой фотокопии и писем Шиккарда, сумели построить действующую модель «вычисляющих часов».



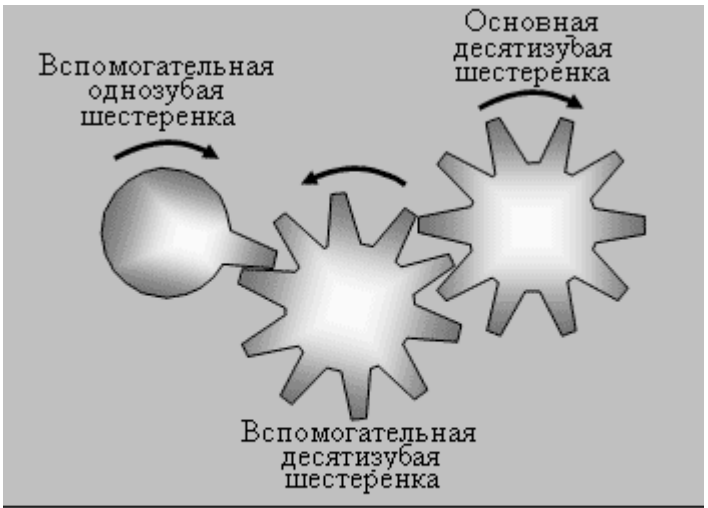
Эскиз Вычисляющих часов, сделанный Вильгелем Шикардом

Вычисляющие часы состояли из трех частей: верхней – множительной части (для умножения и деления), средней – суммирующей части (для вычитания и сложения) и нижней части - механизма для сохранения промежуточных результатов счета.



Модель Вычисляющих часов

Суммирующее устройство состояло из шести осей, на каждую из которых крепился барабан, на боковую сторону которого наносились числа от нуля до девяти, шестеренка с десятью зубьями и вспомогательная однозубая шестеренка. Однозубая шестеренка использовалась для передачи переполнения в старший разряд (для поворота десятизубой шестеренки старшего разряда на 1/10 оборота после того, как предыдущая десятизубая шестеренка совершит полный оборот).



Пример использования вспомогательной шестеренки при переносе разряда переполнения

Для того, чтобы при переносе разряда переполнения все десятизубые шестеренки вращались в одном направлении, использовалась вспомогательная десятизубая шестеренка.

Всего в суммирующем устройстве использовалось 11 десятизубых шестеренок и шесть однозубых.

Для ввода слагаемых использовалось шесть тумблеров, результаты ввода отображались в окошках над этими тумблерами.

Вычисление суммы состояло в последовательном вводе слагаемых с помощью тумблеров и считывания результата. Для вычитания сначала вводили уменьшаемое, а затем обратным вращением тумблеров вводилось вычитаемое. Деление заменялось повторным вычитанием делителя из делимого.

Множительное устройство работало по принципу«палочек Непера». В верхней части «Вычисляющих часов» располагались шесть параллельных цилиндров с нанесенными на них таблицами умножения (цилиндрические палочки Непера). На рисунке справа изображена таблица, наносимая на цилиндры, в развернутом виде.

1	2	3	4	5	6	7	8	9	0
2	4	6	8	1/0	1/2	1/4	1/6	1/8	0
3	6	9	1/2	1/5	1/8	2/1	2/4	2/7	0
4	8	1/2	1/6	2/0	2/4	2/8	3/2	3/6	0
5	1/0	1/5	2/0	2/5	3/0	3/5	4/0	4/5	0
6	1/2	1/8	2/4	3/0	3/6	4/2	4/8	5/4	0
7	1/4	2/1	2/8	3/5	4/2	4/9	5/6	6/3	0
8	1/6	2/4	3/2	4/0	4/8	5/6	6/4	7/2	0
9	1/8	2/7	3/6	4/5	5/4	6/3	7/2	8/1	0

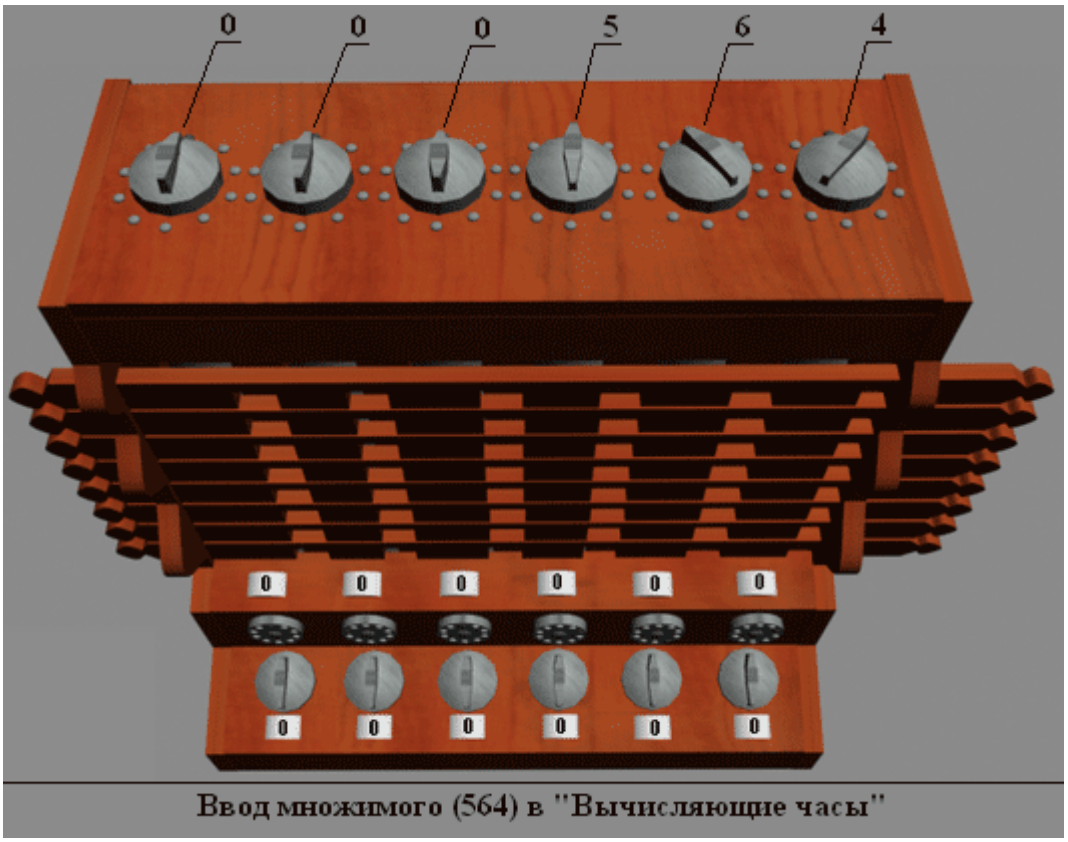
Таблица, нанесенная на цилиндры, в развернутом виде.

Поворотом цилиндров с помощью специальных ручек, располагающихся сверху «Вычисляющих часов», устанавливалось множимое. В результате в окошках устройства отображалась таблица для умножения по технологии «палочек Непера». Для удобства использования полученной таблицы применялась панель с девятью рейками с окошками (на каждой рейке было 6 окошек). На рейках были нанесены числа от 1 до 9. Рейка с цифрой 1 располагалась сверху на панели, за ней шла рейка с цифрой 2 и так далее до 9.

Сдвигая рейки, соответствующие разрядам множителя, открывались строки таблицы, используемые при умножении. Остальные строки (ненужные для умножения) оставались скрытыми и не отвлекали от вычислений.

Рассмотрим пример работы устройства при умножении 564 на 37 (564x37=20868):

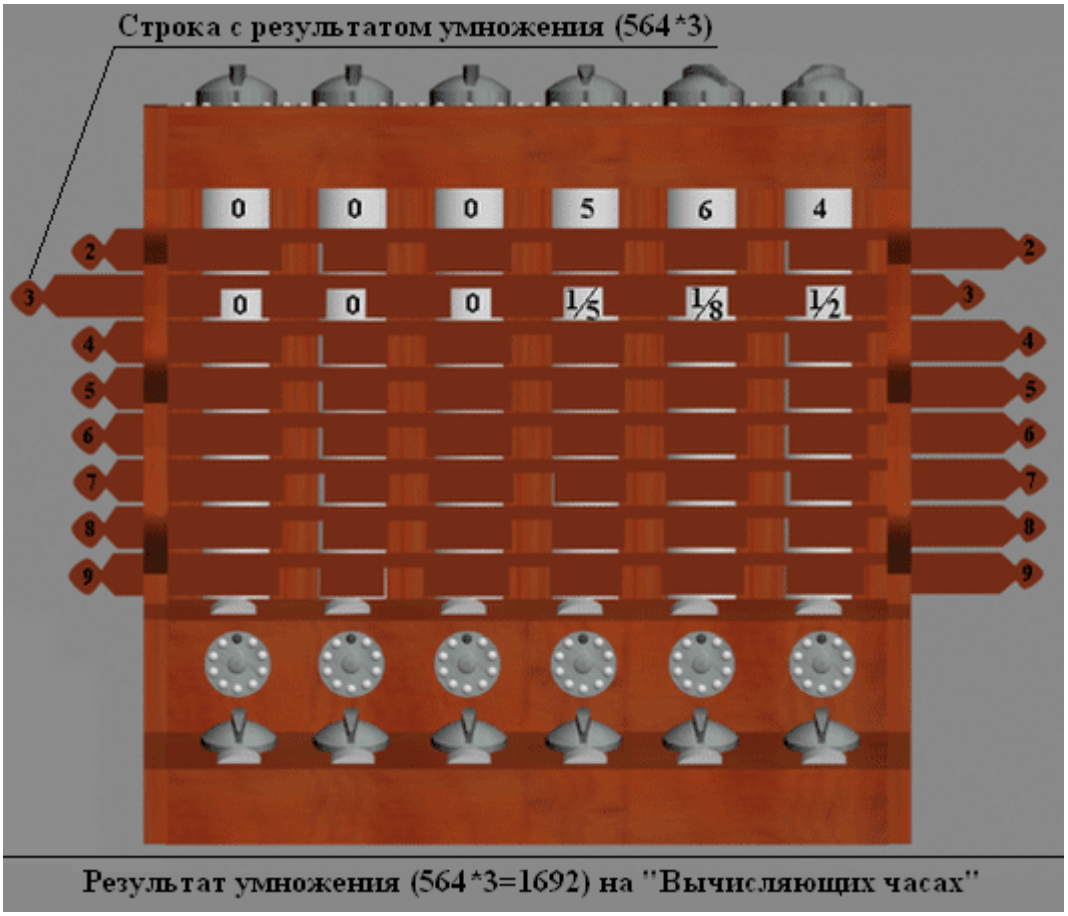
1. Вводим множимое (564) с помощью тумблеров, располагаемых в верхней части «Вычисляющих часов».



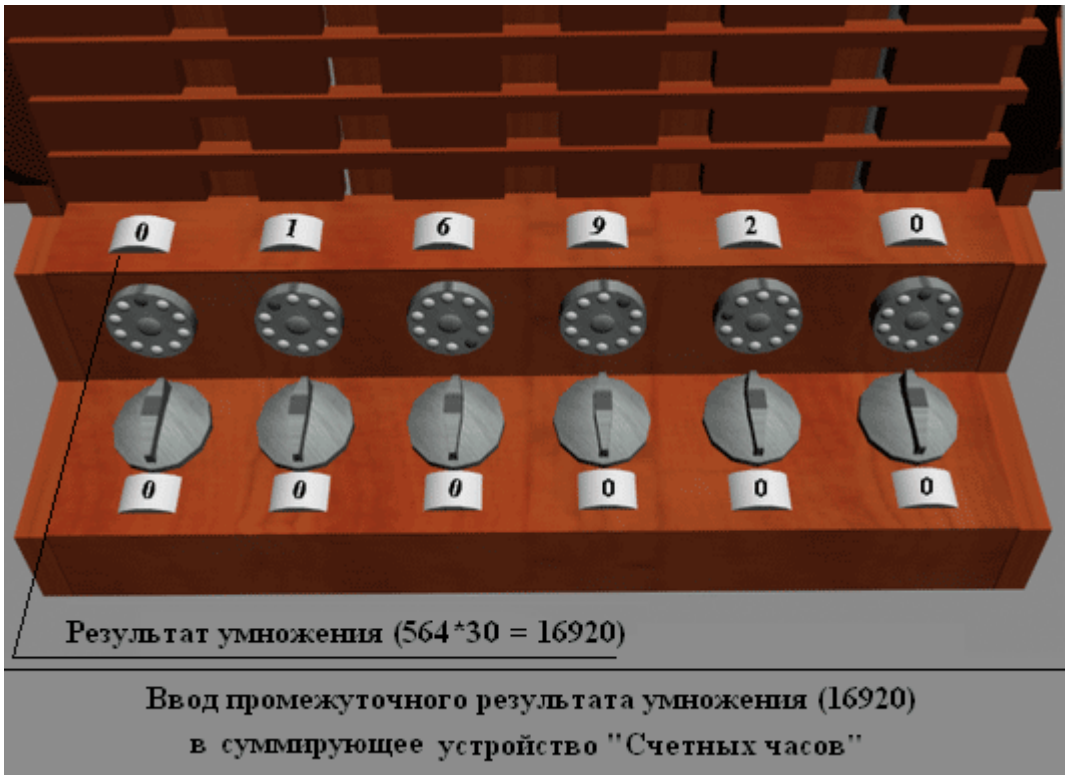
Ввод множимого (564) в "Вычисляющие часы"

2. Выдвигаем планку с цифрой 3, открывая строку с результатом умножения 564 на 3.

3. Считываем **по** наклонной плоскости результат умножения (1692) и приписываем к нему справа нолик, так как умножение осуществлялось на разряд десятков.

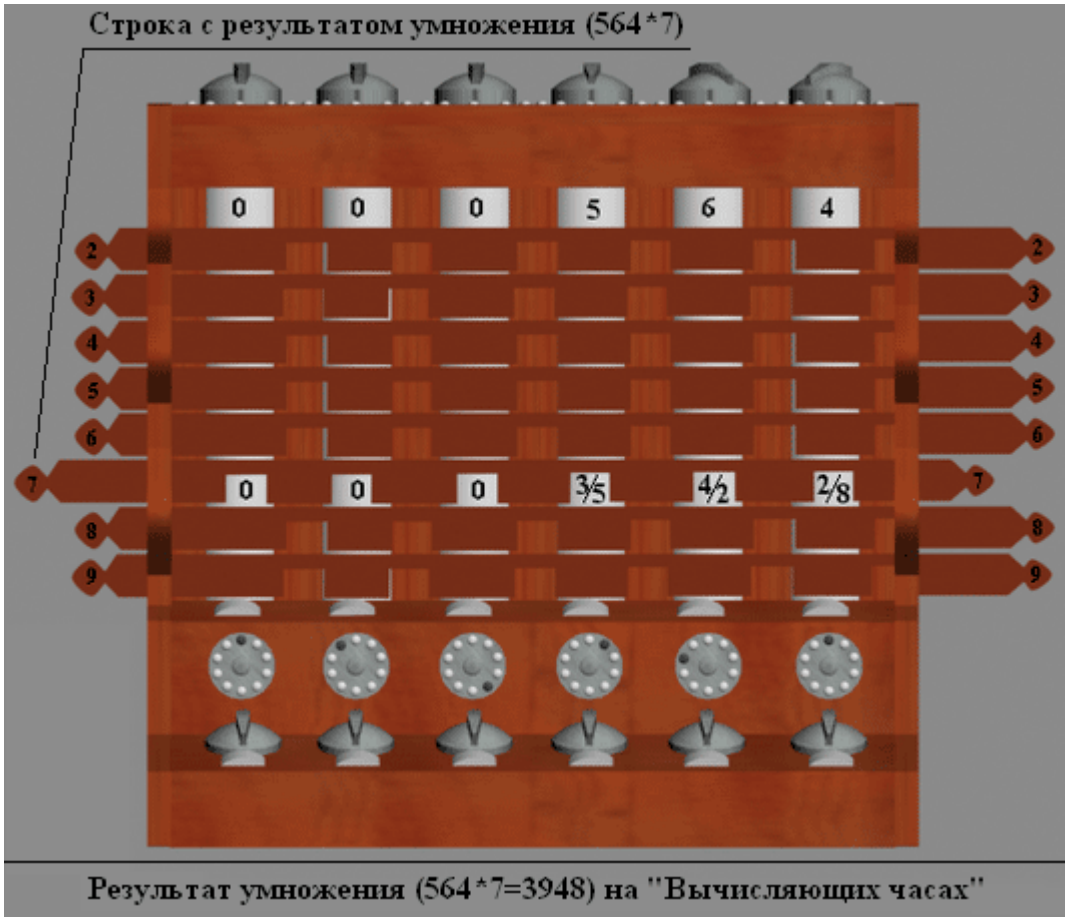


4. Водим полученный результат (16920) в суммирующее устройство вычисляющих часов.



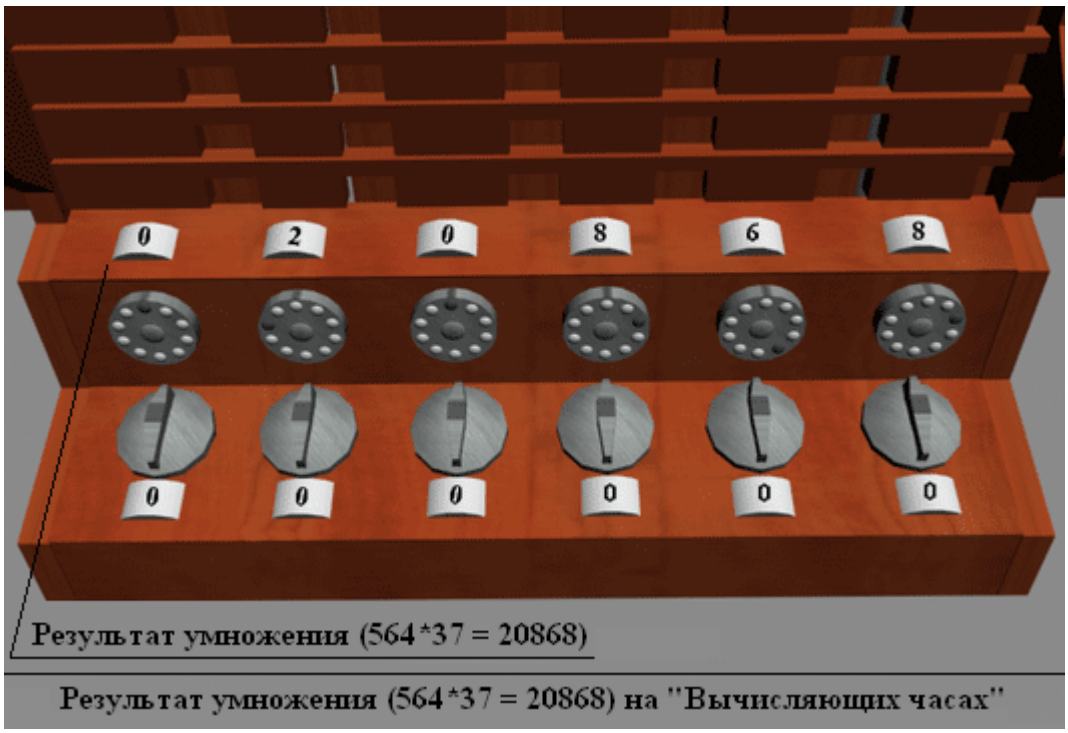
5. Задвигаем планку с цифрой 3 и выдвигаем планку с цифрой 7, открывая строку с результатом умножения 564 на 7.

6. Считываем по наклонной плоскости результат умножения (3948).



7. Вводим полученный результат (3948) в суммирующее устройство вычисляющих часов.

8. Считываем из окошек результата сложения искомое произведение: 20868



Результат умножения (564*37 = 20868) на "Вычисляющих часах"

Третья часть вычисляющих часов Шиккарда предназначалась для хранения промежуточных результатов вычислений, и состояла из шести осей с нанесенными на них цифрами от 0 до 9, шести тумблеров для ввода чисел и шести окошек, в которых отображалось сохраненное число.

В доме-музее Иоганна Кеплера, на его родине в городе Вайле, хранится действующая модель «Вычисляющих часов», изготовленная по чертежам, найденным в письмах Шикарда.

Паскалин.


Блез Паскаль

Первое вычислительное устройство, получившее известность еще при жизни автора, было «Паскалин» или, как его иногда называют, «Паскалево колесо». Оно было создано в 1644 году Блезом Паскалем (19.06.1623-19.08.1662) и на столетия заняло место первой счетной машины, так как в то время о «Вычисляющих часах» Шиккарда было известно крайне узкому кругу людей.

Создание «Паскалины» было вызвано желанием Паскаля помочь своему отцу. Дело в том, что отец великого ученого Этьен Паскаль в 1638 году возглавлял группу рантьеров, протестовавших против решения правительства отменить выплату ренты, за что и впал в немилость кардиналу Ришелье, приказавшему арестовать бунтовщика. Отцу Паскаля пришлось бежать.

Четвертого апреля 1939 года, благодаря Жаклин, младшей дочери отца ученого, и герцогине д'Эгийон, удалось выпросить прощение кардинала. Этьен Паскаль был

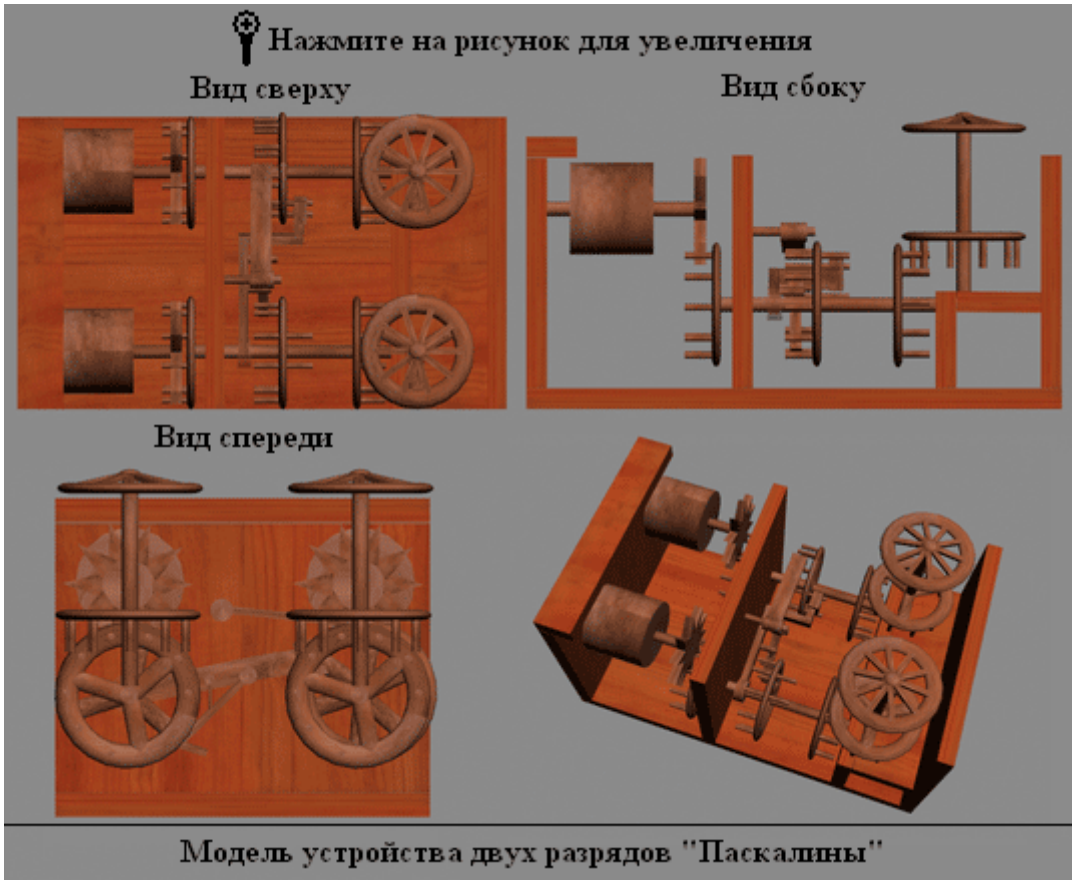
назначен на пост интенданта Руанского генеральства, и 2 января 1640 года семейство Паскалей прибыло в Руан. Отец Паскаля сразу же погрузился в работу, день и ночь просиживая над подсчетами налоговых сборов. В 1642 году, в возрасте 19 лет, Блез Паскаль, желая облегчить работу своего отца, начал работу над суммирующей машиной.

Первая созданная модель его не удовлетворила, и он немедля преступил к ее улучшению. Всего было создано около 50 различных моделей вычислительных устройств. Паскаль так писал о своем труде: «Я не экономил ни времени, ни труда, ни средств, чтобы довести ее до состояния быть тебе полезной... Я имел терпение сделать до 50 различных моделей: одни деревянные, другие из слоновой кости, из эбенового дерева, из меди…». Окончательный вариант устройства был создан в 1645 году.

Впервые описание «Паскалины» появилось в «Энциклопедии» Дидро в 18 веке.

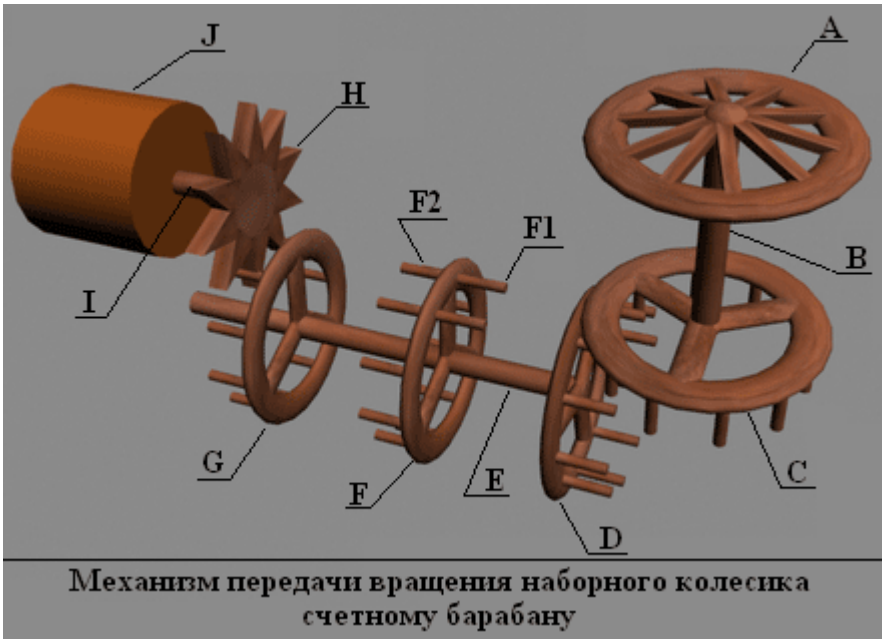
Модель "Паскалины"

Она представляла собой небольшой латунный ящик размером 36х13х8 см, содержащий внутри множество связанных между собой шестеренок и имеющий несколько наборных колесиков с делениями от 0 до 9, при помощи которых осуществлялось управление – ввод чисел для операций над ними и отображение результатов операций в окошках.



Каждое наборное колесико соответствовало одному разряду числа. Первые варианты устройства были пятиразрядными, впоследствии Паскаль создал шести- и даже восьмиразрядные варианты.

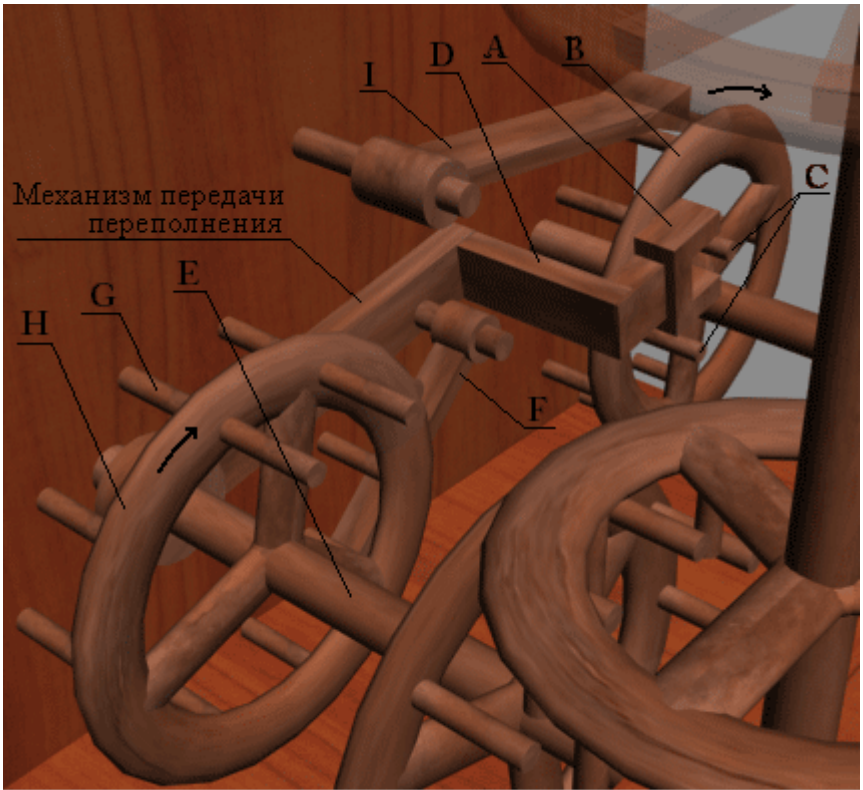
Два младших разряда восьмиразрядной «Паскалины» были приспособлены для оперирования с денье и су, т.е. первый разряд был двадцатеричным, а второй двенадцатеричным, потому что в те времена французская монетная система была сложнее современной. В ливре было 12 денье, а в денье – 20 су. При выполнении обычных десятичных операций можно было отключать разряды, предназначенные для разменной монеты. Шести- и пятиразрядные версии машин могли работать только с десятичными цифрами.



Наборные колесики поворачивались вручную с помощью ведущего штифта, который

вставлялся между зубчиками, количество которых для десятичных разрядов было десять, для двенадцатеричных – двенадцать, а для двадцатеричных – двадцать. Для удобства ввода данных использовали неподвижный упор, закрепленный снизу наборного колесика, чуть левее цифры 0.

Поворот наборного колесика передавался счетному барабану с помощью специального приспособления, изображенного на рисунке слева. Наборное колесико (A) жестко соединялось с корончатым колесом (C) с помощью стержня (B). Корончатое колесо (C) входило в зацепление с корончатым колесом (D), располагающимся под прямым углом относительно корончатого колеса (C). Так передавалось вращение наборного колесика (A) корончатому колесу (D), которое жестко соединялось со стержнем (E), на котором закреплялось корончатое колесо (F), используемое для передачи переполнения в старший разряд с помощью зубцов (F1) и для приема переполнения от младшего разряда с помощью зубцов (F2). Также на стержне (E) закреплялось корончатое колесо (G), используемое для передачи вращения наборного колесика (A) счетному барабану (J) с помощью зубчатого колеса (H).

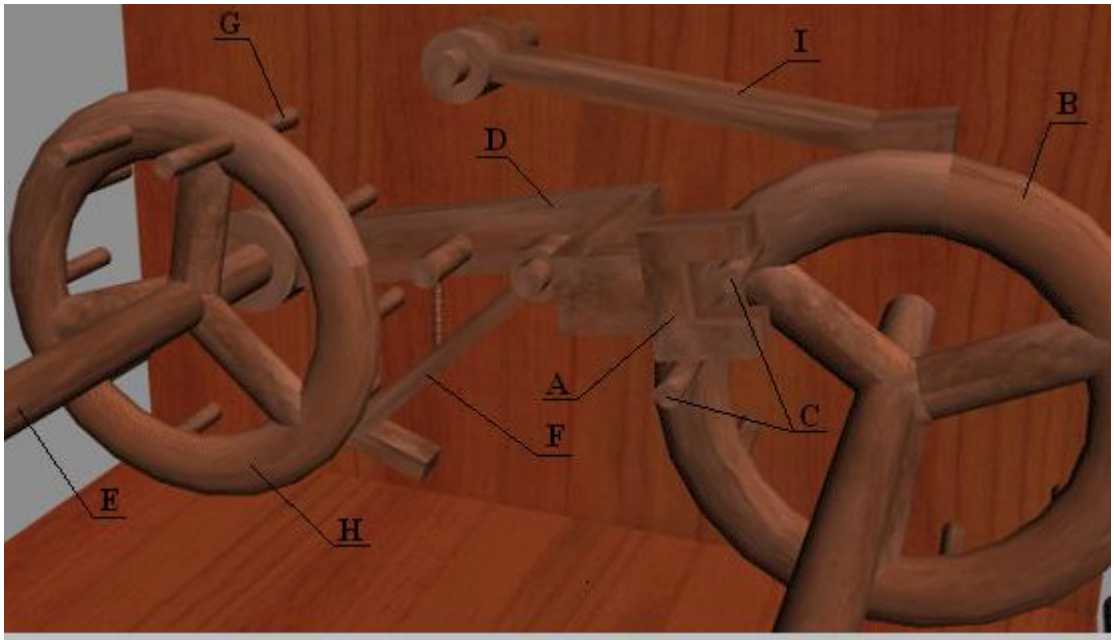


Механизм переноса переполнения в "Паскалине"

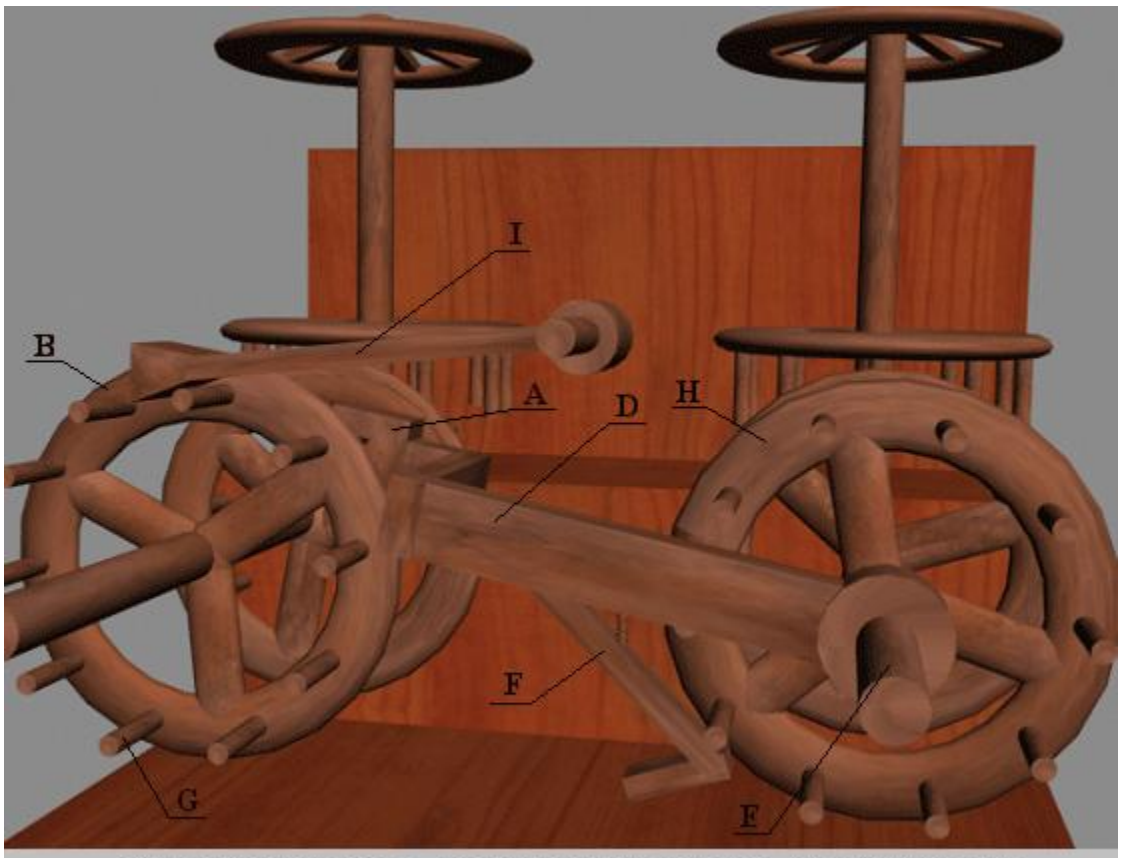
При полном повороте наборного колесика в старший разряд «Паскалины» передавался результат переполнения с помощью механизма, изображенного на рисунках «Механизм переноса переполнения в «Паскалине».

Для передачи переполнения использовались два корончатых колеса (B и H) соседних разрядов. На корончатом колесе (B) младшего разряда имелись два стержня (C), которые могли входить в зацепления с вилкой (A), закрепленной на двухколенчатом рычагом D. Этот рычаг свободно вращался вокруг оси (E) старшего разряда. Также на этом рычаге закреплялась подпружиненная собачка (F).

Когда наборное колесико младшего разряда достигало цифры 6, стержни (C) входили в зацепление с вилкой (A). В момент, когда наборное колесико переходило от цифры 9 к цифре 0, вилка выходила из зацепления со стержнями (C) и под действием собственного веса падала вниз, при этом собачка входила в зацепление со стержнями (G) корончатого колеса (E) старшего разряда и передвигала его на один шаг вперед.

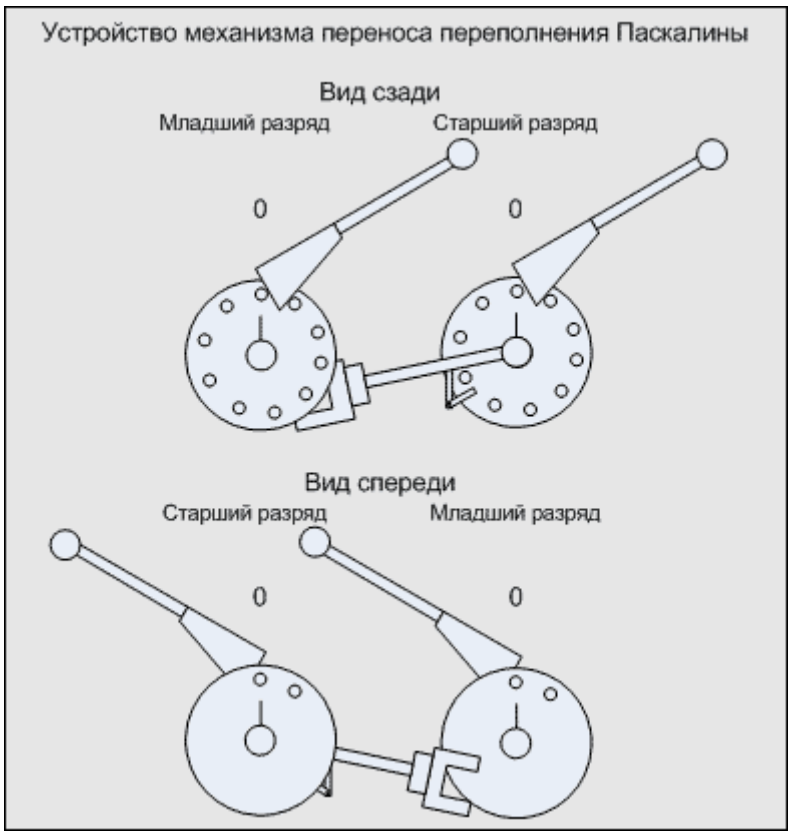


Механизм переноса переполнения в "Паскалине" (Вид спереди)



Механизм переноса переполнения в "Паскалине" (Вид сзади)

Принцип работы механизма переноса переполнения в «Паскалине» иллюстрируется на анимации снизу.



Основным назначением устройства было сложение. Для сложения нужно было проделать ряд несложных операций:

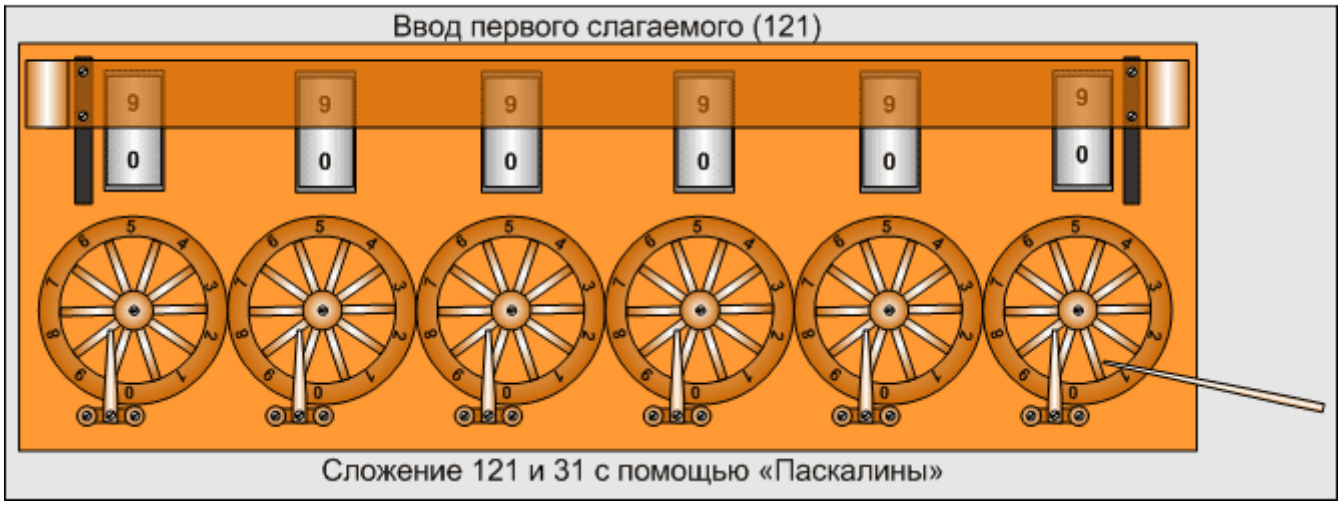
1. Сбросить предыдущий результат, вращая наборные колесики, начиная с младшего разряда до тех пор, пока в каждом из окошек не появятся нули.

2. С помощью этих же колесиков вводится первое слагаемое, начиная с младшего разряда.

3. Затем колесики вращаются дальше, чтобы выставить значение следующего слагаемого, после чего в окошках устройства отображается полученный результат.

На анимации внизу иллюстрируется работа «Паскалины» на примере сложения 121 и 32.

Нажмите на рисунок для просмотра анимации (800 Кб)



Вычитание производилось немного сложнее, так как перенос разрядов переполнения происходил только при вращении наборных колесиков по часовой стрелке. Для предотвращения вращения наборных колесиков против часовой стрелки использовался стопорный рычаг (I).

Подобное устройство переноса разряда переполнения привело к проблеме в реализации вычитания на Паскалине, путем вращения наборных колесиков в обратном направлении, как это было сделано в «Счетных часах» Шикарда. Поэтому Паскаль заменил операцию вычитания на сложение с дополнением до девяти.

Поясню способ, используемый Паскалем, на примере. Допустим, необходимо решить уравнение Y=64-37=27. С помощью метода дополнения представим число 64 как разность чисел 99 и 35 (64=99-35), таким образом наше уравнение сводится к следующему виду: У=64-37=99-35-37=99-(35+37)=27. Как видно из преобразования, вычитание частично заменилось на сложение и вычитание результата сложения из 99, что есть преобразование обратное дополнению. Следовательно, Паскалю оставалось решить задачу автоматического дополнения до девяти, для чего он на счетном барабане ввел два ряда цифр так, чтобы сумма двух цифр, располагающихся друг под другом, всегда равнялась 9. Таким образом, число, отображаемое в верхнем ряду окошка результата вычислений, представляло собой дополнение числа нижнего ряда до 9.

0	1	2	3	4	5	6	7	8	9
9	8	7	6	5	4	3	2	1	0

Цифры, нанесенные на счетный барабан Паскалины

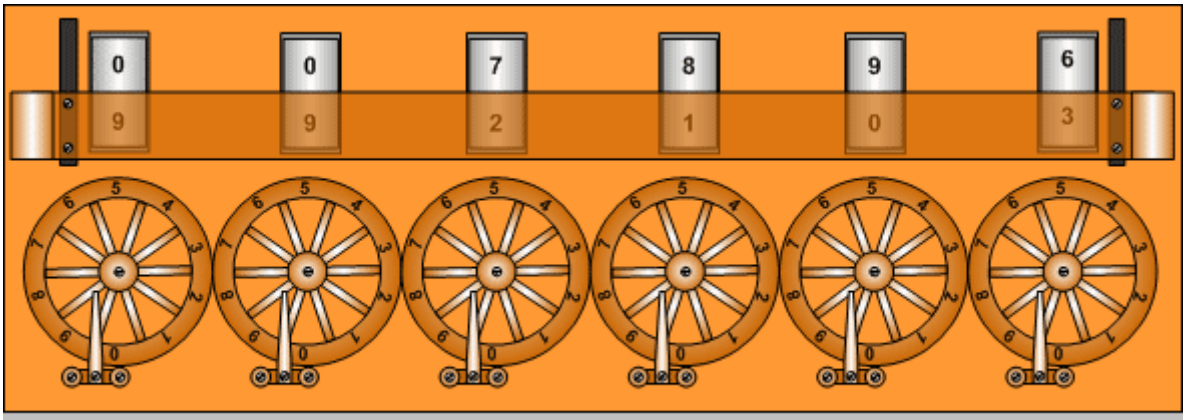
В развернутом виде ряды, нанесенные на цилиндр, изображены на рисунке слева.

Нижний ряд использовался при сложении, а верхний ряд при вычитании. Для того, чтобы неиспользуемый ряд не отвлекал от вычислений его прикрывали планкой.

Рассмотрим работу Паскалины на примере вычитания 132 из 7896 (7896-132=7764):

1. Закрываем нижний ряд окошек, используемый для сложения.

2. Поворачиваем наборные колесики так, чтобы в верхнем ряду отобразилось число 7896, при этом в нижнем закрытом ряду будет отображено число 992103.



Ввод уменьшаемого при вычитании с помощью "Паскалины"

3. Вводим вычитаемое так же, как вводим слагаемые при сложении. Для числа 132 это делается так:

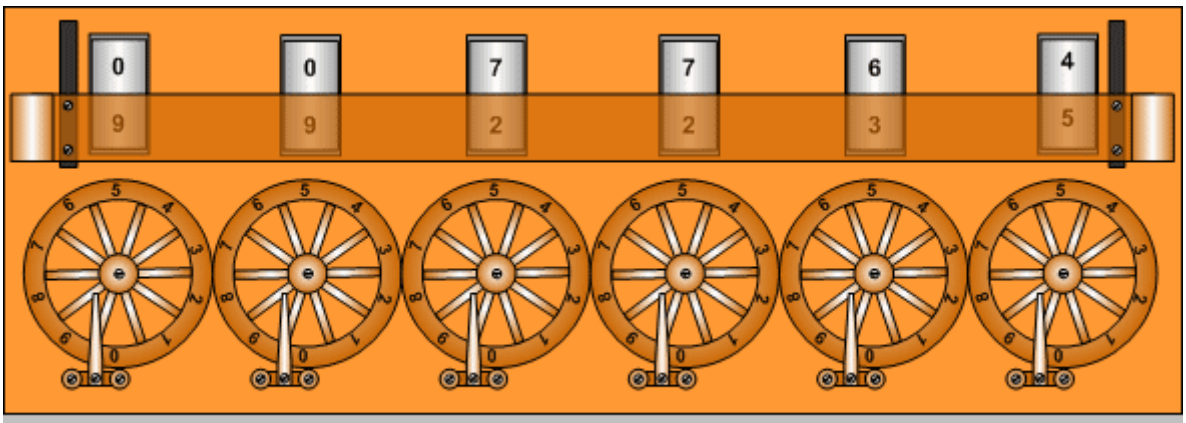
- устанавливается штифт напротив цифры 2 младшего разряда «Паскалины», и по часовой стрелки поворачивается наборное колесико, пока штифт не упрется в упор.

- устанавливается штифт напротив цифры 3 второго разряда «Паскалины», и по часовой стрелки поворачивается наборное колесико, пока штифт не упрется в упор.

- устанавливается штифт напротив цифры 1 третьего разряда «Паскалины», и по часовой стрелки поворачивается наборное колесико, пока штифт не упрется в упор.

- остальные разряды не изменяются.

4. В верхнем ряду окошек будет отображен результат вычитания 7896-132=7764.



Результат вычитания (7896-132=7764) с помощью "Паскалины"

Умножение в устройстве выполнилось в виде многократного сложения, для деления числа можно было использовать многократное вычитание.

При разработке счетной машины Паскаль столкнулся со множеством проблем, наиболее острым из которых было изготовление узлов и шестеренок. Рабочие плохо понимали идеи ученого, и технология приборостроения была низка. Иногда Паскалю самому приходилось брать в руки инструменты и доводить до ума те или иные детали машины, или упрощать их конфигурацию, чтобы мастера могли их изготовить.

Одну из первых удачных моделей «Паскалины» изобретатель подарил канцлеру Сегье, что помогло ему 22 мая 1649 года получить королевскую привилегию, подтверждавшую авторство изобретения и закрепляющую за Паскалем право на производство и продажу машины. За 10 лет было создано примерно 50 моделей вычислительной машины и продано около дюжины. До нашего времени дошли 8 образцов.

Хотя машина и была революционна для своего времени и вызывала всеобщий восторг, она не принесла богатство создателю, так как практического применения не получила, хотя о них много говорилось и писалось. Возможно, потому что клерки, в помощь которым предназначалась машина, боялись потерять из-за нее работу, а работодатели скупились покупать дорогое устройство, предпочитая дешевую рабочую силу.

Тем не менее, идеи, заложенные в основу построения «Паскалины», стали основой для развития вычислительной техники. У Паскаля были и непосредственные преемники. Так Родригес Перейра, известный своей системой обучения глухонемых, сконструировал две счетные машины, основанные на принципах работы «Паскалины», но в результате ряда доработок, оказавшимися более совершенными.

Калькулятор Лейбница.

Готфрид Вильгельм Лейбниц

Первая счетная машина, позволявшая производить умножение и деление также легко, как сложение и вычитание, была изобретена в Германии в 1673 году Готфридом Вильгельмом Лейбницем (1646-1716), и называлась «Калькулятор Лейбница».

Идея создать такую машину у Вильгельма Лейбница появилась после знакомства с голландским астрономом и математиком Христианом Гюйгенсом. Видя нескончаемые вычисления, которые астроному приходилось производить, обрабатывая свои наблюдения, Лейбниц решил создать устройство, которое ускорило и облегчило бы эту работу.

Первое описание своей машины Лейбниц сделал в 1670 году. Через два года ученый составил новое эскизное описание, на основе которого в 1673 году построил действующее арифметическое устройство и продемонстрировал его в феврале 1673 года на заседании Лондонского Королевского общества. В заключение своего выступления он признал, что устройство не совершенно, и пообещал его улучшить.

В 1674 – 1676 годах Лейбниц провел большую работу по улучшению изобретения и привез в Лондон новый вариант калькулятора. Это была малоразрядная модель счетной машины, не пригодная для практического применения. И только в 1694 году Лейбниц сконструировал 12 разрядную модель. Впоследствии калькулятор несколько раз дорабатывался. Последний вариант был создан в 1710 году. По образцу двенадцатиразрядной счетной машины Лейбница в 1708 году профессор Вагнер и мастер Левин создали шестнадцатиразрядную счетную машину.

Как видно, работа над изобретением была длительной, но не непрерывной. Лейбниц одновременно трудился в самых разных областях науки. В 1695 году он писал: «Уже свыше двадцати лет назад французы и англичане видели мою счетную машину... с тех пор Ольденбург, Гюйгенс и Арно, сами или через своих друзей, побуждали меня издать описание этого искусного устройства, а я все откладывал это, потому что я сперва имел только маленькую модель этой машины, которая годится для демонстрации механику, но не для пользования. Теперь же с помощью собранных мною рабочих готова машина, позволяющая перемножать до двенадцати разрядов. Уже год, как я этого достиг, но рабочие еще при мне, чтобы можно было изготовить другие подобные машины, так как их требуют из разных мест».

Работа над калькулятором Лейбницу обошлась в 24 000 талеров. Для сравнения, годовая зарплата министра по тем временам составляла 1 – 2 тысячи талеров.

К сожалению, с полной уверенностью не об одной из сохранившихся моделей калькулятора Лейбница нельзя сказать, что она была создана именно автором. Из-за чего существует много предположений относительно изобретения Лейбница. Есть мнения, что ученый только высказал идею применения ступенчатого валика, или что

он не создавал калькулятор целиком, а только демонстрировал работу отдельных механизмов устройства. Но, несмотря на все сомнения, можно точно утверждать, что идеи Лейбница надолго определили путь развития вычислительной техники.

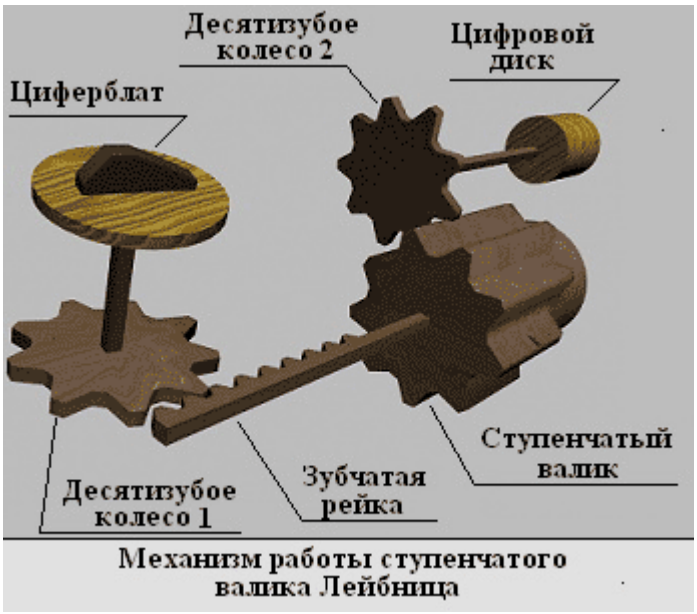
Мы будем вести описание калькулятора Лейбница на основе одной из сохранившихся моделей, находящейся в музее в Ганновере. Она представляет собой ящик около метра длинной, 30 сантиметров шириной и около 25 сантиметров высотой.

Изначально, Лейбниц пытался лишь улучшить уже существующее устройство Паскаля, но вскоре он понял, что операция умножения и деления требуют принципиально нового решения, которое бы позволяло вводить множимое только один раз.

О своей машине Лейбниц писал: «Мне посчастливилось построить такую арифметическую машину, которая бесконечно отличается от машины Паскаля, так как моя машина дает возможность совершать умножение и деление над огромными числами мгновенно, притом не прибегая к последовательному сложению и вычитанию».

Это стало возможно, благодаря разработанному Лейбницем цилиндру, на боковой поверхности которого, параллельно образующей, располагались зубья различной длины. Этот цилиндр получил название «Ступенчатый валик».

К ступенчатому валику крепится зубчатая рейка. Эта рейка входит в сцепление с десятизубым колесом №1, к которому прикреплялся циферблат с цифрами от 0 до 10. Поворотом этого циферблата задается значение соответствующего разряда множимого.



Механизм работы ступенчатого валика Лейбница

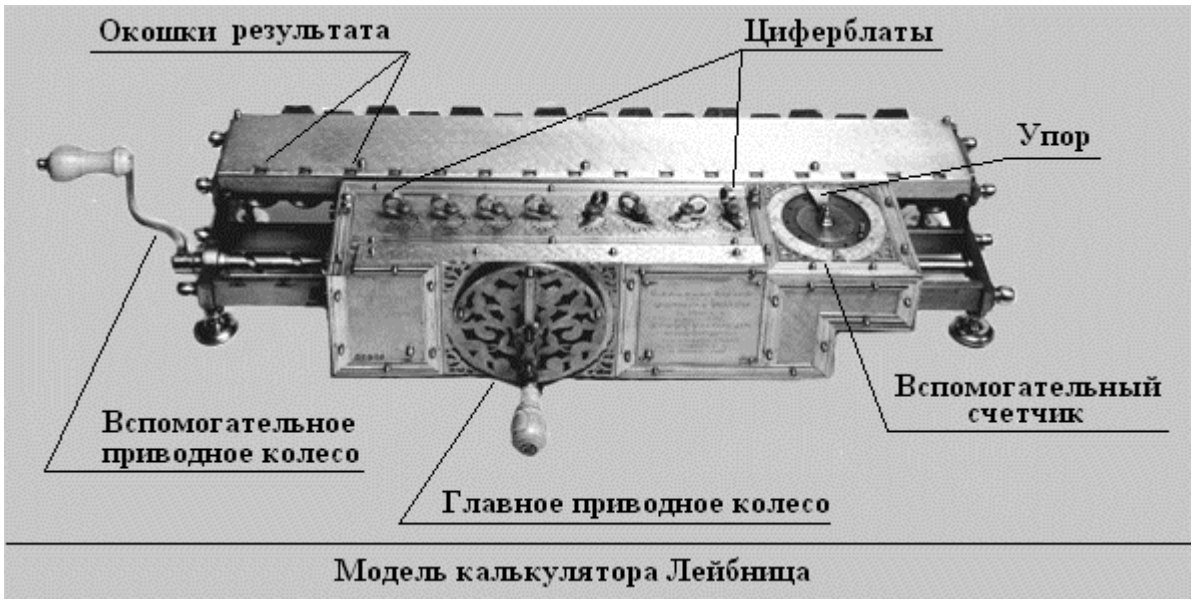
Например, если второй разряд множимого равнялся 5, то циферблат, отвечающий за установку этого разряда, поворачивался в положение 5. В результате десятизубое колесо № 1, с помощью зубчатой рейки, так перемещало ступенчатый валик, что при повороте на 360 градусов он входит в зацеплении с десятизубым колесом № 2 только пятью наиболее длинными ребрами. Соответственно, десятизубое колесо №2 поворачивалось на пять частей полного оборота, на столько же поворачивался и связанный с ним цифровой диск, отображающий результирующее значение выполненной операции.

При следующем обороте валика на цифровой диск снова перенесется пятерка. Если цифровой диск совершал полный оборот, то результат переполнения переносился на следующий разряд.

Поворот ступенчатых валиков осуществлялся с помощью специальной ручки – главного приводного колеса.

Таким образом, при выполнении операции умножения не требовалось многократно вводить множимое, а достаточно вести его один раз и повернуть ручку главного приводного колеса столько раз, на сколько необходимо произвести умножение. Однако, если множитель будет велик, то операция умножения займет длительное время. Для решения этой проблемы Лейбниц использовал сдвиг множимого, т.е. отдельно происходило умножение на единицы, десятки, сотни и так далее множителя.

Для возможности сдвига множимого устройство было разделено на две части - подвижную и неподвижную. В неподвижной части размещался основной счетчик и ступенчатые валики устройства ввода множимого. Установочная часть устройства ввода множимого, вспомогательный счетчик и, главное, приводное колесо располагаются на подвижной части. Для сдвига восьмиразрядного множимого использовалось вспомогательное приводное колесо.



Модель калькулятора Лейбница

Так же для облегчения умножения и деления Лейбниц разработал вспомогательный счетчик, состоящий из трех частей.

Наружная часть вспомогательного счетчика - неподвижная. На ней нанесены числа от 0 до 9 для отсчета количества сложений множимого при произведении операции умножения. Между цифрами 0 и 9 расположен упор, предназначенный остановить вращение вспомогательного счетчика, когда штифт достигнет упора.

Средняя часть вспомогательного счетчика – подвижная, которая служит для отсчета количества сложений при умножении и вычитаний при делении. На ней имеется десять отверстий, напротив цифр внешней и внутренней частей счетчика, в которые вставляется штифт для ограничения вращения счетчика.

Внутренняя часть - неподвижная, которая служит для отчета количества вычитаний при выполнении операции деления. На ней нанесены цифры от 0 до 9 в обратном, относительно наружной части, порядке.

При полном повороте главного приводного колеса средняя часть вспомогательного счетчика поворачивается на одно деление. Если предварительно вставить штифт, например, в отверстие напротив цифры 4 внешней части вспомогательного счетчика, то после четырех оборотов главного приводного колеса этот штифт наткнется на неподвижный упор и остановит вращение главного приводного колеса.

Рассмотрим принцип работы калькулятора Лейбница на примере умножения 10456 на 472:

1. С помощью циферблатов вводится множимое (10456).

2. Устанавливается штифт в среднюю часть вспомогательного счетчика, напротив цифры 2, нанесенной на наружную часть вспомогательного счетчика.

3. Поворачивают главное приводное колесо по часовой стрелки, пока штифт, вставленный в вспомогательный счетчик, не упрется в упор (два поворота).

4. Сдвигается подвижная часть калькулятора Лейбница на одно деление влево, используя вспомогательное приводное колесо.

5. Устанавливается штифт в среднюю часть вспомогательного счетчика, напротив цифры, соответствующей количеству десяток множителя (7).

6. Поворачивается главное приводное колесо по часовой стрелки, пока штифт, вставленный в вспомогательный счетчик, не упрется в упор (семь поворотов).

7. Подвижная часть калькулятора Лейбница сдвигается еще на одно деление влево.

8. Устанавливается штифт в среднюю часть вспомогательного счетчика, напротив цифры, соответствующей количеству сотен множителя (4).

9. Поворачивают главное приводное колесо по часовой стрелки, пока штифт, вставленный в вспомогательный счетчик, не упрется в упор (четыре поворота).

10. Число, появившиеся в окошках отображения результата, – искомое произведение 10456 на 472 (10456 х 472 = 4 935 232).

При делении, сначала, в калькулятор Лейбница вводится делимое с помощью циферблатов, и один раз поворачивается главное приводное колесо по часовой стрелке. Затем, с помощью циферблатов вводится делитель, и главное приводное колесо начинает вращаться против часовой стрелки. При этом результат деления – это количество оборотов главного приводного колеса, а в окошках отображения результатов индицировался остаток от деления.

Если делимое много больше делителя, то для ускорения деления используют сдвиг делителя на необходимое количество разрядов влево с помощью вспомогательного приводного колеса. При этом, во время подсчета количества оборотов главного приводного колеса, необходимо учитывать сдвиг (один оборот главного приводного колеса при сдвиге подвижной части калькулятора Лейбница на одну позицию влево приравнивается к десяти оборотам главного приводного колеса).

Рассмотрим принцип работы калькулятора Лейбница на примере деления 863 на 64:

1. С помощью циферблатов вводим делимое (863).

2. Поворачиваем ручку главного приводного колеса по часовой стрелки один раз.

3. С помощью циферблатов вводим делитель (863).

4. Сдвигаем движущуюся часть калькулятора Лейбница на одну позицию влево с помощью вспомогательного приводного колеса.

5. Поворачиваем главное приводное колесо один раз против часовой стрелки и получаем первую часть результата деления - количество оборотов главного приводного колеса, умноженное на разрядность (положение подвижной части калькулятора). Для нашего случая - это 1х10. Таким образом, первая часть результата деления будет равна 10. В окошках результата отобразится остаток от первой операции деления (223).

6. Сдвигаем движущуюся часть калькулятора Лейбница на одну позицию вправо с помощью вспомогательного приводного колеса.

7. Поворачиваем главное приводное колесо против часовой стрелки до тех пор, пока остаток, отображающийся в окошках результата, не станет меньше делителя. Для нашего случая - это 3 оборота. Таким образом, вторая часть результата будет равна 3. Складываем обе части результата и получаем

частное (результат деления) - 13. Остаток от деления отображается в окошках результата и составляет 31.

Сложение осуществляется следующим способом:

1. С помощью установки циферблатов в необходимое положение, вводится первое слагаемое

2. Поворачивается ручка главного приводного колеса по часовой стрелки один раз.

3. Вводится второе слагаемое по той же технологии, как и первое.

4. Еще раз поворачивается ручка главного приводного колеса.

5. В окне результата отображается результат сложения.

Для вычитания необходимо:

1. С помощью установки циферблатов в необходимое положение, вводится уменьшаемое.

2. Поворачивается ручка главного приводного колеса по часовой стрелки один раз.

3. С помощью циферблатов вводится вычитаемое.

4. Поворачивается ручка главного приводного колеса один раз против часовой стрелки.

5. В окне результата отображается результат вычитания.

Несмотря на то, что о машине Лейбница было известно в большинстве стран Европы, она не получила большого распространения из-за высокой себестоимости, сложности изготовления и ошибок, изредка возникающих при переносе разрядов переполнения. Но основные идеи - ступенчатый валик и сдвиг множителя, позволяющие работать с многоразрядными числами, оставили заметный след в истории развития вычислительной техники.

Идеи, изложенные Лейбницем, имели большое количество последователей. Так, в конце 18 века над усовершенствованием калькулятора работали Вагнер и механик Левин, а после смерти Лейбница – математик Тоблер. В 1710 году машину, аналогичную калькулятору Лейбница, построил Буркхардт. Усовершенствованием изобретения занимались и Кнутцен, и Мюллер, и другие выдающиеся ученые того времени.

Рабдологический абак

Рабдологический абак – суммирующая машина, изобретенная Клодом Перро (25.09.1613 – 09.10.1688), братом знаменитого сказочника Шарля Перро. Впервые, упоминание об этом изобретении встречается в 1700 году в книге «Сборник большого числа машин собственного сочинения», изданной Клодом Перро посмертно. В этой книге описывается множество изобретений автора таких, как «маятниковые часы», «машина для поднятия тяжести», «машина для увеличения эффекта огнестрельного оружия», в том числе и интересующая нас суммирующая машина, значащаяся под номером десять.

Рабдологический абак – компактная и просто устроенная вычислительная машина, отметившаяся в веках истории в основном за счет принципиального отличия от устройства всех предыдущих изобретений в этой области. В рабдологическом абаке взамен зубчатых колес, предложенных Паскалем, используются зубчатые рейки (кремальеры).

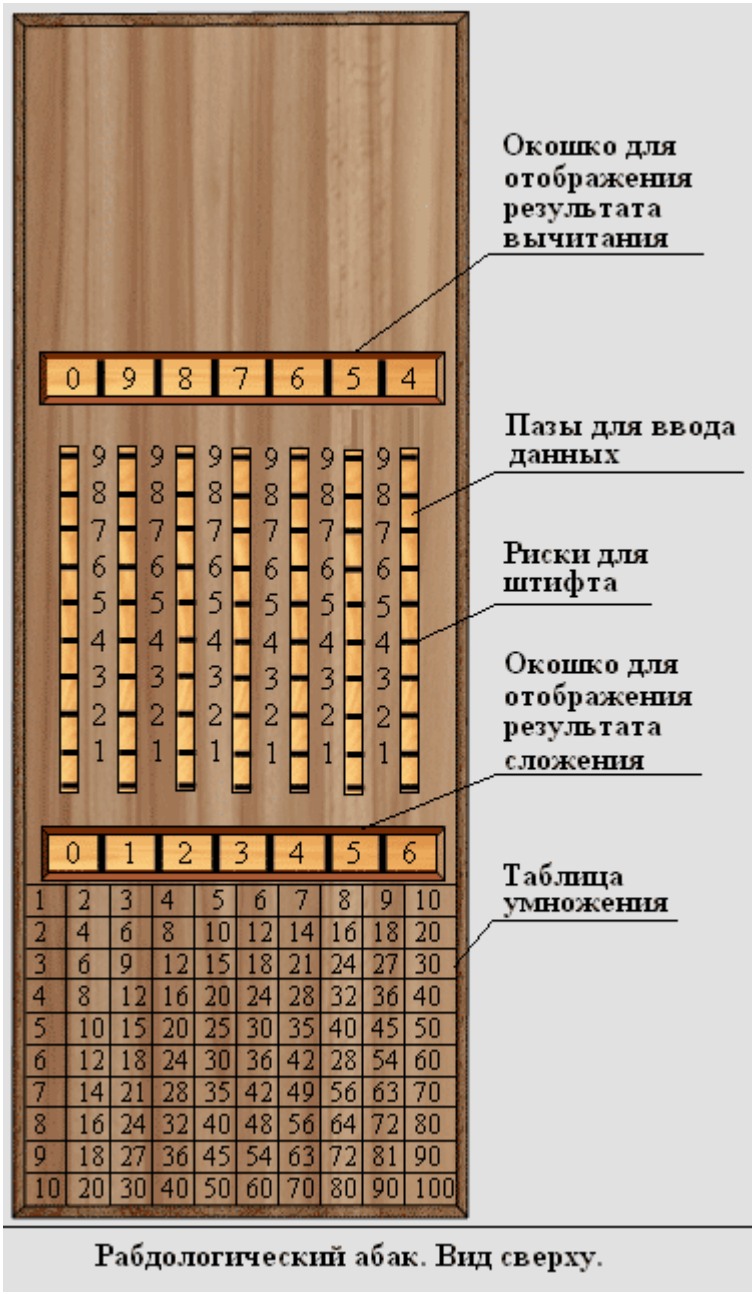
Клод Перро так пишет о своем изобретении: «Я назвал эту машину «рабдологический абак», потому что древние называли абаком небольшую доску, на которой написаны цифры, а рабдологией — науку выполнения арифметических операций с помощью маленьких палочек с цифрами...»

Клод Перро

Счетная машина, действительно, представляла собой пластину в палец толщиной, примерно 30 сантиметров высотой и 14 – шириной. На лицевой стороне машины было вырезано два окошка для отображения результатов. В верхнем окошке показывался результат вычитания, а в нижнем – сложения.

В нижней части лицевой стороны была выгравирована таблица умножения.

Также на лицевой стороне было прорезано семь пазов, вдоль которых были нанесены шкалы с делениями 1,2,3…9. В пазах виднелись линейки, которые можно было передвигать вверх и опускать вниз, к основанию машины с помощью штифта с заостренным кончиком.



Рабдологический абак. Вид сверху.

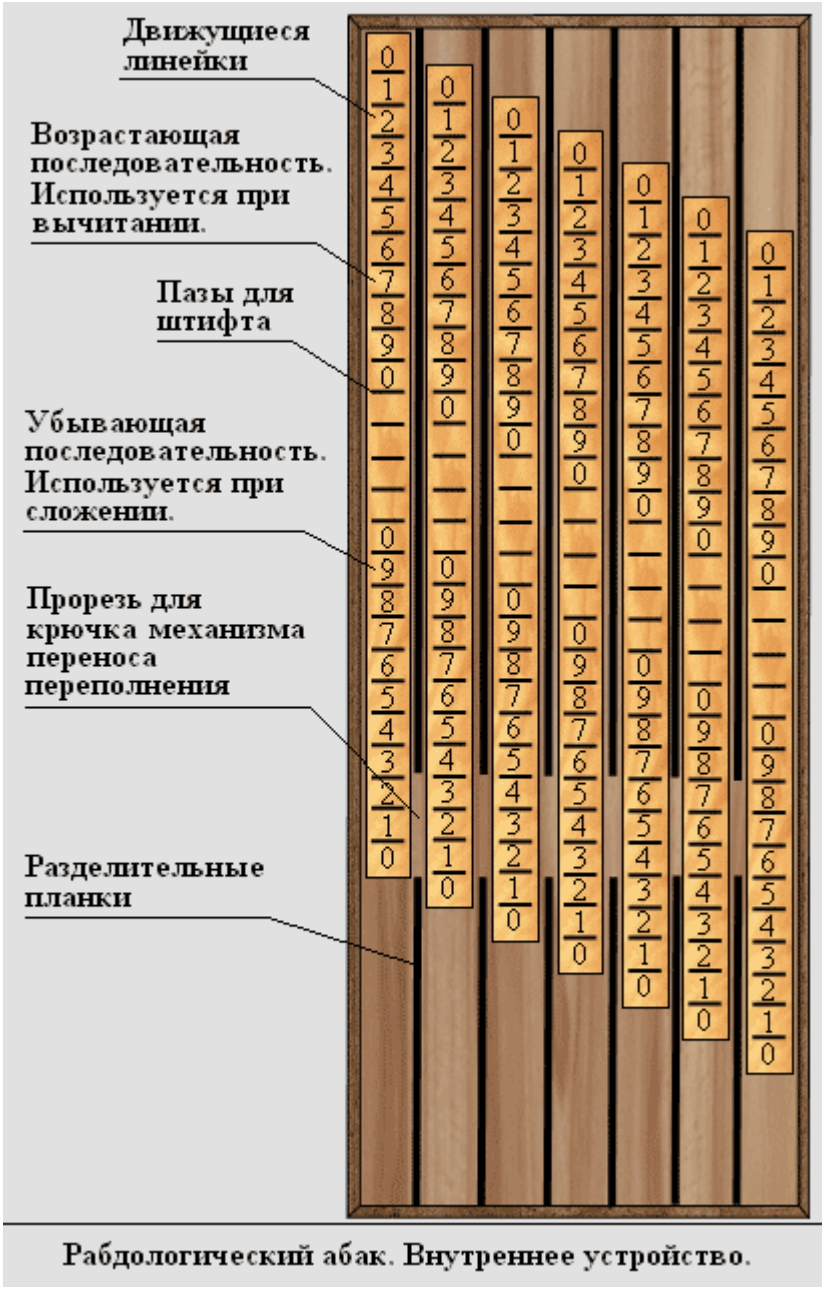
Каждая линейка была разделена на 26 частей глубокими рисками, используемыми для перемещения линеек (в риски вставлялось острие штифта, что позволяло легко и точно перемещать линейки). В верхних одиннадцати делениях линейки находилась возрастающая последовательность (0,1,2,3,4,5,6,7,8,9,0), используемая при вычитании. В нижних одиннадцати делениях находилась убывающая последовательность (0,9,8,7,6,5,4,3,2,1,0), используемая для сложения. Цифры этих последовательностей отображались в окошках результата, являясь искомой величиной сложения (нижнее окошко) или вычитания (верхнее окошко).

Последовательности, нанесенные на линейки, отделялись друг от друга четырьмя пустыми делениями.

Всего в устройстве использовалось семь линеек, отделенных друг от друга тонкими пластинками. Крайняя правая линейка символизировала разряд единиц, следующая за ней – разряд десятков и так далее, вплоть до разряда миллионов.

В каждой пластинке, отделяющей линейки друг от друга, имелось отверстие, используемое для переноса переполнения из младшего разряда в старший. Это

отверстие находилось у основания линейки, придвинутой к самому верху устройства, и было длинной в три деления линейки.



Рабдологический абак. Внутреннее устройство.

У основания правой стороны каждой линейки располагалось 11 зубцов, по одному на деление линейки. С другой стороны линейки (под 11 и 12 делением, если считать снизу) располагался подпружиненный крючок. Зубцы и крючок использовались для переноса переполнения из младшего разряда в старший, происходившего следующим образом.

Когда планка младшего разряда находилась в верхней части рабдологического абака, то есть в окошках результата сложения и вычитания находились нули, подпружиненный крючок был скрыт в теле линейки, упираясь в пластинку, располагающуюся слева от линейки. По мере продвижения линейки вниз, подпружиненный крючок приближался к отверстию в пластинке. Когда в окошке результата сложения появлялась цифра семь, принадлежащая передвигаемой линейке, подпружиненный крючок начинал выдвигаться в отверстие планки, и в момент, когда в окошке результата появлялась цифра девять, входил в зацепление с зубцами линейки старшего разряда. Дальнейшее передвижение линейки младшего разряда приводило к перемещению линейки старшего разряда. Таким образом, когда в окошке результатов сложения появлялась цифра ноль линейки младшего разряда, следующая за цифрой девять, линейка старшего разряда передвигалась, за счет сцепления крючка с зубцами, ровно на одно деление вниз. Механика переноса разряда переполнения иллюстрируются на анимации «Рабдологический абак. Устройство переноса переполнения», расположенной ниже.

Рассмотрим операцию сложения на Рабдологическом абаке на примере «127+65»:

1. Устанавливаем все разряды рабдологического абака в ноль, для чего, с помощью штифта, передвигаем линейки в крайнее верхнее положение.

2. Ставим штифт в паз на риску линейки младшего разряда, находящуюся напротив цифры семь, и сдвигают линейку до тех пор, пока штифт не упрется в нижний торец паза. При этом в окошке результата сложения в младшем разряде отобразится вводимое число (семь). Стоит заметить, что в окошке результата вычитания в младшем разряде будет отображено число, необходимое для дополнения семерки до десяти, то есть три.

3. Аналогичную операцию проделываем и с разрядом десяток, только в этом случае паз устанавливаем напротив цифры 2.

4. Для разряда сотен паз устанавливаем напротив цифры один и передвигаем его вниз до упора, то есть на одно деление. В результате, в окошке результата сложения будет отображено число 127.

Рабдологический абак.
Устройство движущейся линейки.

5. Переходим к вводу второго слагаемого. Устанавливаем штифт в паз на риску линейки младшего разряда, находящуюся напротив цифры пять, и сдвигаем линейку вниз до упора. При этом штифт остановится напротив цифры два, так как линейка упрется в нижнюю стенку устройства раньше, чем штифт достигнет нижнего торца паза, а линейка разряда десяток опустится на одно деление вниз, за счет работы механизма переноса переполнения. В окне результата сложения отобразится цифра 130.

6. Для того, чтобы получить верную цифру в разряде единиц (то есть 2), следует, не извлекая штифта из прорези, продвинуть линейку вверх, пока штифт не упрется в торец паза. Таким образом, в окне результата сложения отобразится цифра 132.

7. Вводим разряд десятков второго слагаемого, для чего устанавливаем штифт в паз на риску линейки разряда десятков, находящуюся напротив цифры шесть, и сдвигаем линейку вниз до упора. На этом вычисления заканчиваются, а в окошке результата сложения отображается искомое значение (192).

При вычитании с помощью рабдологического абака использовался прием дополнения до десяти, аналогичный способу, используемому в Паскалине. Поясним способ дополнения на примере.

Допустим, необходимо решить уравнение: Y=68-23=45. С помощью метода дополнения, представим число 68, как разность чисел 100 и 32 (68=10-32). Таким образом, наше уравнение сводится к следующему виду: Y=68-23=100-32-23=100-(32+23)=27. Как видно из преобразования, вычитание заменилось на сложение и вычитание результата сложения из 100, что есть преобразование, обратное дополнению. Следовательно, остается решить задачу автоматического дополнения до десяти, для чего на всех линейках нанесено две последовательности цифр, а на крышке рабдологического абака - два окошка вывода результата, расположенных так, что суммы двух чисел, отображенных в окошках и располагающихся друг под другом, всегда равняются десяти.

Рабдологический абак.
Устройство переноса переполнения.

Рассмотрим работу рабдологического абака при вычитании на примере 68-23:

1. Устанавливаем все разряды рабдологического абака в ноль, для чего с помощью штифта передвигаем линейки в крайнее верхнее положение.

2. Вводим младший разряд уменьшаемого. Передвигаем линейку младшего разряда так, чтобы в окошке результата вычитания в позиции младшего разряда отобразилась цифра 8, для чего ставим штифт в паз на риску, находящуюся напротив цифры два, и сдвигаем линейку вниз до тех пор, пока штифт не упрется в торец паза.

3. Вводим разряд десяток уменьшаемого, для чего передвигаем соответствующую линейку так, чтобы в окошке результата вычитания на второй позиции отобразилась цифра 6. Для этого ставим штифт в паз на риску, находящуюся напротив цифры четыре, и сдвигаем линейку вниз до тех пор, пока штифт не упрется в торец паза. В результате, в окне результата вычитания отобразится число 68.

4. Водим вычитаемое также, как слагаемое при сложении. В нашем случае, для ввода младшего разряда вычитаемого, устанавливаем штифт в паз на риску линейки младшего разряда, напротив цифры 3, и двигаем линейку вниз, пока штифт не упрется в торец паза. Далее вводим старший разряд вычитаемого, для чего устанавливаем штифт в паз на риску второй слева линейки, напротив цифры 2, и двигаем линейку вниз, пока штифт не упрется в торец паза. На этом нахождение разницы двух чисел с помощью рабдологического абака заканчивается, а в окошке результата вычитания отображается искомая разность (45).

Как видно из описания рабдологического абака, несмотря на принципиальные отличия от существующих на тот момент вычислительных машин, его устройство и использование были очень просты. Однако, он не получил распространения. Возможная причина этому - ненадежное устройство подпружиненного крючка, часто выходившего из строя при постоянной эксплуатации, из-за низкого уровня механики конца XVII века. Но, тем не менее, идеи, предложенные Клодом Перро, впоследствии нашли применение в ряде простых и надежных счетных приборах таких, как «счислитель Куммера», «Комптатор Ганса Забельного» и некоторых других.

Счетная машина Перейры.

Родригес Перейра

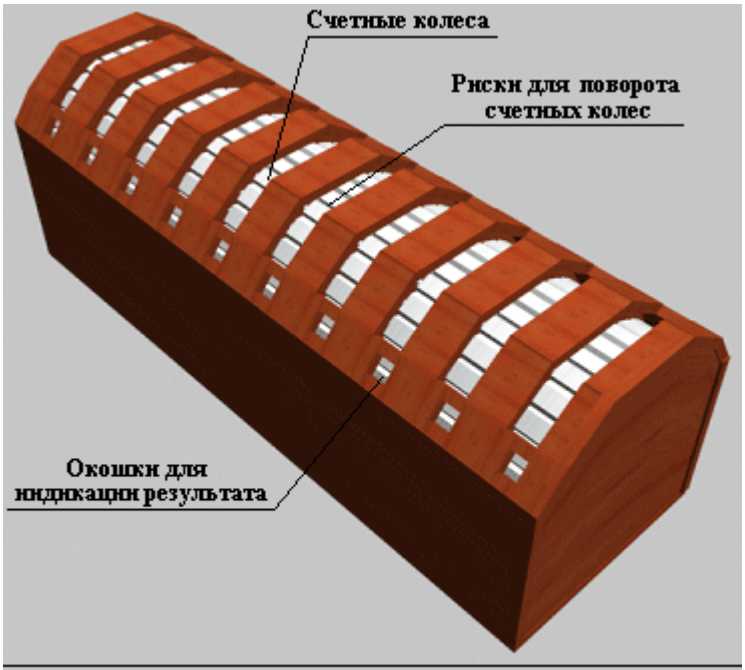
Хакоб Родригес Перейра (1715-1780) нашел совершенно иное применение счетным машинам, нежели все его предшественники. Сконструированное по собственным чертежам устройство, господин Перейра использовал для обучения счету глухонемых детей. История изобретения - очень занимательная, и в ней не обошлось без женщины.

Когда будущему учителю исполнилось восемнадцать лет, он отправился в Бордо с намерениями изучить медицину, где встретил прелестную, но глухонемую от рождения девушку. Перейра проявил к ней сильную симпатию, и с тех пор целью его жизни стало изыскание средства обучения глухонемых разговорной речи и грамоте.

После окончания медицинских курсов Родригес Перейра, несмотря на то, что жил довольно скромно и содержал многочисленную родню, берет на обучение глухонемых детей бедняков и оттачивает на них свои методы обучения. Слухи о талантливом учителе быстро распространяются, и в 1754 году, после успешной демонстрации результатов своего воспитания, господину Перейре дают на воспитание сына богатого землевладельца. С тех пор дела Родригеса Перейры стали налаживаться. Одним из составляющих успеха работы Перейры было использование в обучении глухонемых детей счетной машины собственного изобретения.

11 июня 1749 года, демонстрируя результаты своей работы комиссии Академии наук, господин Перейра рассказал и о своей счетной машине, и о методах обучения глухонемых с ее помощью. Отчет комиссии о работе Родригеса Перейры был опубликован 5 мая 1751 года в «Журнале ученых» и содержал весьма лестные отзывы: «Виденных нами результатов метода г-на Перейры вполне достаточно, чтобы еще раз подтвердить мнение... что такой метод обучения глухонемых в высшей степени практичен и что лицо, которое применяло его с таким успехом, достойно похвалы и поощрения...». В этой же статье приводилось описание счетной машины Перейры, но, к сожалению, в журнале не были опубликованы чертежи изобретения, так, что о некоторых деталях машины теперь можно только догадываться, но, тем не менее, это изобретение стоит в одном ряду с работами таких столпов науки, как Паскаль, Морленд, Полени и других выдающихся изобретателей того времени.

Достоверно известно, что, по крайней мере, один экземпляр счетной машины был изготовлен, но он не сохранился до наших дней, поэтому восстановить облик машины невозможно, и приходится довольствоваться лишь описанием, приведенным в «Журнале ученых».



Модель счетной машины Перейры, построенная на основе описания в "Журнале ученых"

От сконструированных на тот момент машин изобретение Перейры отличается, в первую очередь, расположением счетных колес. Они располагались не на двух осях, а на одной оси, проходящей через все устройство, что делало машину компактной, простой и удобной в обращении.

Вот как описывалась счетная машина Родригеса Перейра в «Журнале наук»:

“Мы проверили по приказу Академии наук исследование арифметической машины, представленной г-ном Перейра, методы обучения немых которого уже одобрены академией наук.

Г-н Перейра сообщил в документе, который он читал в Академии наук 16 декабря прошлого года, что на данный момент все известно о машинах такого типа, среди которых самыми известными являются изобретения Паскаля, Перро, Лепина и Боистиссандеа. Первое и два последних изобретения имеют довольно большой размер и состоят из множества колес, пружин, трещоток (устройств, позволяющих осуществлять непрерывное движение в одну сторону, при этом, предотвращая движение в противоположную сторону) и других частей, что делает их дорогостоящими, с учетом ремонта, и неудобными в использовании.

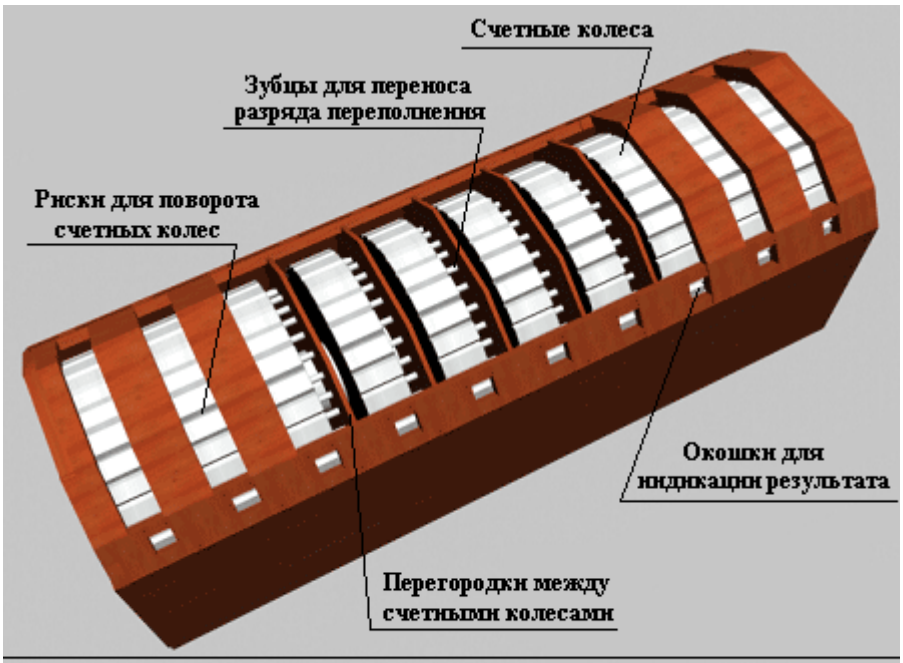
Рабдологический абак г-на Перро намного проще, и именно на этот инструмент машина Перейры похожа больше всего. Рабдологический абак состоял из небольших линеек, каждая из которых содержала две колонки чисел, располагающихся друг под другом и разделенных пустым пространством.

Числа в первом столбце располагались по порядку от 0, 1, 2, 3 и т.д. до 9, во втором столбце числа располагались в обратном порядке, 9, 8, 7 и т.д. до 0. Между числами размещались канавки, использующиеся для передвижения

линеек. После приближения линейки к нижнему положению, подпружиненный крючок, который встроен в тело линейки, через открывшееся окошко входит в паз в соседней линейки, передвигая ее на один шаг, чтобы отметить десятки единиц первой линейки. Если линейка дойдет до конца, и, следовательно, число в нижнем окошке будет отсутствовать, необходимо перевести линейку вверх, пока указатель не достигнет крайнего верхнего положения, индицируя в окошке, в совокупности с разрядом десятков, искомое число. Подробно, как это работает, описано в инструкции использования инструмента.

Вместо того, чтобы размещать две колонки друг над другом на каждой линейке, г-н Перро мог бы разместить их рядом, одну немного выше, чем другую, и соответственно этому, расположить индицирующие окошки, в которых отображаются нужные номера. Таким образом, не нужно было бы делать счеты такими длинными.

В машине господина Перейры используется эта идея, но более изобретательно.



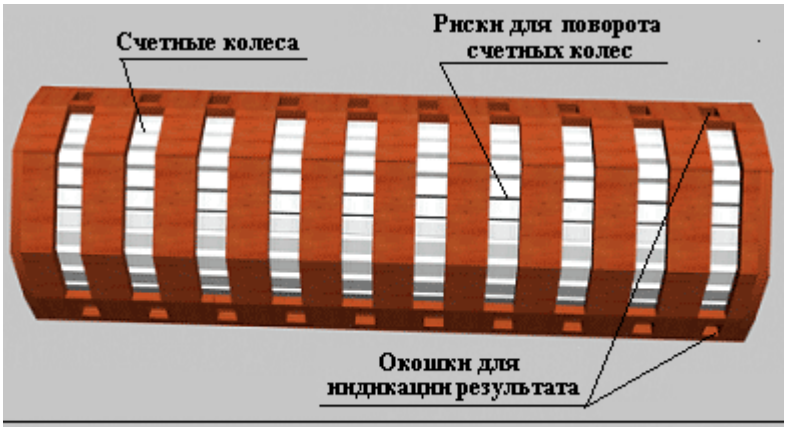
Модель счетной машины Перейры, построенная на основе описания в "Журнале ученых"

Вместо линеек он использует небольшие деревянные колеса или очень короткие цилиндры, размещенные на общей оси. Цилиндрическую поверхность этих колесиков, представляющих собой линейки бесконечной длинны, он разделил по окружности на тридцать равных частей, в которых написал две последовательности чисел. Первая - три раза повторяет последовательность 1 2 3 4 5 6 7 8 9 0, вторая - три раза повторяет последовательность 0 9 8 7 6 5 4 3 2 1. Из этих колес: одно использовалось для подсчета Денье, одно - для Су, одно - для простейших дробей: 1/2, 1/3, 1/4, 1/6 и так далее, и семь - для целых чисел: единиц, десятков, сотен, тысяч и так далее до миллиона. Все эти колесики образуют один цилиндр, 7.6 сантиметров длиной и 1.8 сантиметров в диаметре, заключенный в компактный корпус.

На верхней части корпуса расположены окошки, которых столько же, сколько колесиков. Каждое окошко - размером с одну треть окружности цилиндра. Колесики поворачиваются через эти окошки с помощью кончика пера так же, как и в машине господина Перро, выбирая число, которое нужно, от 1 до 9 и 0. При использовании колесиков есть преимущество. Числа столбцов идут друг за другом, и не нужно поворачивать колесики в обратную сторону для получения результата, что часто требовалось при работе с абаком Перро.

Г-н Перейра разделил окружности колеса на тридцать частей, а не на двадцать, так, что окошки, находящиеся в верхней части окна, занимали лишь одну третью часть длины окружности колеса, а не половину, что было бы не так удобно. Он мог бы разделить их на сорок или пятьдесят частей, тогда окошки занимали бы 1/4 или 1/5 длины окружности колеса, но большее число делений потребовало бы увеличения диаметра колесиков.

Кроме того, в верхней части корпуса, в котором располагались колесики, есть два ряда окошек по всей длине крышки. Один ряд - вверху крышки, а второй - внизу. Один ряд использовался для нахождения суммы или произведения, другой - использовался для нахождения разности или результата деления.



Модель счетной машины Перейры, построенная на основе описания в "Журнале ученых"

Средства, которые г-н Перейра нашел для переноса переполнения, - очень изобретательны. Для этого на одной из плоских сторон каждого колеса он сделал тридцать зубов, более или менее похожих на зубья зубчатого колеса. Другая сторона была зарезервирована для размещения небольшого баланса, имеющего на одной стороне крючок, а на другой наклонную плоскость. Каждый раз, когда колесо поворачивалось на десять делений, наклонная плоскость наталкивалась на зуб, закрепленный на железной пластине между двумя колесами.

Этот зуб вталкивал наклонную плоскость в паз в теле колеса, тем самым другая часть баланса с крючком проходила через пластину, которая имела отверстие, специально для этой цели, зацепляясь за один из тридцати зубцов соседнего колеса, что переводило его на один шаг вперед. После того, как этот шаг был сделан, наклонная плоскость проходила зуб на железной пластине, возвращаясь назад на свое место с помощью пружины, таким образом, крючок возвращался в тело колеса и убирался из соседнего колеса.

Вся эта схема представляется нам хорошо продуманной, простой и удобной, и мы считаем, что она достойна одобрения и включения в перечень машин, утвержденных Академией. В связи с вышесказанным, есть еще две вещи, на которые, мы считаем, хорошо бы обратить внимание читателей:

1. С помощью арифметической машины господина Перейры можно обойтись без карандаша и бумаги, выполняя арифметические операции над фунтами, шиллингами, пенсами, а также числами до семи разрядов. Последняя особенность машины господина Перейры является уникальной и полезной, так как позволяет складывать и вычитать дроби с различными знаменателями с такой же легкостью, как если бы действия выполнялись над целыми числами.

2. Машина господина Перейры, размер которой указан выше, не должна иметь большую цену, но, тем не менее, те, кото готовы приобрести машину, должны рекламировать господина Перейру, так как цена машины может быть ниже, из-за увеличения числа закупок. Почитатели механических изделий будут рады приобрести эту машину, вызывающую высокий интерес, и весьма полезную не только для немых детей, но и для желающих изучать науку чисел.

Как видите, в счетной машине Перейры использованы кое-какие идеи, заимствованные у Паскаля и Перро, но, в общем, она представляет собой совершенно оригинальную конструкцию, оставившую заметный след в истории развития счетных машин.

Модель вычислительной машины Якобсона

Счетная машина Якобсона.

Одна из первых отечественных

счетных машин была создана Евной Якобсоном. Точное время изготовления счетной машины неизвестно, но его можно установить, основываясь на орнаменте и надписях, украшавших верхнюю крышку устройства, способах обработке деталей и других косвенных признаках, позволяющих датировать сохранившуюся модель счетного устройства 18-ым веком.

Внешне изобретение Якобсона представляло собой латунную коробку, длинной 34.2 см, шириной 21.8 см и высотой 3.4 см, располагающуюся на четырех маленьких ножках, высотой около сантиметра. Верхняя крышка была искусно украшена и содержала ряд надписей:

Mechanische Rechnungs Mashine (нем. Механическая счетная машина)

Mechina Mechaniszna do Rachunku (польск. Механическая счетная машина)

Zu der Aufgabe des Addirens, Subtantirens, Multiplicirens, und Devidirens von den Nummer Eins biz kann man hier in der Bruchen zertheilen (нем. Для задачи сложения, вычитания, умножения и деления от числа один до тысячи миллионов, и остающееся от деления можно здесь же расчленить на дроби.).

Erfunden und verfertigen von dem Hebreer Jawna Jacobson, Uhrmacher und Mechanicis in der Stadt Nieswiez in Lithauen, Gouvernement Minsk. (нем. Изобретена и изготовлена Евной Якобсоном, часовым мастером и механиком, в городе Несвиже Литвы, Минское воеводство.).

Именно последняя фраза, несмотря на то, что в ней явно не указана дата создания машины, позволила довольно точно ее определить. Дело в том, что город Несвиж был городом Минского воеводства только до 1793 года, а, следовательно, время изготовления машины должно быть не позднее этой даты. Если принять во внимание расцвет науки и искусства, пришедшийся на 40-80 года 18 века в городе Несвиж, когда под меценатством Михаила Радзивила, избравшего этот город своей резиденцией с 1726 года, было создано огромное число ценных работ, оставивших весомый след в истории искусств, то можно предположить, что и создание счетной машины Якобсона пришлось именно на этот период. К сожалению, точнее оценить дату создания машины не представляется возможно, ввиду отсутствия более развернутых сведений о биографии великого изобретателя.

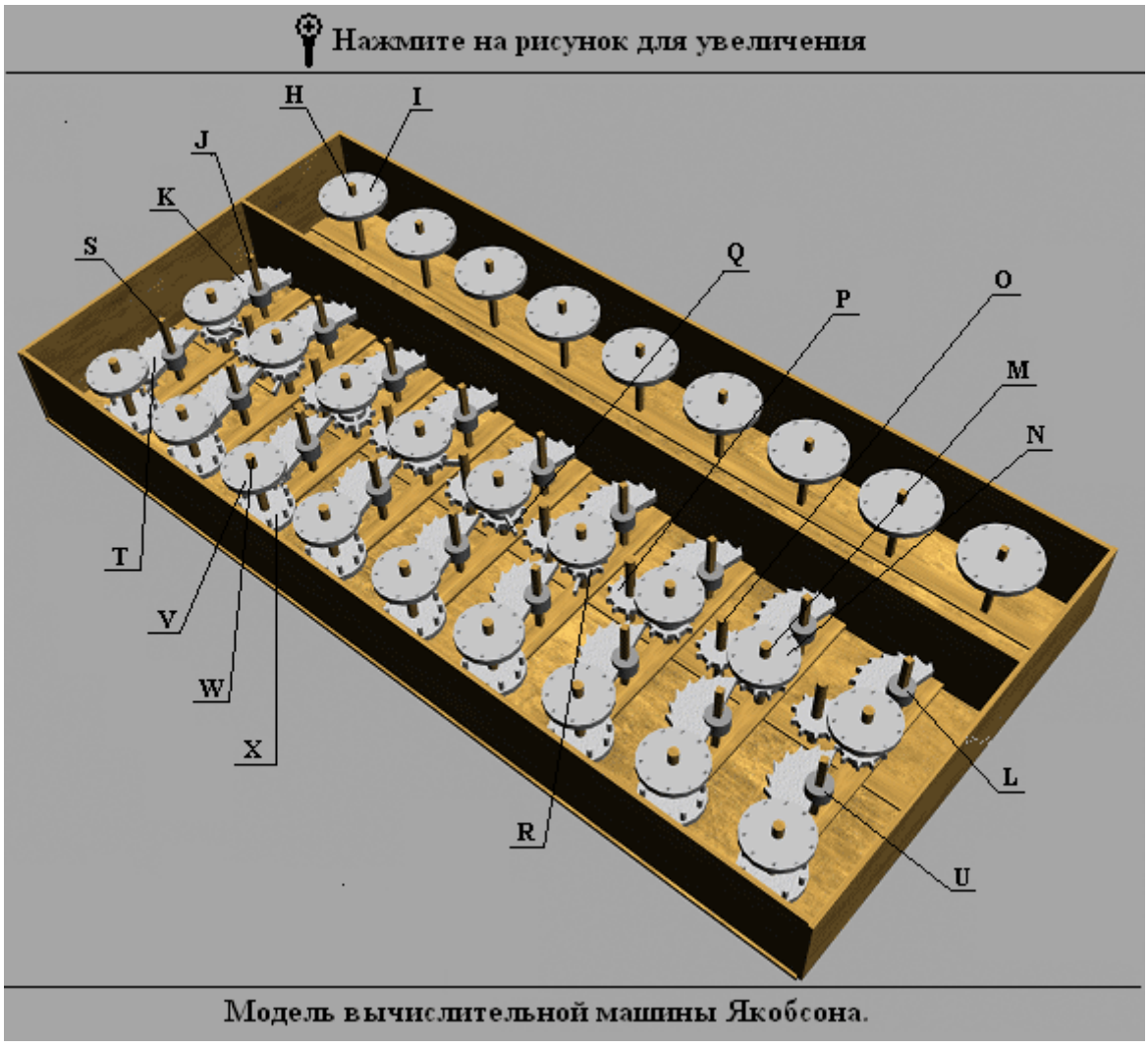
Модель вычислительной машины Якобсона

В отличие

от биографии механика, его изобретения прекрасно сохранилось и на данный момент находится в музее имени М.В. Ломоносова в Санкт-Петербурге. Конструкция вычислительной машины - очень любопытна и заслуживает детального изучения.

Механизм счетной машины закреплялся с нижней стороны верхней крышки, управление осуществлялось с помощью специальных ключей через отверстия в этой же крышке. Всего на верхней стороне устройства располагалось семь рядов отверстий, по девять штук в каждом ряду.

Ряд А использовался для сохранения промежуточных результатов вычисления. Через отверстия этого ряда были выведены девять поводков (H), имевших квадратные сечения (2 на 2 миллиметра). Эти поводки поворачивались с помощью специального ключа. Каждый поводок был жестко соединен с расположенным под крышкой диском (I). На каждом диске (I) были нанесены цифры от 0 до 9, которые просматривались в окошках ряда B. Эти диски не были связаны между собой и с другими частями счетной машины. Единственное их назначение – сохранение промежуточных результатов вычисления для облегчения работы с устройством. Поводки ряда А поворачивали таким образом, чтобы в окошках ряда В отобразилось число, которое необходимо запомнить для использования в последующих вычислениях.

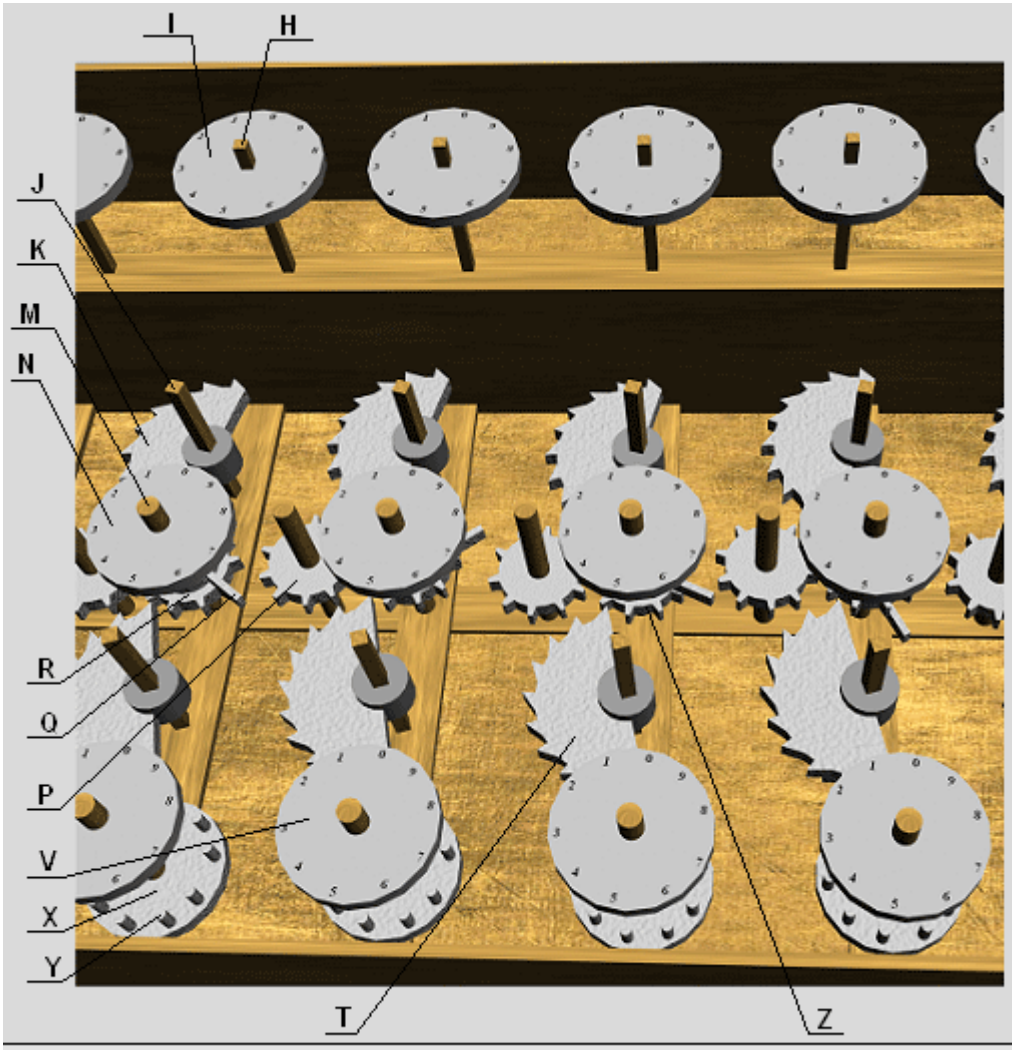


Ряд С содержал такие же поводки (J), как и ряд А, однако, использовались они для проведения операций сложения и умножения. При этом результат операций не должен превышать 109.

Над каждым поводком ряда C была нанесена дуга с числами, располагающимися по часовой стрелке: 0,1,2,3,4,5,6,7,8,9. Рядом с дугой находилась надпись, определяющая разрядность цифры. Дуги располагались в порядке возрастания разрядности справа на лево: Hundert Million (), Zehn Million (десятки миллионов), Eines Million (миллионы), Hundert Tausend (сотни чисел), Zehn Taisend (десятки чисел), Eines Tausend (тысячи), Hundert (сотни), Zehn (десятки), Eins (единицы).

Под каждым из поводков ряда C в круглых углублениях находятся квадратные окошки (D). В этих окошках отображается результат проведенной операции.

Операции сложения и умножения на машине Якобсона производились следующим образом. Вводится первое слагаемое, начиная с младшего разряда. Для этого с помощью ключа последовательно поворачиваются поводки J против часовой стрелки. На поводке J жестко закреплена зубчатая гребенка (K), поворачивающаяся вместе с поводком и входящая в зацепление с зубчатым колесом R, закрепленным на одной оси (M) с цифровым диском (N). Таким образом, цифровой диск (N) поворачивался так, что в окошке (D) отображался введенный разряд первого слагаемого. После того, как установочный ключ отпускался, поводок (J) вместе с гребенкой (K) возвращался в исходное состояние под действием пружины (L). Аналогично, вводились следующие разряды первого слагаемого.



Модель вычислительной машины Якобсона

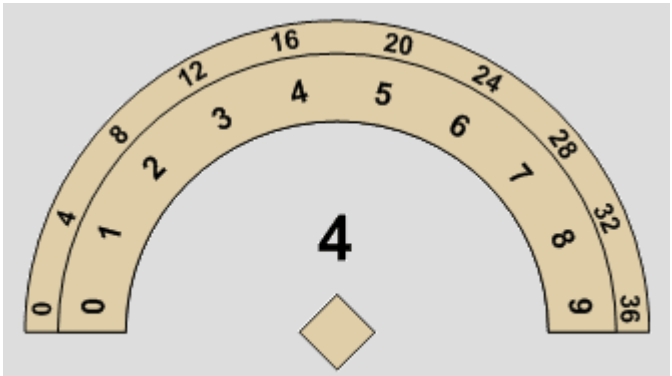
Далее с помощью этих же поводков (J) вводилось второе слагаемое, по описанному выше алгоритму. Если при вводе очередного разряда слагаемого происходило переполнение, то разряд

переполнения переносился в старший разряд.

Происходило это так. На оси (M), к которой был прикреплен цифровой диск (N), располагался однозубый диск(Q), входящий в зацепление с зубчатым диском (P) в момент переполнения разряда и поворачивающий его на одно деление. Зубчатый диск, в свою очередь, входил в зацепление с зубчатым диском (Z), поворачивающим цифровой диск старшего разряда на одно деление.

По окончанию ввода второго слагаемого в окошках (D) отображалась сумма, и можно было начинать ввод третьего слагаемого или вычитание какого-либо числа.

Умножение осуществлялось путем последовательного ввода множимого столько раз, на сколько требовалось его умножить. Для облегчения умножения, под дугами поводков ряда (F) была нанесена таблица умножения, выглядевшая следующим образом. Около каждого была нанесена цифра. У крайнего левого поводка – 1, у второго слева - 2 и так далее до 9. Над цифрами дуговых шкал располагался еще один ряд чисел, символизирующий умножение чисел дуговой шкалы на цифру, выгравированную рядом с поводком. На рисунке приведен пример элемента таблицы умножения, выгравированной на машине Якобсона.



Элемент таблицы умножения выгравированный на машине Якобсона

Ряд поводков (I) использовался для установки счетной машины в исходное состояние после проведения операций сложения, вычитания, умножения или деления. Поводки (I) – это стержни (M), на которых закреплены: цифровой диск (N), зубчатое колесо (R) и однозубое колес переноса переполнения (Q). Поворачивая эти поводки, пользователь устанавливал цифровые диски так, чтобы в окошках результата отображались нули.

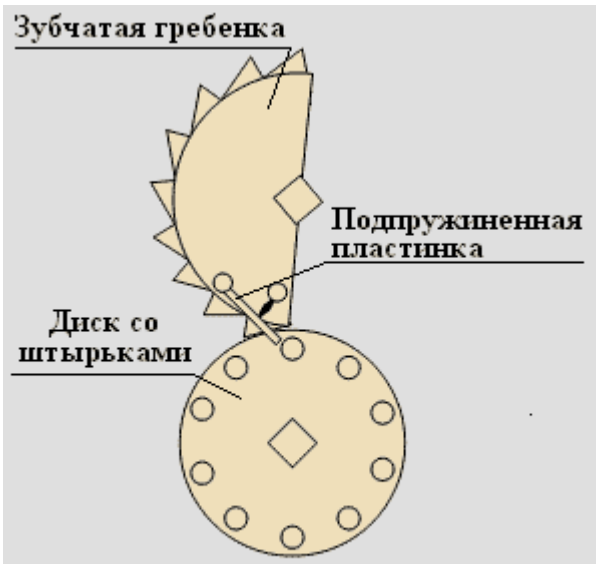
Модель вычислительной машины также позволяла проводить вычитание и деление. Для этого использовались поводки ряда (F). Организация процесса вычитания и деления было организована схоже с процессом сложения. Внешне для пользователя счетной машины это отличие заключалось в направлении поворота поводков (W). Так при сложении они поворачивались против часовой стрелки, а при вычитании - по часовой стрелке.

Если рассмотреть вычитание на уровне устройства механизма счетной машины, то организация вычитания также схожа со сложением. Поводок (W) поворачивал зубчатую гребенку (T), которая входила в зацепление с

зубчатым колесом (Z), поворачивая его. При этом поворот зубчатого колеса осуществлялся в противоположную сторону, относительно поворота этого же зубчатого колеса при сложении. После окончания ввода вычитаемого в окошках отображения результата (D) показывался результат вычитания. Если при вычитании величины уменьшаемого разряда не хватало, то от старшего разряда отнималась единица и прибавлялась к младшему. Осуществлялось это автоматически, с помощью того же механизма, который использовался для переноса переполнения при сложении.

Модель счетной машины Якобсона

Деление выполнялось, как последовательное вычитание с автоматическим подсчетом количества произведенных вычитаний. Для установки делимого использовались поводки сложения (ряд C), а делитель вводился с помощью поводков вычитания (ряд F) до тех пор, пока в окошках результата не отображалось число, меньшее делителя. При этом количество вводов делителя автоматически отображалось в окошках ряда (G).



Устройство механизма подсчета количества ввода делителя

Автоматический подсчет количества вычитаний происходил следующим образом. Когда водился делитель с помощью поводков ряда (F), одновременно с ними, по часовой стрелке, поворачивались зубчатые гребенки (T). Внизу гребенки располагалась подпружиненная планка. При повороте по часовой

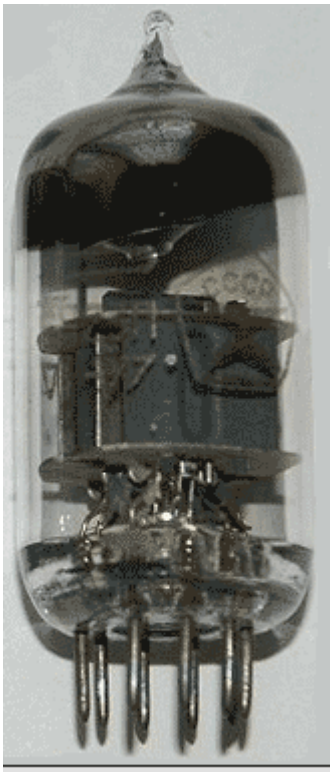
стрелке подпружиненная планка задевала за штыри (Y) диска (X) и заходила в паз гребенки. Когда нижняя часть гребенки отворачивалась от штырьков, подпружиненная планка выходила из паза.

После окончания ввода разряда делителя гребенка (T) под действием пружины (U) возвращалась в исходное состояние, при этом, расположенная в нижней части гребенки, подпружиненная пластинка входила в зацепление со штырьками (Y) диска (X), поворачивая его на одно деление. Диск (X) был жестко закреплен на стрежне (W) с цифровым колесом (V), которое также поворачивалось на одно деление. Таким образом, в окошках ряда (G) отображалось число, показывающее количество вводов всех разрядов делителя.

Как видно из описания, устройство машины Якобсона было удобным для обращения и надежным, а размещение зубчатых колес в несколько уровней делало его еще и компактным. Были продуманы мельчайшие детали, вплоть до маркировки деталей одного разряда одним и тем же знаком, чтобы при разборке и сборке машины, в случае ремонта, они не были перепутаны, и небольших угублений в пазах для ключей установки, позволяющие точно фиксировать поворотные ключи в требуемом положении.

Устройство, изобретенное Якобсоном, было выдающимся достижением научной мысли своего времени и заслуженно занимает почетное место в истории вычислительной техники.

Первое поколение ЭВМ.



Электронная лампа

Первое поколение ЭВМ создавалось на электронных лампах в период с 1944 по 1954 гг.

Электронная лампа – это прибор, работа которого осуществляется за счет изменения потока электронов, двигающихся в вакууме от катода к аноду.

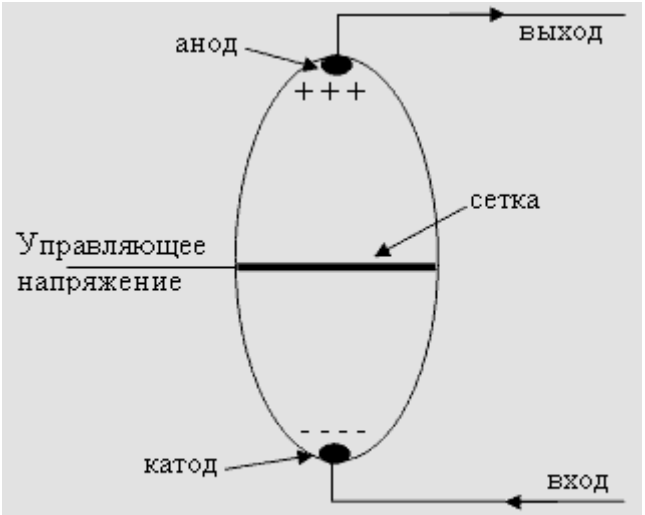
Движение электронов происходит за счет термоэлектронной эмиссии – испускания электронов с поверхности нагретых металлов. Дело в том, что металлы обладают большой концентрацией свободных электронов, обладающих различной энергией, а, следовательно, и скоростями движения. По мере нагревания металла энергия электронов возрастает, и некоторые из них преодолевают потенциальный барьер на границе металла.

Принцип работы электронной лампы следующий. Если на вход лампы подается логическая единица (например, напряжение 2 Вольта), то на выходе с лампы мы получим либо логический ноль (напряжение менее 1В), или логическую единицу (2В). Логическую единицу получим, если управляющее напряжение отсутствует, так как ток беспрепятственно пройдет от катода к аноду. Если же на сетку подать отрицательное напряжение, то электроны, идущие от катода к аноду, будут отталкиваться от сетки, и, в результате, ток протекать не будет, и на выходе с лампы будет логический ноль. Используя этот принцип, строились все логические элементы ламповых ЭВМ.

В простейшем случае катодом служит нить из тугоплавкого металла (например, вольфрама), накаливаемая электрическим током, а анодом – небольшой металлический цилиндр. При подаче напряжения на катода под действием термоэлектронной эмиссии с катода начнут исходить электроны, которые в свою очередь будут приниматься анодом.

Применение электронных ламп резко повысило вычислительные возможности ЭВМ, что способствовало быстрому переходу от первых автоматических релейных вычислительных машин к ламповым ЭВМ первого поколения.

Однако, не обошлось без проблем. Использование электронных ламп омрачала их низкая надежность, высокое энергопотребление и большие габариты. Первые ЭВМ были поистине гигантских размеров и занимали несколько комнат в научно-исследовательских институтах. Обслуживание таких ЭВМ было крайне сложным и трудоемким,



Принцип работы электронной лампы

постоянно выходили из строя лампы, происходили сбои при вводе данных, и возникало множество других проблем. Не менее сложными и дорогостоящими приходилось делать и системы питания (нужно было прокладывать специальные силовые шины для обеспечения питания ЭВМ и делать сложную разводку, чтобы подвести кабели ко всем элементам), и системы охлаждения (лампы сильно грелись, от чего еще чаще выходили из строя).

Несмотря на это, конструкция ЭВМ быстро развивалась, скорость вычисления достигала нескольких тысяч операций в секунду, емкость ОЗУ – порядка 2048 машинных слов. В ЭВМ первого поколения программа уже хранилась в памяти, и использовалась параллельная обработка разрядов машинных слов.

Создаваемые ЭВМ, в основном, были универсальными и использовались для решения научно-технических задач. Со временем производство ЭВМ становится серийным, и они начинают использоваться в коммерческих целях.

В этот же период происходит становление архитектуры Фон-неймановского типа, и многие постулаты, нашедшие свое применение в ЭВМ первого поколения, остаются популярными и по сей день.

Основные критерии разработки ЭВМ, сформулированные Фон-Нейманом в 1946 году, перечислены ниже:

1. ЭВМ должны работать в двоичной системе счисления;

2. все действия, выполняемые ЭВМ, должны быть представлены в виде программы, состоящей из последовательного набора команд. Каждая команда должна содержать код операции, адреса операндов и набор служебных признаков;

3. команды должны храниться в памяти ЭВМ в двоичном коде, так как это позволяет:

   а) сохранять промежуточные результаты вычислений, константы и другие числа в том же запоминающем устройстве, где размещается программа;

   б) двоичная запись команд позволяет производить операции над величинами, которыми они закодированы;

   в) появляется возможность передачи управления на различные участки программы, в зависимости от результатов вычислений;

4. память должна иметь иерархичную организацию, так как скорость работы запоминающих устройств значительно отстает от быстродействия логических схем;

5. арифметические операции должны выполняться на основе схем, выполняющих только операции сложения, а создание специальных устройств – нецелесообразно;

6. для увеличения быстродействия необходимо использовать параллельную организацию вычислительного процесса, т.е. операции над словами будут производиться одновременно во всех разрядах слова.

Стоит отметить, что ЭВМ первого поколения создавались не с нуля. В то время уже были наработки в области построения электронных схем, например, в радиолокации и других смежных областях науки и техники. Однако, наиболее серьезные вопросы были связаны с разработкой запоминающих устройств. Ранее они практически не были востребованы, поэтому какого-либо серьезного опыта в их разработки накоплено не было. Следовательно, каждый прорыв в разработке запоминающих устройств приводил к серьезному шагу вперед в конструировании ЭВМ, так как разработка быстродействующей и емкой памяти – это неотъемлемое условие разработки мощной и быстродействующей ЭВМ.

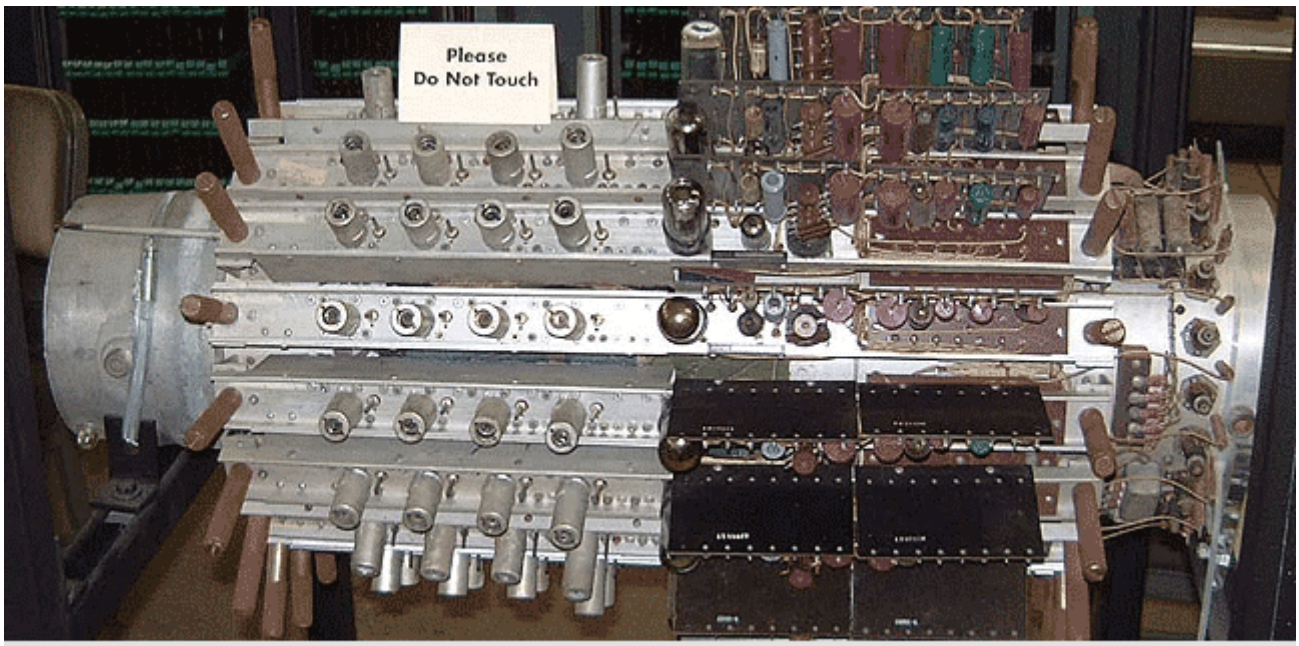
Первые ЭВМ использовали в качестве запоминающего устройства – статические триггеры на ламповых триодах. Однако, получить запоминающее устройство на электронных лампах приемлемой емкости требовало неимоверных затрат. Для запоминания одного двоичного разряда требовалось два триода, при этом для сохранения информации они должны были непрерывно потреблять энергию. Это, в свою очередь, приводило к серьезным выделениям тепла и катастрофическому снижению надежности. В результате, запоминающее устройство было крайне громоздким, дорогим и ненадежным.

В 1944 году начал разрабатываться новый тип запоминающих устройств, основанный на использовании ультразвуковых ртутных линий задержки. Идея была заимствована из устройства уменьшения помех от неподвижных предметов и земли, разработанного для радаров во время Второй Мировой Войны.

Чтобы убрать неподвижные объекты с экрана радара отражённый сигнал разделяли на два, один из которых посылался непосредственно на экран радара, а второй задерживался. При одновременном выводе на экран нормального и запаздывающего сигналов любое появлявшееся из-за задержки и обратной полярности совпадение стиралось, оставляя только подвижные объекты.

Задержка сигнала осуществлялась с помощью линий задержки - наполненных ртутью трубок с пьезокристаллическим преобразователем на концах. Сигналы от радарного усилителя посылались на пьезокристалл в одном конце трубки, и тот, получая импульс, генерировал небольшое колебание ртути. Колебание быстро передавалось на другой конец трубки, где другой пьезокристалл его инвертировал и передавал на экран.

Ртуть использовалась, потому что её удельное акустическое сопротивление почти равно акустическому сопротивлению пьезокристаллов. Это минимизировало энергетические потери, происходящие при передаче сигнала от кристалла к ртути и обратно.



Память на ртутных линиях задержки. UNIVAC I (1951 год)

Для использования в качестве памяти, ртутные линии задержки были несколько доработаны. На принимающем конце трубки был установлен повторитель, который посылал входной сигнал обратно на вход линии задержки, таким образом, импульс, посланный в систему хранения данных, продолжал циркулировать в линии задержки, а, следовательно, сохранялся бит информации до тех пор, пока было электропитание.

Каждая линия задержки сохраняла не один импульс (бит данных), а целый набор импульсов, количество которых определялось скоростью прохождения импульса через ртутную линию задержки (1450 м/с), длительностью импульсов, интервалом между ними и длинной трубки.

Впервые такое устройство хранения данных было использовано в английской ЭВМ – ЭДСАК, вышедшей в свет в 1949 году.

Память на ртутных линиях задержки была огромным шагом вперед, по сравнению с памятью на ламповых триодах, и привела к скачку в развитии вычислительной техники. Однако, она обладала рядом серьезных недостатков:

1. линии задержки требовали строгой синхронизации с устройством считывания данных. Импульсы должны были поступать на приёмник именно в тот момент, когда компьютер был готов считать их;

2. для минимизации энергетических потерь, происходящих при передаче сигнала в линии задержки, ртуть надо содержать при температуре в 40°C, так как при этой температуре ртути удается достигнуть максимального согласования акустических сопротивлений ртути и пьезокристаллов. Это тяжелая и некомфортная работа;

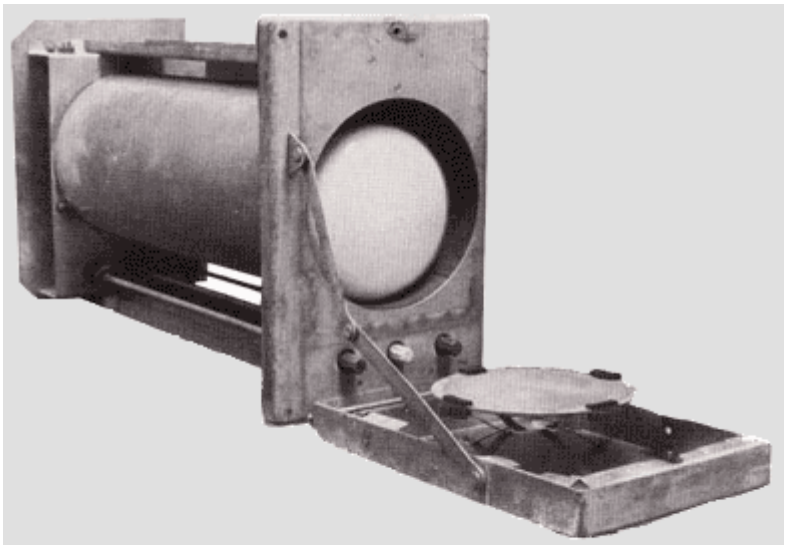
3. изменение температуры ртути также приводило к уменьшению скорости прохождения звука. Приходилось поддерживать температуру в строго заданных рамках, либо регулировать тактовую частоту компьютера, подстраиваясь под скорость распространения звука в ртути при текущей температуре;

4. сигнал мог отражаться от стенок и концов трубки. Приходилось применять серьезные методы для устранения отражений и тщательно настраивать положение пьезокристаллов;

5. скорость работы памяти на ртутных линиях задержки была невелика и ограничивалась скоростью звука в ртути. В результате, она была слишком медленной и значительно отставала от вычислительных возможностей ЭВМ, что сдерживало их развитие. В результате, скорость ЭВМ с памятью на ультразвуковых ртутных линиях задержки составляла несколько тысяч операций в секунду;

6. ртуть – чрезвычайно токсичный и дорогой материал, применение которого связано с необходимостью соблюдения жестких норм безопасности.

Поэтому требовалась новая, более быстрая память для продолжения развития ЭВМ. Вскоре, после создания первой ЭВМ на ультразвуковых ртутных линиях задержки, начались работы по исследованию нового типа памяти, использующего электронно-лучевые трубки, представляющие собой модификацию осциллографических трубок.



Запоминающая электронно-лучевая трубка Фредерика Уильямса

Впервые, способ хранения данных с помощью электронно-лучевых трубок был разработан в 1946 году Фредериком Уильямсом. Изобретение Уильямсона могло сохранять всего один бит и работало следующим образом.

С помощью электронно-лучевой трубки пучок электронов фокусировался на участке пластины, покрытой специальным веществом. В результате, этот участок под действием вторичной эмиссии испускал электроны и приобретал положительный заряд, который сохранялся доли секунды, даже после отключения луча. Если через короткие интервалы времени повторять бомбардировку электронами, то заряд участка можно сохранять столько, сколько потребуется.

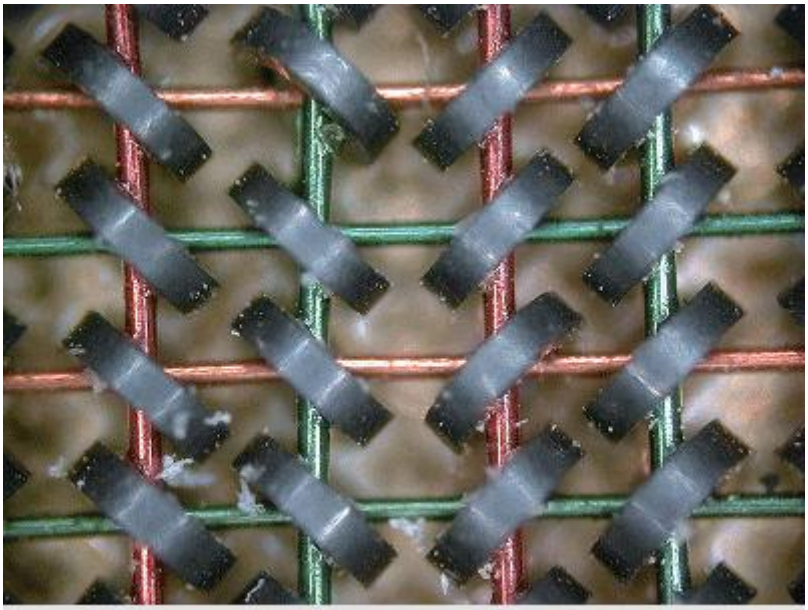
Если же луч, не отключая, чуть передвинуть на соседний участок, то электроны, испущенные соседним участком, будут поглощены первым участком, и он примет нейтральный заряд.

Таким образом, в ячейку, состоящую из двух смежных участков, можно быстро записывать 1 бит информации. Ячейка без заряда – 1, ячейка с положительным зарядом – 0.

Для считывания сохраненного бита информации, с противоположной стороны пластины прикреплялись электроды, измеряющие величину изменения заряда ячейки,

а сама ячейка подвергалась повторному воздействию лучом электронов. В результате, независимо от первоначального состояния, она получала положительный заряд. Если ячейка уже имела положительный заряд, то изменение ее заряда было меньше, чем, если бы она имела нейтральный заряд. Анализируя величину изменения заряда, определяли значение сохраненного в этой ячейке бита.

Однако, процесс считывания данных уничтожал информацию, сохраненную в ячейке, поэтому после операции чтения приходилось повторно записывать данные. В этом процесс работы с памятью на электронно-лучевых трубках был очень похож на работу с современной динамической памятью.



Память на магнитных сердечниках

Первый компьютер с такой памятью появился летом 1948 года и позволял сохранять до тридцати двух тридцати двух разрядных двоичных слов.

Со временем память на электронно-лучевых трубках была заменена памятью с магнитными сердечниками. Этот тип памяти был разработан Дж. Форрестером и У. Папяном, и введен в эксплуатацию в 1953 году.

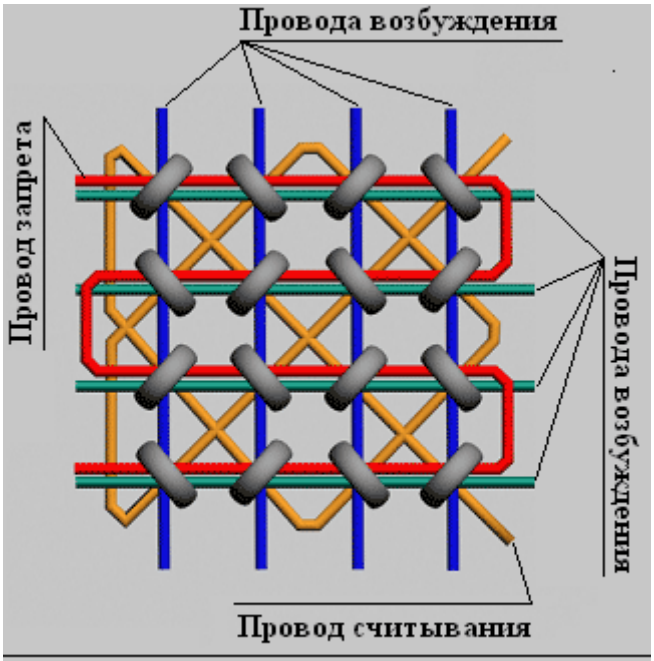
Память на магнитных сердечниках хранила данные в виде направления намагниченности небольших ферритовых колец. Каждое кольцо сохраняло 1 бит информации, а вся память представляла собой прямоугольную матрицу.

В простейшем случае устройство памяти было следующим.

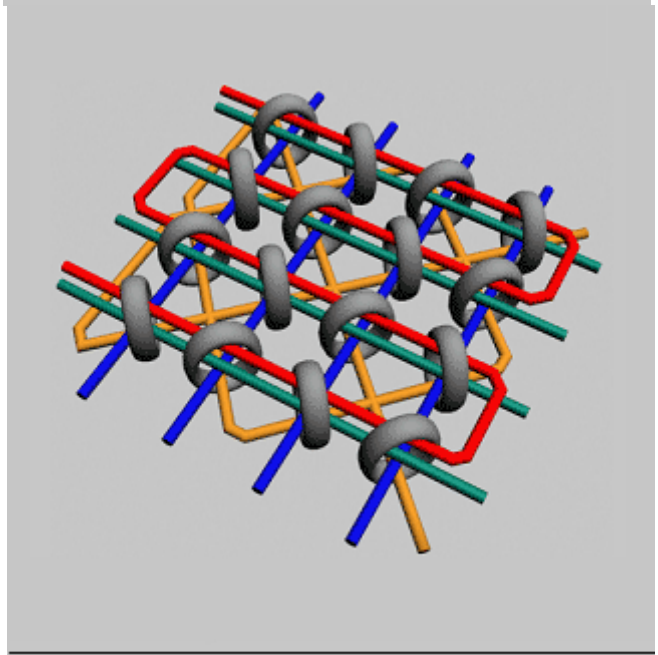
Вдоль строк матрицы через кольца пропускались провода возбуждения (на рисунке они выделены зеленым цветом). Аналогичные провода пропускались через кольца вдоль столбцов матрицы (синий цвет).

Ток, проходящий через эти провода, устанавливал направление намагниченности колец. Причем, сила тока была такова, что один провод не мог изменить направление намагниченности, а, следовательно, направление намагниченности изменялось только в кольце, находящемся на пересечении красного и синего провода. Это было необходимо, так как на каждый провод возбуждения было нанизано несколько десятков ферритовых колец, а изменять состояние нужно было только в одном кольце.

Если в выбранном кольце изменять состояние намагниченности не требовалось, то подавали ток в провод запрета (красный цвет) в направлении, противоположном току в проводах возбуждения. В результате, сумма токов была недостаточной для изменения намагниченности кольца.



Память на магнитных стержнях (16 бит)



Память на магнитных стержнях (16 бит)

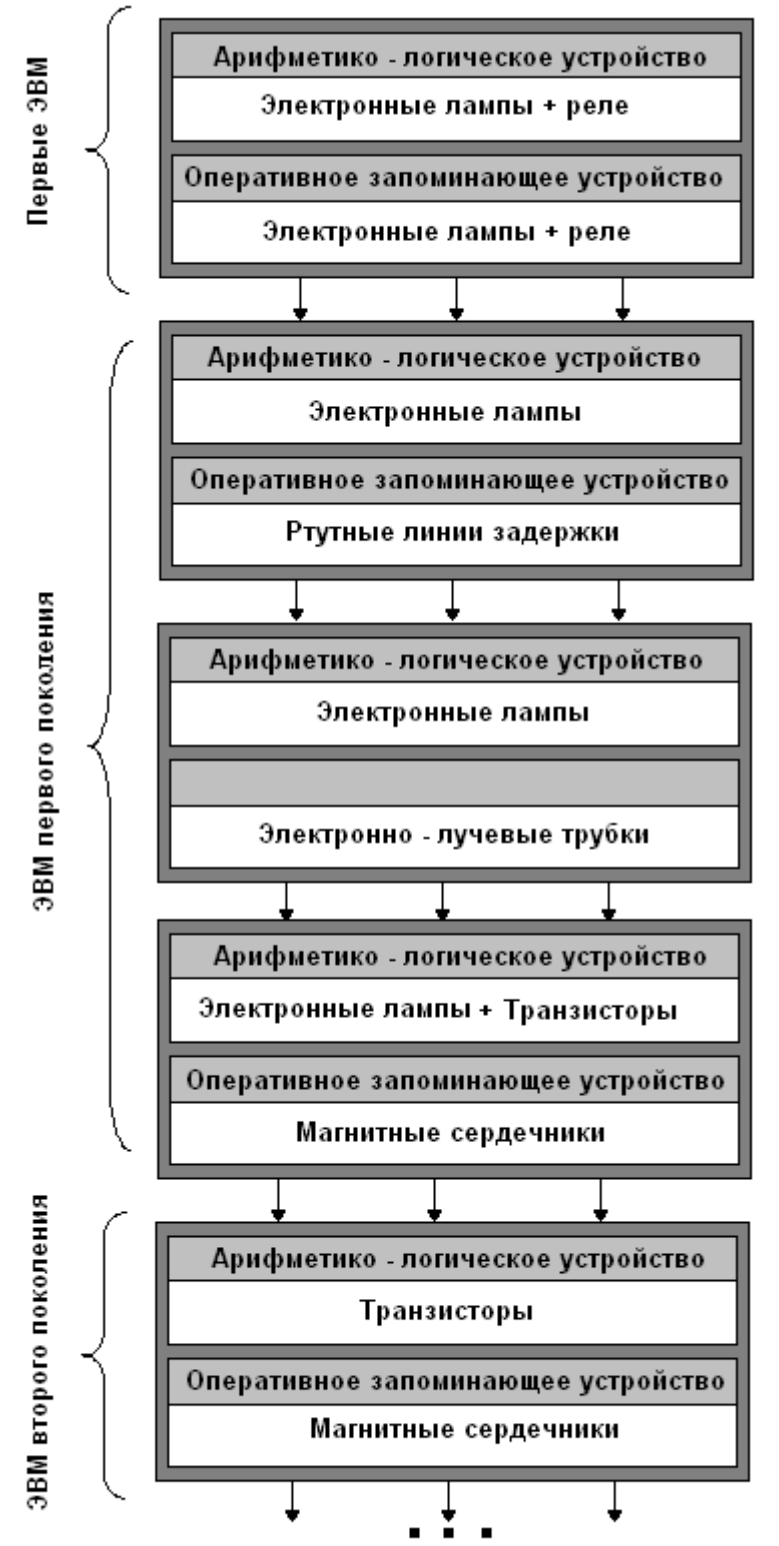
Таким образом, в каждом колечке могли храниться 1 или 0, в зависимости от направления намагниченности.

Для считывания данных с выбранного ферритового кольца, на него по проводам возбуждения подавались такие импульсы тока, что их сумма приводила к намагниченности кольца в определенном направлении, независимо от первоначального намагничивания.

При изменении намагниченности кольца в проводе считывания возникал индукционный ток. Измеряя его, можно было определить, насколько изменилось направление намагниченности в кольце, а, следовательно, узнать хранимое им значение.

Как видите, процесс считывания уничтожал данные (также, как и в современной динамической памяти), поэтому после считывания было необходимо заново записать данные.

Вскоре, этот тип памяти стал доминирующим, вытеснив электронно-лучевые трубки и ультразвуковые ртутные линии задержки. Это дало еще один скачок в производительности ЭВМ.



Тенденции развития ЭВМ первого поколения

Дальнейшее развитие и совершенствование ЭВМ позволило им прочно занять свою нишу в области науки и техники.

К числу передовых ЭВМ первого поколения можно отнести:

ENIAC - первый широкомасштабный электронный цифровой компьютер, созданный в 1946 году по заказу армии США в лаборатории баллистических исследований для расчётов таблиц стрельбы. В эксплуатацию введен 14 февраля 1946 года;

EDVAC — одна из первых электронных вычислительных машин, разработанная в лаборатории баллистических исследований армии США, представленная публике в 1949 году;

EDSAC — электронная вычислительная машина, созданная в 1949 году в Кембриджском Университете (Великобритания) группой во главе с Морисом Уилксом;

UNIVAC - универсальный автоматический компьютер, созданный в 1951 году Д.

Моучли и Дж. Преспер Эккерт;

IAS - ЭВМ Института Перспективных Исследований, разработанная под руководством Дж. Неймана в 1952 году;

Whirlwind – ЭВМ, созданная в Массачусетском Технологическом Университете в марте 1951 года;

МЭСМ - Малая Электронная Счетная Машина – первая отечественная ЭВМ, созданная в 1950 году С.А. Лебедевым;

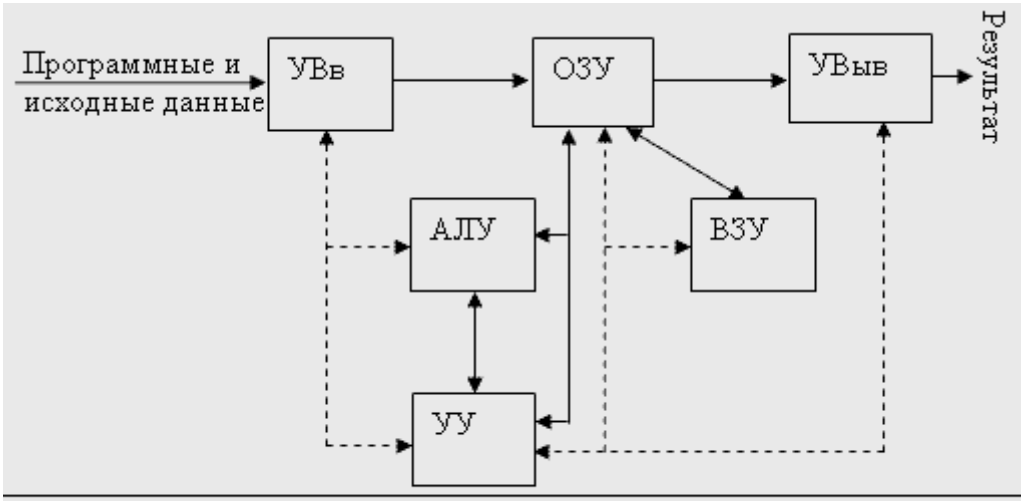
БЭСМ - Большая Электронная Счетная Машина, разработанная Институтом Точной Механики и Вычислительной Техники Академии наук СССР.

Все эти и многие другие вычислительные машины первого поколения подготовили надежную стартовую площадку для победного марша ЭВМ по всему миру.

Стоит отметить, что не было резкого перехода от ЭВМ первого поколения на электронных лампах к ЭВМ второго поколения на транзисторах. Электронные лампы постепенно заменялись, вытесняясь твердотельными транзисторами. В первую очередь, были вытеснены электронные лампы из устройств хранения данных, а затем постепенно они вытеснялись из арифметико-логических устройств.

Слева, схематично изображен переход от чисто ламповых ЭВМ к ЭВМ второго поколения.

За время существования ламповых ЭВМ их структура, изображенная на рисунке ниже, не претерпела серьезных изменений. Переход ко второму поколению ЭВМ также не внес существенных изменений в их структурное построение. В основном, изменилась только элементная база. Серьезные изменения структуры построения ЭВМ начались ближе к третьему поколению ЭВМ, когда начали появляться первые интегральные схемы.



Структурная схема ЭВМ первого поколения

С помощью устройства ввода данных (УВв), в ЭВМ вводились программы и исходные данные к ним. Введенная информация целиком или полностью запоминалась в оперативном запоминающим устройстве (ОЗУ). Затем, при необходимости, она

заносилась во внешнее запоминающее устройство (ВЗУ), откуда по мере надобности могла подгружаться в ОЗУ.

После ввода данных или считывания их из ВЗУ, программная информация, команда за командой, считывалась из ОЗУ и передавалась в устройство управления (УУ).

Устройство управления дешифрировало команду, определяло адреса операндов и номер следующей команды, которую нужно было считать из ОЗУ. Затем, путем принудительной координации всех элементов ЭВМ, УУ организовывало исполнение команды и запрашивало следующую. Цепи сигналов управления показаны на рисунке штриховыми линиями.

Арифметико-логическое устройство (АЛУ) выполняло арифметические и логические операции над данными. Основной частью АЛУ является вычислительное ядро, в состав которого входят сумматоры, счетчики, регистры, логические преобразователи и др.

Промежуточные результаты, полученные после выполнения отдельных команд, сохранялись в ОЗУ. Результаты, полученные после выполнения всей программы вычисления, передавались на устройство вывода (УВыв). В качестве УВыв использовались: экран дисплея, принтер, графопостроитель и т.д.

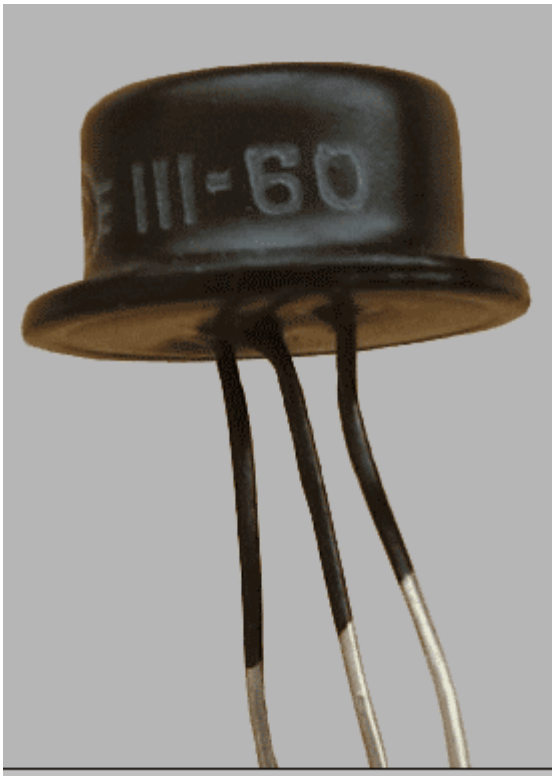
Как видно из приведенной выше структурной схемы, ЭВМ первого поколения имели сильную централизацию. Устройство управления отвечало не только за выполнение команд, но и контролировало работу устройств ввода и вывода данных, пересылку данных между запоминающими устройствами и другие функции ЭВМ. Также были жестко стандартизированы форматы команд, данных и циклов выполнения операций.

Все это позволяло несколько упростить аппаратуру ЭВМ, ужасно сложную, громоздкую и без изысков организации вычислительного процесса, но значительно сдерживало рост их производительности.

Первая ЭВМ на электронных лампах была создана в США и называлась ЭНИАК. Она оказала существенное влияние на направление развития вычислительной техники. Вскоре, за примером США последовали и многие другие промышленно-развитые страны (Великобритания, Швейцария, СССР и др.), уделявшие развитию вычислительной техники в послевоенный период много внимания.

Однако, наибольшее значение в развитии вычислительной техники оказали исследования, проводимые в США, СССР и Великобритании. В других же странах, например во Франции, ФРГ, Японии, ЭВМ, относящиеся к первому поколению, не получили серьезного развития. В частности, для ФРГ, Испании и Японии даже трудно отделить рамки перехода от ЭВМ первого поколения к ЭВМ второго поколения, так как, наряду с первыми ламповыми ЭВМ, в конце пятидесятых годов начинали создаваться и первые ЭВМ на полупроводниковой основе.

Второе поколение ЭВМ.



Полупроводниковый триод

Второе поколение ЭВМ создавалось в период с 1955 по 1964 года. На самом деле, четко ограничивать рамки поколений сложно, так как в одно и то же время выпускались ЭВМ, относящиеся к разным поколениям, да и сам переход от поколения к поколению был не резким, а постепенным. Вначале заменялись одни элементы ЭВМ, затем – другие, и так, постепенно, за несколько лет, осуществлялся переход.

Переход на новую элементную базу оказался неизбежным, так как рост производительности и надежность ЭВМ первого поколения достигли своего максимума. Основные причины, приведшие к необходимости замены электронных ламп, были следующими:

1. Нить накаливания в электронных лампах со временем теряет свои эмиссионные свойства и перегорает. В среднем, срок службы лампы не превышал 10 000 часов. Таким образом, в ЭВМ, состоящей из 104 электронных ламп, в среднем, каждый час, выходила из строя одна электронная лампа. Столь низкие показатели надежности были головной болью разработчиков, заставляли применять сложные и дорогостоящие способы повышения надежности, и сильно сдерживали рост производительности ЭВМ. Для сравнения, транзисторы в то время имели срок службы, превосходящий срок службы электронных ламп в тысячи раз.

2. ЭВМ на электронных лампах требуют мощных источников питания, при этом почти 75% энергии растрачивается на тепловых потерях. Это, в свою очередь, приводит к необходимости организации дорогостоящих и сложных систем охлаждения. Транзисторы потребляют на порядок меньше энергии и слабее греются.

3. Большие габариты электронных ламп. Самые миниатюрные радиолампы не позволяли в одном кубическом дециметре разместить более 1000 элементов, в то же время использование транзисторов позволяло на порядок увеличить плотность монтажа.

4. Радиолампы – это хрупкий элемент. Его установка требует осторожности и аккуратности, и с большим трудом поддается автоматизации. В то же время транзисторы - гораздо более надежны и прочны, что позволяет легко автоматизировать процесс их производства и монтажа, а это приводит к снижению себестоимости транзисторов и ЭВМ в целом.

Таким образом, основой ЭВМ второго поколения стало использование новой элементной базы - полупроводниковых транзисторов (триодов), составляющих основную часть конструкции ЭВМ.

Копия транзистора разработанного в лаборатории Белла 23 декабря 1947 года

История создания транзисторов началась еще 22 октября 1925 года, когда Юлием Эдгаром Лилиенфельдом был зарегистрирован патент на принцип работы полевого транзистора. Теория работы полевых транзисторов - проще биполярных, поэтому обоснована и запатентована она была значительно раньше биполярных транзисторов. В общем случае принцип действия полевого транзистора аналогичен работе электронных ламп. Исток в полевом транзисторе подобен катоду вакуумного триода, затвор — сетке, сток — аноду. Однако, трудности в практической реализации полевых транзисторов позволили создать действующую модель лишь в 1960 году, значительно позже создания биполярного транзистора, и только в девяностых годах технология полевых транзисторов стала доминировать над биполярными.

Первый действующий транзистор был биполярным, и создали его в 1947 году ведущие специалисты Уильям Шокли, Джон Бардин и Уолтер Браттейн из фирмы «Bell Labs». Официальная демонстрация устройства состоялась 23 декабря 1947 года, и именно эта дата считается официальным днем изобретения транзистора.

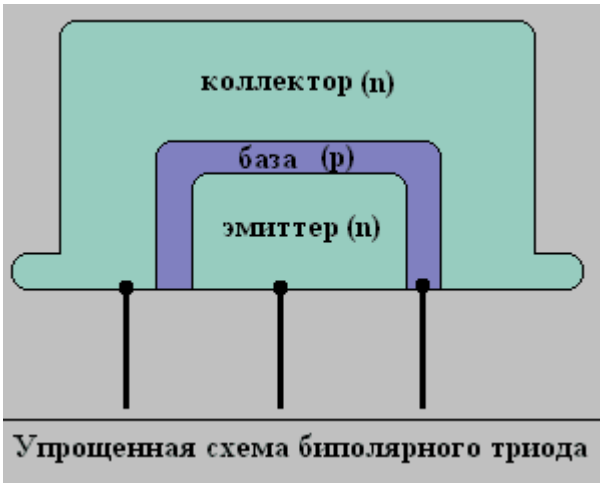
Первый биполярный транзистор представлял собой прибор, в котором два металлических контакта соединялись с бруском из поликристаллического германия. Его копия изображена на фотографии справа.

Таким образом, основой ЭВМ второго поколения стали биполярные транзисторы, представляющие собой три последовательно расположенные слоя полупроводников: эмиттера, базы и коллектора.

Полупроводники — это вещества, удельное сопротивление которых изменяется в зависимости от температуры, наличия примесей или изменением освещенности. При построении транзисторов использовали полупроводники с различными примесными проводимостями.

Примеси бывают двух типов – донорной и акцепторной. При добавлении донорной примеси в полупроводнике образуются «лишние» электроны. Такие полупроводники называются полупроводниками n-типа. Например, для кремния с валентностью n = 4 донорной примесью является мышьяк с валентностью n=5. Каждый атом примеси мышьяка приведет к образованию одного электрона проводимости.

При добавлении акцепторной примеси в полупроводнике образуются «лишние» частицы с положительным зарядом, численно равным заряду электрона. Такие частиц называются дырками, а полупроводники с лишними дырками называются полупроводниками p-тип. Например, для кремния акцепторной примесью является индий с валентностью n = 3. Каждый атом индия приведет к образованию лишней «дырки».



Упрощенная схема биполярного триода

При контакте двух полупроводников различного типа, электроны из полупроводника n-типа начинают переходить в полупроводник p-типа, а дырки из полупроводника p-типа - в полупроводник n-типа. Однако, как только пограничный слой полупроводника n-типа «насытится» дырками, а пограничный слоя полупроводника p-типа насытится электронами, процесс диффузии дырок и электронов прекратится из-за образования, так называемого, запирающего слоя.

Но стоит подать на полупроводник n-типа отрицательное напряжение, а на полупроводник p-типа - положительное, как запирающий слой разрушится, и диффузия дырок и электронов возобновится. Если же на полупроводник n-типа подать положительное напряжение, а на p-типа – отрицательное, то запирающий слой увеличится. То есть, при подаче на коллектор логической единицы (например, напряжение -5 вольт), на эмиттере можем получить либо логический ноль (напряжение меньше 1 вольта), либо логическую единицу (напряжение 5 вольт). Логическую единицу получаем, если на базу подаем положительное напряжение (например, 5 вольт), иначе на эмиттере имеем логический ноль. На основе этих элементов и строились ЭВМ второго поколения.

Как видите, принцип работы полупроводниковых транзисторов не сильно отличается от принципа работы электронных ламп. Однако, их использование позволило значительно усовершенствовать ЭВМ без существенных изменений в структурной схеме. Так производительность ЭВМ выросла примерно на два порядка, а габариты уменьшились на порядок. Значительно (на несколько порядков) повысилась надежность. При этом стоимость ЭВМ снизилась!

Эту ситуацию хорошо иллюстрирует переход от ламповых ЭВМ на полупроводниковые ЭВМ, выполненный фирмой IBM в линейке моделей 709 и 7090. IBM 709 – это ламповая ЭВМ, созданная в августе 1958 года. IBM 7090 – это полупроводниковая ЭВМ, созданная в июне 1960 года, схожая по структуре с IBM 7090. При этом полупроводниковая ЭВМ была более, чем в 6 раз, быстрее своего лампового собрата.

	ЭВМ первого поколения		ЭВМ второго поколения	
	1950г	1955г	1960г	1965г
Время выполнения операции сложения (мкс)	240	15	4	0.8
Объем ОЗУ (бит)	$1.5*10^4$	$1.5*10^5$	$1.5*10^6$	$6*10^6$
Время доступа к ОЗУ (мкс)	282	12	4	0.5
Плотность монтажа элементов ($эл/м^3$)	160	$2.9*10^3$	$2.9*10^4$	$4.4*10^4$

Для сравнения, в таблице слева приведены усредненные данные по производительности и габаритам для ЭВМ первого и второго поколения. Данные взяты из книги «Развитие вычислительных машин», авторы Апокин И.А., Мейстров Л.Е.

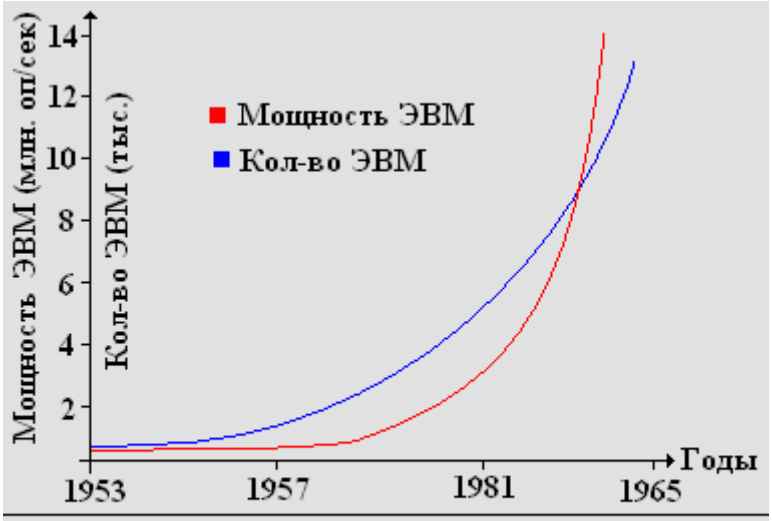
Стоит отметить, что замена электронных ламп на новые элементы шла не только в одном направлении (использование транзисторов). Были предприняты и другие способы усовершенствования ЭВМ. Так в Японии в 1958 году серийно выпускались ЭВМ на параметронах.

Параметрон – это электронный элемент, принцип действия которого основан на особенностях параметрического возбуждения и усиления электрических колебаний. Как описано в большой советской энциклопедии, простейший параметрон представляет собой колебательный контур, настроенный на частоту f0. При периодическом изменении под воздействием сигнала накачки с частотой fn, равной примерно 2*f0, одного из энергоёмких параметров контура, в нём возникает колебание с частотой fm = fn/2, когерентное по отношению к возбуждающему колебанию. При этом фаза возбуждённых в параметроне колебаний может принимать одно из двух отличающихся на 180° значений, условно обозначаемых (0, p), и сколь угодно долго находиться в этом состоянии. Именно эта способность параметрона и позволяет использовать его в качестве основы для построения ЭВМ.

Также были выпущены ЭВМ (в СССР – Сетунь, а во Франции - КАБ-500), использующие вместо электронных ламп магнитные элементы (ферритовые сердечники) в качестве логических элементов и запоминающих устройств.

Однако, эти направления развития ЭВМ не выдержали конкуренции с транзисторами, так как транзисторы были более технологическими, легче подвергались миниатюризации и позволяли использовать технологии интегральных схем.

Существенный рост производительности и повышение надежности, снижение массы, габаритов и потребляемой мощности значительно повысили спрос на ЭВМ и расширили область их применения. Появились предпосылки для использования ЭВМ в авиации, космонавтике, машиностроении и других быстро развивающихся областях науки и техники.



Рост количества ЭВМ и их суммарной производительности в США

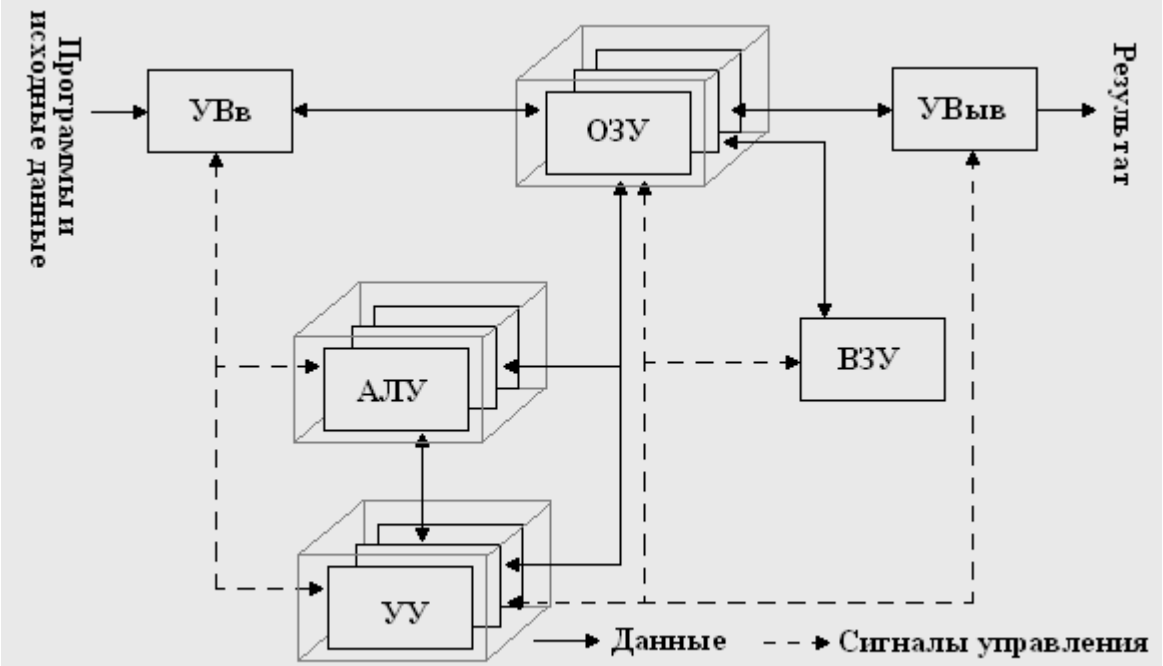
Наметились явные тенденции к значительному росту парка ЭВМ и их мощностей. На графике справа приведены тенденции развития парка ЭВМ для США по данным, приведенным в книге «Развитие вычислительных машин», авторы Апокин И.А., Мейстров Л.Е.

При этом основные тенденции развития ЭВМ были связаны с совершенствованием элементной базы, поэтому структурная схема ЭВМ, изображенная на рисунке ниже, не претерпела кардинальных изменений, по сравнению со структурной схемой ЭВМ первого поколения. Однако, наметились тенденции к распараллеливанию вычислительных ресурсов ЭВМ и многопрограммному принципу работы.

ЭВМ, зачастую, содержали несколько параллельно работающих устройств управления, несколько оперативных запоминающих устройств и даже несколько арифметико-логических блоков. Причем часто устройства, выполняющие одну и ту же функцию, могли быть, как однотипные, так и специализированные. Например, могло быть одно центральное арифметико-логическое устройство и несколько вспомогательных устройств, оптимизированных для решения специфических задач.

Так в ЭВМ «Ханиуэлл-800», разработанной в США в 1960 году, использовалось несколько параллельно работающих оперативных запоминающих устройств, подчиняющихся одному устройству управления. Это позволяло значительно компенсировать медленную работу схем памяти на магнитных сердечниках и более эффективно использовать потенциал логических схем. А в ЭВМ «Гамма-60», созданной во Франции в 1960 году, было несколько устройств управления, работающих с одним блоком оперативной памяти. Подобная структурная схема выгодна при сложной и длительной обработке данных, сравнительно небольших объемов. ЭВМ RW-400, разработанная в США в 1960 году фирмой «Рамо Вулдридж», была снабжена несколькими независимыми блоками оперативной памяти и несколькими устройствами управления. Такая структурная схема в наибольшей степени соответствовала принципам параллельной работы и позволяла значительно повысить производительность ЭВМ.

Структурная схема ЭВМ второго поколения, отражающая тенденции развития вычислительной техники, изображена на рисунке снизу.



Структурная схема ЭВМ второго поколения

На схеме:

УВв – устройство ввода;
УВыв – устройство вывода;
ОЗУ – одно или несколько оперативных запоминающих устройств;
АЛУ - одно или несколько арифметико-логических устройств;
УУ - одно или несколько устройств управления;
ВЗУ – внешнее запоминающее устройство.

Изменение структуры ЭВМ в сторону использования различных принципов параллелизма привело к созданию ряда требований, предъявляемых к многопрограммным ЭВМ, верно сформулированных Б.Л. Райли в книге «Communs ACM»:

1. Программы, вводимые в ЭВМ или сохраненные в ПЗУ, должны быть независимы от абсолютных машинных адресов.

2. Должна иметься система приоритетов программ, с помощью которой можно с минимальной задержкой выбирать соответствующую программу, когда появляется возможность выбора между несколькими программами.

3. Должна быть предусмотрена система, которая сохраняла бы текущее состояние каждой исполняемой программы.

4. Любой регистр или любой другой элемент системы, не используемый в данный момент времени, должен быть доступен для любой другой параллельно выполняемой программы.

5. Должна быть обеспечена система прерываний выполняемой программы методом опроса (устройство управления переключается в соответствии с состоянием опрашиваемых устройств) или методом приостановки (сигналы из

других устройств поступают в устройство управления и вызывают соответствующую передачу управления другой программе).

6. Должны существовать прямые связи между двумя любыми устройствами системы, которые могут обмениваться информацией. Не следует использовать некое третье устройство в качестве промежуточного элемента при обмене.

7. Система должна быть организована таким образом, чтобы осуществление наблюдения и управления, необходимых для выполнения нескольких программ, не требовало бы совсем или требовало бы минимум дополнительного времени.

8. Объем преобразования и пересылок данных внутри системы должен быть сведен к минимуму.

Усложнение структуры ЭВМ второго поколения, возможность распараллеливания задач, идеи мультипрограммирования, расширение области применения сделали процесс программирования сложной, трудоемкой и востребованной работой. Требовались инструменты для облегчения этой задачи и уменьшения времени разработки программ. Поэтому стали бурно развиваться алгоритмические языки программирования. К концу шестидесятых годов их насчитывалось уже более 1000. Среди них наиболее известными были:

Алгол, разработанный в 1957 году и ориентированный на научно-технические расчеты;

Фортран, разработанный специалистами фирмы IBM 1957 году для задач численного анализа. Этот язык программирования широко используется и по сей день;

Кобол, разработанный в США в 1958 году, ориентированный на решение экономических задач;

Лисп, разработанный в 1958 году в США и ориентированный на символьную обработку данных, и процессы принятия решений. На данный момент широко используется;

ИПЛ, разработанный в США в Массачусетском Технологическом Институте в 1960 году. Позволял манипулировать словами и выражениями на естественном языке. В этом языке впервые появилось понятие списка;

ПЛ-1, разработанный фирмой IBM в 1960 году. Универсальный язык программирования.

Широкое развитие языков программирования еще больше способствовало популярности ЭВМ и их внедрению во все новые и новые области применения. Перечислим наиболее значимые разработки в области вычислительной техники, относящиеся ко второму поколению ЭВМ:

ТХ-0 - первый экспериментальный компьютер на транзисторах, разработанный в 1953 году в Массачусетском Технологическом Институте (в 1955 году был введен в эксплуатацию).

TRADIC – одна из первых транзисторных ЭВМ, созданная в США в 1955 году. В ее состав входило 800 транзисторов и 11 000 германиевых диодов.

Stretch (IBM-7030), разработанная в 1960 годах в США фирмой IBM, оказала сильнейшее влияние на развитие вычислительной техники. В этой ЭВМ были собраны практически все известные на 1960 год достижения в области вычислительной техники. Широкое использование принципов параллельной работы, большой набор команд (свыше 600), огромное количество высококачественных элементов (169000 транзисторов) позволили достичь небывалой производительности. Так операция сложения 64-разрядных чисел с плавающей запятой выполнялась за 1,5 мкс, а операция умножения – за 2,7 мкс. Всего было выпущено 5 экземпляров этой машины.

FX1, рразработана Линкольновской лабораторией технологического института в апреле 1961 года. Основной целью разработки было достижение максимальных вычислительных возможностей, для чего использовались самые передовые достижения в технологии. Например, впервые, в качестве основного запоминающего устройства была использована память на магнитных пленках.

CDC 6600 – ЭВМ, разработанная фирмой Control Data в 1960 году по заказу комиссии по атомной энергетике США. В этой ЭВМ широко использовались принципы параллельной обработки данных, для которой предназначался центральный процессор с запоминающим устройством на 131 тысячу слов и десять периферийных вычислителей, каждый из которых был снабжен своей памятью на 4096 слов. До выпуска первых ЭВМ на интегральных схемах (1065 год) CCD-6600 оставалась самой быстродействующей ЭВМ в мире. Ее производительность превышала три миллиона операций в секунду.

Раздан 2, созданная в СССР в 1961 году. ЭВМ предназначалась для научно-технических и инженерных расчетов. Производительность этой ЭВМ составляла примерно 5000 операций в секунду. Оперативное запоминающее устройство было выполнено на ферритовых сердечниках, внешнее запоминающее устройство – накопитель на магнитной ленте.

Минск-2 – ЭВМ, разработанная Минским заводом вычислительной техники им. Серго Орджоникидзе в 1963 году. Она предназначалась для решения научно-технических и планово-экономических задач.

МИР - малая электронная цифровая вычислительная машина, разработанная в Институте Кибернетики АН УССР под руководством В. М. Глушкова в 1965 году.

БЭСМ-6 – ЭВМ, созданная в 1966 году в СССР на элементной базе второго поколения. В ее состав входило 60 000 транзисторов и 200 000 полупроводниковых диодов, а производительность достигала 1 миллиона операций в секунду.

Список можно продолжать еще очень долго, и все это говорит о том, что ЭВМ второго поколения показали, что будущее человечества тесно связано с развитием и использованием вычислительной техники. С этого момента ЭВМ стали неотъемлемой частью жизни человечества.

Основные законы логики :

$A = A$	– закон тождества
$A \& \overline{A} = 0$	– закон непротиворечия
$A \vee \overline{A} = 1$	– закон исключенного третьего
$\overline{\overline{A}} = A$	– закон двойного отрицания

Свойства констант:

$\overline{0} = 1$ $\quad \overline{1} = 0$

$A \vee 0 = A$ $\quad A \& 0 = 0$

$A \vee 1 = 1$ $\quad A \& 1 = 1$

Законы идемпотентности:

$A \vee A = A$

$A \& A = A$

Законы коммутативности:

$A \vee B = B \vee A$

$A \& B = B \& A$

Законы ассоциативности:

$A \vee (B \vee C) = (A \vee B) \vee C$

$A \& (B \& C) = (A \& B) \& C$

Законы дистрибутивности:

$A \vee (B \& C) = (A \vee B) \& (A \vee C)$

$A \& (B \vee C) = (A \& B) \vee (A \& C)$

Законы поглощения:

$A \vee (A \& B) = A$

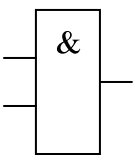
$A \& (A \vee B) = A$

Законы де Моргана:

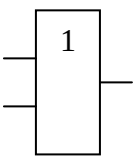
$\overline{A \vee B} = \overline{A} \& \overline{B}$

$\overline{A \& B} = \overline{A} \vee \overline{B}$

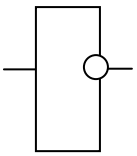
Базовые логические элементы компьютера



Логический элемент **И**
конъюнктор



Логический элемент **ИЛИ**
дизъюнктор



Логический элемент **НЕ**
инвертор

Лабораторная работа

Задание 1. Ветвление с двумя альтернативами

Задание: Составить схему алгоритма и программу на языке Паскаль для вычисления значений функции y = f(x) при произвольных значениях x. Получить результат работы программы для двух заданных значениий **x**. Варианты заданий в таблице 2.1.

Варианты заданий на ветвление с двумя альтернативами

Номер вар.	Y=f(x)	Исходные данные
1	$y=\begin{cases} b+2\ln\lvert x\rvert & при\ x\le 3, \\ \dfrac{x^2}{x^2+a} & при\ x>3 \end{cases}$	$a=10{,}2$ $b=13{,}4$ $1) x=4{,}5$ $2) x=1{,}72$
2	$y=\begin{cases} a+\dfrac{1}{2}e^{-x} & при\ x>0, \\ \cos(bx+1) & при\ x\le 0 \end{cases}$	$a=8{,}53$ $b=17{,}1$ $1) x=2{,}5$ $2) x=-3{,}1$
3	$y=\begin{cases} \dfrac{1}{a^2+x^2} & при\ x\le 1, \\ b\cdot\ln\lvert x\rvert & при\ x>1 \end{cases}$	$a=7{,}2$ $b=5{,}7$ $1) x=2{,}92$ $2) x=-3{,}57$
4	$y=\begin{cases} \dfrac{a+x^2}{b+\ln(\lvert x\rvert+1)} & при\ x\le 2, \\ e^x+x^2 & при\ x>2 \end{cases}$	$a=9{,}1$ $b=3{,}6$ $1) x=5{,}41$ $2) x=0{,}71$
5	$y=\begin{cases} a\sin^2 x+\sqrt{x} & при\ x\le 1, \\ be^{x^2} & при\ x>1 \end{cases}$	$a=1{,}1$ $b=3{,}2$ $1) x=4{,}23$ $2) x=0{,}93$
6	$y=\begin{cases} a\cdot tg(x^2) & при\ x\le -1, \\ b+\dfrac{x^2}{x^2+a} & при\ x>-1 \end{cases}$	$a=9{,}5$ $b=3{,}8$ $1) x=-4{,}52$ $2) x=1{,}83$
7	$y=\begin{cases} (a+x)arctg(ax) & при\ x>3, \\ \cos^2(b+x^3) & при\ x\le 3 \end{cases}$	$a=4{,}1$ $b=2{,}9$ $1) x=6{,}81$ $2) x=2{,}17$

8	$y=\begin{cases}\sin^3(a+x) & при\ x<5,\\ \ln\sqrt{\lvert b-x\rvert} & при\ x\ge 5\end{cases}$	$a=1{,}9$ $b=3{,}4$ 1) $x=7{,}39$ 2) $x=0{,}62$
9	$y=\begin{cases}\sqrt{1+x\sqrt{ax}} & при\ x\ge 2,\\ \sin(bx)+3 & при\ x<2\end{cases}$	$a=4{,}6$ $b=3{,}2$ 1) $x=3{,}78$ 2) $x=1{,}54$
10	$y=\begin{cases}\sqrt{e^{2x-b}}-1 & при\ x\le 0,\\ \dfrac{1}{x^2+a} & при\ x>0\end{cases}$	$a=6{,}7$ $b=1{,}8$ 1) $x=-0{,}24$ 2) $x=2{,}13$
11	$y=\begin{cases}\sqrt{a+\lvert\sin x\rvert} & при\ x>4,\\ tg(bx) & при\ x\le 4\end{cases}$	$a=3{,}9$ $b=4{,}8$ 1) $x=5{,}17$ 2) $x=-2{,}35$
12	$y=\begin{cases}2x^2+a\cos(bx) & при\ x\le 1,\\ e^x+tg\ x^3 & при\ x>1\end{cases}$	$a=1{,}71$ $b=0{,}83$ 1) $x=-2{,}16$ 2) $x=3{,}37$
13	$y=\begin{cases}\ln(a+x^2) & при\ x\ge 2,\\ e^{\sin x}+2b & при\ x<2\end{cases}$	$a=5{,}9$ $b=6{,}1$ 1) $x=6{,}72$ 2) $x=1{,}23$
14	$y=\begin{cases}0{,}2x^3+a & при\ x>-1,\\ bx^2+\ln\lvert x+3\rvert & при\ x\le -1\end{cases}$	$a=2{,}9$ $b=1{,}6$ 1) $x=3{,}18$ 2) $x=-1{,}17$
15	$y=\begin{cases}\sin(x+a^2) & при\ x<2,\\ \ln(x^2+2x+b) & при\ x\ge 2\end{cases}$	$a=1{,}39$ $b=2{,}76$ 1) $x=3{,}68$ 2) $x=0{,}91$
16	$y=\begin{cases}a-b^2x & при\ x\le -3,\\ \dfrac{1}{x^2+e^{bx}} & при\ x>-3\end{cases}$	$a=7{,}5$ $b=1{,}4$ 1) $x=-4{,}13$ 2) $x=0{,}77$
17	$y=\begin{cases}\sqrt{\lvert\sin ax\rvert} & при\ x<-1,\\ \ln\sqrt{1+(bx)^2} & при\ x\ge -1\end{cases}$	$a=1{,}57$ $b=2{,}38$ 1) $x=-0{,}1$ 2) $x=-4{,}25$
18	$y=\begin{cases}\sqrt{(a+x)^3} & при\ x\ge 1,\\ e^{bx-2} & при\ x<1\end{cases}$	$a=4{,}92$ $b=5{,}18$ 1) $x=5{,}13$ 2) $x=-1{,}32$

19	$y=\begin{cases}\sqrt{2\lvert x\rvert+\cos^2 x} & при\ x\le 6,\\ b\sin^3(ax) & при\ x>6\end{cases}$	$a=4{,}49$ $b=5{,}18$ 1)$x=4{,}41$ 2)$x=7{,}69$
20	$y=\begin{cases}\sqrt{2+\lvert x\rvert}+\cos(b+x) & при\ x\le -3,\\ a\sin(x^2) & при\ x>-3\end{cases}$	$a=1{,}89$ $b=2{,}7$ 1)$x=-2{,}37$ 2)$x=-5{,}72$
21	$y=\begin{cases}\dfrac{1}{\cos^2\left(\dfrac{b^3}{x}\right)} & при\ x\ge 2,\\ \ln(1+ax) & при\ x<2\end{cases}$	$a=1{,}89$ $b=0{,}78$ 1)$x=2{,}63$ 2)$x=-0{,}12$
22	$y=\begin{cases}2+e^{a+\sqrt{x}} & при\ x>0,\\ \sin^3 bx & при\ x\le 0\end{cases}$	$a=4{,}17$ $b=2{,}24$ 1)$x=-1{,}93$ 2)$x=3{,}27$
23	$y=\begin{cases}a+2\sqrt{\sin ax+3} & при\ x\le -1,\\ \ln\sqrt{x^2+b} & при\ x>-1\end{cases}$	$a=1{,}43$ $b=4{,}18$ 1)$x=-3{,}29$ 2)$x=1{,}64$
24	$y=\begin{cases}a+\dfrac{1}{\sqrt{x^2+1}} & при\ x\le -4,\\ \ln(b+\lvert x\rvert) & при\ x>-4\end{cases}$	$a=6{,}18$ $b=3{,}52$ 1)$x=-5{,}22$ 2)$x=2{,}15$
25	$y=\begin{cases}a+be^x & при\ x\le -1,\\ \cos^3(ax)^2 & при\ x>-1\end{cases}$	$a=1{,}83$ $b=2{,}27$ 1)$x=3{,}67$ 2)$x=-0{,}48$
26	$y=\begin{cases}a^x+\sqrt{\lvert x\rvert+\lvert a\rvert} & при\ x\le 0,\\ \dfrac{v^3}{u+v^3/(u+v^3)} & при\ x>0\end{cases}$	a=1,25 u=-0,22 v=0,01 1)x=-0,85 2)x=2,34
27	$y=\begin{cases}\left\lvert x^{\frac{z}{x}}-\sqrt{a\lvert x\rvert}\right\rvert & при\ x\le 2,\\ (a-x)\dfrac{a-z/(a-x)}{1+(a-x)^2} & при\ x>2\end{cases}$	a=18,225 z=-3,298 1)x=1,825 2)x=3,546

28	$y=\begin{cases}\sqrt{10(\sqrt[3]{x}+x^{a+2})} & npu\ x\le 10,\\ (\sin z)^2+\lvert x+a\rvert & npu\ x>10\end{cases}$	a=-2,75 z=0,15 1)x=8,45 2)x=16,55
29	$y=\begin{cases}e^{\lvert a-x\rvert}(tg^2 z+1)^x & npu\ x\le 0,\\ \dfrac{\sqrt[3]{8+\lvert x-a\rvert^2+1}}{x^2+a^2+2} & npu\ x>0\end{cases}$	a=0,750 z=0,845 1)x=-4,500 2)x=2,320
30	$y=\begin{cases}a+\dfrac{x}{a+x^3} & npu\ x\le 1\\ (1+tg^2\dfrac{z}{2})^{\sqrt{\lvert a\rvert+6}} & npu\ \ x>1\end{cases}$	a=-8,750 z=0,765 1)x=0,100 2)x=2,76

Лабораторная работа

Тема: Арифметическое выражение

Варианты заданий

Задание: Составить блок-схему алгоритма и программу на языке Паскаль для вычисления значений функции y=f(x) при заданном значении x, которое вводится с клавиатуры.

Таблица 1.1

Номер варианта	y=f(x)	Исходные данные
1	$y=\frac{\sqrt{cx+62,7e^{x}}}{ax^{2}+7x+b\ln x}$	$a=7,2$ $b=14,3$ $c=13,4$ $x=5,6$
2	$y=\frac{ax+3,8tgx}{\sqrt{bx^{3}+c}}$	$a=1,23$ $b=5,14$ $c=3,97$ $x=7,1$
3	$y=\left(\frac{a}{bx^{2}+1}+cx^{3}+b\sin^{2}x\right)^{2}$	$a=2,27$ $b=1,18$ $c=3,92$ $x=0,78$
4	$y=\left(a\sqrt{4,19x^{3}-1}-\sqrt{b\ln x+c}\right)^{-1}$	$a=9,2$ $b=3,5$ $c=12,3$ $x=3,2$
5	$y=\ln\left\|a\sin x+b\cos\left(x^{2}\right)\right\|$	$a=1,2$ $b=2,3$ $x=5,6$
6	$y=\sqrt{\frac{ax^{3}+arctgx}{cx+b\left\|\ln x\right\|}}$	$a=2,71$ $b=1,63$ $c=0,81$ $x=0,51$
7	$y=\frac{ax}{\sqrt{b^{2}+2e^{x}-bx}}$	$a=6,32$ $b=3,704$ $x=7,15$
8	$y=\cos(ax)+b\ln\left(1+bx+e^{x}\right)$	$a=7,1$ $b=1,8$ $x=0,9$
9	$y=\frac{\sqrt{e^{ax}+x^{2}}\cdot\ln\left(x^{2}+bx+10\right)}{\sin(cx)+4,2}$	$a=5,7$ $b=6,4$ $c=3,1$ $x=2,8$
10	$y=\frac{\sqrt{e^{2x+b}}-1,7\cos(cx)}{\ln\left(x^{2}+a\right)}+x^{3}$	$a=2,1$ $b=5,3$ $c=1,4$ $x=1,2$
11	$y=\frac{\ln\sqrt{x^{2}+b}+cx^{3}}{e^{x}+a}$	$a=4,7$ $b=7,21$ $c=1,72$ $x=0,91$

12	$y = \dfrac{\sin\sqrt{e^x + ax^2 + b\ln x}}{ax^2 + cx + 13{,}7}$	$a = 3{,}7$ $b = 4{,}9$ $c = 2{,}5$ $x = 1{,}3$
13	$y = \sqrt{\dfrac{a}{1+bx^2}} + b\,ctg\ x + e^{cx}$	$a = 4{,}5$ $b = 2{,}2$ $c = -1{,}5$ $x = 0{,}85$
14	$y = \dfrac{(cx)^2 - e^{bx}}{\sqrt{x} + \cos(ax)}$	$a = 4{,}5$ $b = 2{,}2$ $c = 1{,}67$ $x = 2{,}36$
15	$y = \dfrac{\sin\left(x^2 + a^2\right)\cdot e^{b+x}}{\sqrt{ax^3 + c}}$	$a = 4{,}26$ $b = 1{,}71$ $c = 3{,}86$ $x = 2{,}73$
16	$y = \dfrac{\sqrt{\ln^2(ax+2) + \sin\left(bx^2 - 1\right)}}{x^2}$	$a = 4{,}3$ $b = 2{,}9$ $x = 1{,}8$
17	$y = \sqrt{x + e^{ax}} \cdot \ln\dfrac{bx^2 - 1}{cx^2 + 3}$	$a = 2{,}44$ $b = 1{,}39$ $c = 6{,}21$ $x = 3{,}10$
18	$y = \dfrac{2\sqrt{\sin\left(ax^3 + 3\right) + bx^2}}{e^{-x} + 3{,}2}$	$a = 4{,}17$ $b = 3{,}69$ $x = 1{,}2$
19	$y = \sqrt{ax^2 + bx^3 + 9{,}2}\cdot\ln(2 + \cos x)$	$a = 6{,}27$ $b = 2{,}73$ $x = 2{,}83$
20	$y = e^{\sqrt{\cos(bx)+x}}\cdot\sin\left(\dfrac{\sqrt{ax+1}}{c}\right)$	$a = 2{,}13$ $b = 4{,}7$ $c = 2{,}6$ $x = 1{,}2$
21	$y = x^{\ln x}\cdot e^{\sqrt{ax + tg(bx)}}$	$a = 3{,}2$ $b = 1{,}67$ $x = 3{,}49$
22	$y = \dfrac{\sqrt{ax} + b\cos x}{e^{cx} + 2}$	$a = 2{,}71$ $b = -6{,}23$ $c = 3{,}34$ $x = 2{,}43$
23	$y = \dfrac{ax^2 + \sqrt{\ln x + a^2}}{b\cos x + 4{,}7}$	$a = -1{,}83$ $b = -2{,}15$ $x = 3{,}57$
24	$y = arctg\left(e^{-ax}\right) + \dfrac{\ln(b+x)}{x^3}$	$a = 0{,}21$ $b = 2{,}19$ $x = 3{,}74$
25	$y = \dfrac{a\cos x + be^{\sin x}}{\ln x + cx^4}$	$a = 1{,}93$ $b = 3{,}48$ $c = 0{,}27$ $x = 1{,}44$
26	$y = \dfrac{\sqrt{a + cx} + \ln x}{\left\lvert ax^2 + x + b\right\rvert}$	$a = 5{,}72$ $b = 4{,}48$ $c = 1{,}72$ $x = 1{,}29$
27	$y = \dfrac{a^2 + x^2 e^{-b+x^3}}{\sin(cx) + 4{,}79}$	$a = 0{,}83$ $b = 1{,}16$ $c = 2{,}72$ $x = 1{,}63$

28	$y = \frac{tg\left(c + x^2\right) - \sqrt{e^x + bx}}{a^2 + x}$	$a = 1{,}3$ $b = 2{,}8$ $c = 0{,}9$ $x = 3{,}5$
29	$y = \frac{\ln\left(ax^2 + c\right) + \sin\left(bx\right)}{e^{2x-4}}$	$a = 4{,}53$ $b = 3{,}19$ $c = 1{,}73$ $x = 0{,}58$
30	$y = \frac{a \cdot ctgx + x^{\cos x}}{b + e^{-cx}}$	$a = 2{,}63$ $b = 3{,}71$ $c = 0{,}32$ $x = 1{,}29$

Лабораторная работа

2.1. Задание 2. Вложенные ветвления

Задание. Составить схему алгоритма и два варианта программы на языке Паскаль для вычисления значений функции y=f(x) при произвольных значениях x. Варианты заданий в таблице 2.2.

2.1.1. Варианты заданий на вложенные ветвления:

Таблица 2.2

Номер варианта	Y=F(x)	Исходные данные
1	$y=\begin{cases} x^3+2a & при\ \ x<-2 \\ \ln\lvert\cos bx\rvert & при\ \ -2\le x\le 5 \\ x^2e^x & при\ \ x>5 \end{cases}$	$a=2,1$ $b=6,7$ 1)$x=-2,37$ 2)$x=-0,49$ 3)$x=7,51$
2	$y=\begin{cases} a+\frac{1}{2}e^{-x} & при\ x\le 0 \\ \sin(b^2x) & при\ \ 0<x<4 \\ \sqrt{x^2+2a} & при\ \ x\ge 4 \end{cases}$	$a=7,1$ $b=3,2$ 1)$x=-3,04$ 2)$x=2,16$ 3)$x=5,37$
3	$y=\begin{cases} \sin\left(\ln\lvert x\rvert\right) & при\ x\le 1 \\ (4x+b)^2 & при\ 1<x\le 3 \\ \frac{1}{x^2+a^2} & при\ x>3 \end{cases}$	$a=2,73$ $b=1,68$ 1)$x=-0,37$ 2)$x=1,9$ 3)$x=4,58$
4	$y=\begin{cases} x+\frac{\cos(ax)}{x^2+1} & при\ 3\le x\le 5 \\ b\sin\frac{a}{x} & при\ x>5 \\ e^x+\ln\lvert x\rvert & при\ x<3 \end{cases}$	$a=3,9$ $b=4,6$ 1)$x=3,57$ 2)$x=7,49$ 3)$x=-1,73$
5	$y=\begin{cases} 2\cos^2(ax^2-b) & при\ x\le -2 \\ 3x^2+b & при\ \ x>3 \\ \sqrt{x^2+e^{ax}} & при\ \ -2<x\le 3 \end{cases}$	$a=1,3$ $b=2,5$ 1)$x=-3,16$ 2)$x=4,16$ 3)$x=1.78$

6	$y=\begin{cases} b-x^2-1 & при\ x\le 3 \\ \sqrt{\ln(x+a)} & при\ x\ge 8 \\ \cos^2(ax^2+3) & при\ 3<x<8 \end{cases}$	$a=7,1$ $b=4,2$ 1)$x=1,48$ 2)$x=9,17$ 3)$x=6,23$
7	$y=\begin{cases} a\cos^2 x-b\sin x^2 & при\ x\le 1 \\ b\ln x+x^3 & при\ 1<x\le 4 \\ \sqrt{x^2+ab} & при\ x>4 \end{cases}$	$a=2,6$ $b=5,1$ 1)$x=0,44$ 2)$x=3,67$ 3)$x=5,38$
8	$y=\begin{cases} \cos^3(ax)^2 & при\ x>2 \\ \sin^2 x+\dfrac{b}{x} & при\ x\le -1 \\ (2-x^2)^3 & при\ -1<x\le 2 \end{cases}$	$a=2,7$ $b=-3,59$ 1)$x=4,27$ 2)$x=-2,63$ 3)$x=1,39$
9	$y=\begin{cases} (ax+1)^4 & при\ x\le 3 \\ \dfrac{1}{2x^2+b\ln x} & при\ 3<x\le 5 \\ a\cos(b+x)^2 & при\ x>5 \end{cases}$	$a=1,8$ $b=3,3$ 1)$x=2,46$ 2)$x=4,3$ 3)$x=6,82$
10	$y=\begin{cases} 1+\sqrt{a+\lvert x\rvert} & при\ x\le 1 \\ 2+(ax)^2+e^x & при\ x>6 \\ x\sqrt{1+b\ln(a^2x)} & при\ 1<x\le 6 \end{cases}$	$a=6,72$ $b=4,85$ 1)$x=0,4$ 2)$x=7,5$ 3)$x=4,45$
11	$y=\begin{cases} x^2-ax & при\ x\le -1 \\ \dfrac{1}{x^2+2} & при\ x>4 \\ \sqrt[3]{(x+1)^2} & при\ -1<x\le 4 \end{cases}$	$a=1,7$ 1)$x=-2,61$ 2)$x=1,49$ 3)$x=5,56$
12	$y=\begin{cases} \dfrac{1}{1+a\lvert x\rvert} & при\ -2<x\le 0 \\ \cos(bx^2)+0,5x & при\ x\le -2 \\ \sqrt{1+e^{ax}} & при\ x>0 \end{cases}$	$a=2,1$ $b=0,7$ 1)$x=-1,47$ 2)$x=-4,28$ 3)$x=5,07$

13	$y=\begin{cases}\ln(\lvert x\rvert+\sqrt{ax^2+1}) & при\ x\le -2\\ arctg\dfrac{b}{x^2+1} & при\ x>5\\ \sqrt{a^2+x^2} & при\ -2<x\le 5\end{cases}$	$a=4,8$ $b=0,51$ 1)$x=-3,24$ 2)$x=7,62$ 3)$x=0,28$
14	$y=\begin{cases}e^{\sin x} & при\ x\le -1\\ \ln^2\lvert bx\rvert & при\ x>5\\ \sqrt{1+(ax)^2} & при\ -1<x\le 5\end{cases}$	$a=0,19$ $b=6,1$ 1)$x=-4,38$ 2)$x=8,2$ 3)$x=3,74$
15	$y=\begin{cases}e^x+1 & при\ x\ge 1\\ \cos^2\sqrt{ax} & при\ 0<x<1\\ \ln\left(b+\sqrt{\lvert x\rvert}\right) & при\ x\le 0\end{cases}$	$a=5,5$ $b=3,1$ 1)$x=2,61$ 2)$x=0,53$ 3)$x=-4,39$
16	$y=\begin{cases}ax+bx^2 & при\ x<3\\ e^x+x^2 & при\ 3\le x\le 6\\ \sin^2 bx & при\ x>6\end{cases}$	$a=7,2$ $b=3,9$ 1)$x=-0,38$ 2)$x=4,19$ 3)$x=9,13$
17	$y=\begin{cases}\dfrac{1}{(1+x)^2} & при\ x\le -1\\ x^2+\cos a & при\ x>1\\ \sin(ax+b) & при\ -1<x\le 1\end{cases}$	$a=2,7$ $b=1,5$ 1)$x=-4,5$ 2)$x=-0,33$ 3)$x=2,53$
18	$y=\begin{cases}x+\cos ax & при\ 2\le x\le 5\\ \ln\lvert x+\sqrt{ax}\rvert & при\ x>5\\ arctg\dfrac{b}{x^2+1} & при\ x<2\end{cases}$	$a=4,8$ $b=0,64$ 1)$x=3,68$ 2)$x=6,7$ 3)$x=-4,51$

19	$y=\begin{cases}\dfrac{a+x}{1+\sqrt{\lvert x\rvert}} & при\ x\le 3\\ e^{b+x} & при\ x>5\\ \ln(ax+bx^2) & при\ 3<x\le 5\end{cases}$	$a=3,9$ $b=2,4$ 1) $x=1,38$ 2) $x=5,47$ 3) $x=3,2$
20	$y=\begin{cases}\sqrt{ax^2+b} & при\ x\le -2\\ \cos\dfrac{1}{1+\sqrt{a\lvert x\rvert}} & при\ -2<x\le 4\\ \ln\lvert x+\sin\ bx\rvert & при\ x>4\end{cases}$	$a=4,27$ $b=1,39$ 1) $x=-4,51$ 2) $x=2,75$ 3) $x=5,32$
21	$y=\begin{cases}\dfrac{e^{ax}+e^{-bx}}{2} & при\ 6<x\le 8\\ \sin ax+2 & при\ x\le 6\\ \cos^2 bx & при\ x>8\end{cases}$	$a=3,6$ $b=1,7$ 1) $x=7,24$ 2) $x=9,63$ 3) $x=-0,48$
22	$y=\begin{cases}\sqrt{x^2+\cos x} & при\ x\ge 2\\ e^{\sin ax} & при\ 0\le x<2\\ \ln(x^2+b) & при\ x<0\end{cases}$	$a=6,27$ $b=5,13$ 1) $x=3,18$ 2) $x=-4,6$ 3) $x=1,12$
23	$y=\begin{cases}\sin^2\sqrt{a+\lvert x\rvert} & при\ x\ge 5\\ e^{\frac{b}{x}}+1 & при\ x\le -3\\ \sqrt{x^2+ab^3} & при\ -3<x<5\end{cases}$	$a=2,2$ $b=3,4$ 1) $x=6,47$ 2) $x=-5,9$ 3) $x=1,94$
24	$y=\begin{cases}e^{bx+1} & при\ 2<x<7\\ \dfrac{1}{ax^3+1} & при\ x\ge 7\\ \ln\sqrt{1+(ax)^2} & при\ x\le 2\end{cases}$	$a=4,9$ $b=1,3$ 1) $x=4,27$ 2) $x=8,5$ 3) $x=-1,48$
25	$y=\begin{cases}\sqrt{ax^2+1} & при\ x\le 3\\ \ln(bx) & при\ 3<x<6\\ \cos\dfrac{3x^2}{1+ax} & при\ x\ge 6\end{cases}$	$a=2,7$ $b=4,4$ 1) $x=1,49$ 2) $x=5,3$ 3) $x=7,28$

26	$y=\begin{cases} ax+\dfrac{b}{x+1} & npu\ 1\le x\le 4 \\ \sin\dfrac{1}{bx+2} & npu\ x>4 \\ e^{\sqrt{a\lvert x\rvert+b}} & npu\ x<1 \end{cases}$	$a=0,46$ $b=1,39$ 1) $x=2,91$ 2) $x=5,62$ 3) $x=-0,76$
27	$y=\begin{cases} a\ \mathrm{tg}\,(\mathrm{bx}) & npu\ 0\le x\le 1 \\ \dfrac{1}{x}+\sin bx & npu\ x>1 \\ e^{-x+a} & npu\ x<0 \end{cases}$	$a=1,24$ $b=5,17$ 1) $x=0,61$ 2) $x=4,8$ 3) $x=-0,95$
28	$y=\begin{cases} b\sqrt{\lvert x\rvert^3} & npu\ x\ge 5 \\ a+3^x & npu\ 5>x>2 \\ e^{-x} & npu\ x\le 2 \end{cases}$	$a=1,76$ $b=2,34$ 1) $x=0,59$ 2) $x=1,06$ 3) $x=-0,58$
29	$y=\begin{cases} 2x+\sqrt{bx+3} & npu\ \ x\le -1 \\ \arccos\ x & npu\ \ -1<x<1 \\ \ln\,(x+a)^2 & npu\ x\ge 1 \end{cases}$	$a=0,65$ $b=1,43$ 1) $x=1,9$ 2) $x=0,16$ 3) $x=-2,52$
30	$y=\begin{cases} \sin bx+1 & npu\ \ x<0,5 \\ ctg\ x & npu\ \ 0,5\le x\le 2,5 \\ ax^3 & npu\ x>2,5 \end{cases}$	$a=0,55$ $b=4,31$ 1) $x=2,98$ 2) $x=0,21$ 3) $x=1,27$

Лабораторная работа

2.1. Задание 3. Оператор выбора

Задание. Составить схему алгоритма и программу на языке Турбо Паскаль для вычисления значений функции y=f(x) при произвольных значениях x. Варианты заданий в таблице 2.3.

2.1.1. Варианты заданий на оператор выбора

Таблица 2.3

Номер варианта	Y=F(x)	Исходные данные
1	$y=\begin{cases} x^3+2a & при\ \ x=-3 \\ \ln\lvert\cos bx\rvert & при\ \ x=4 \\ x^2e^x & при\ \ x=6 \end{cases}$	a=2,1 b=6,7 x=-2; 4; 6; 8
2	$y=\begin{cases} a+\frac{1}{2}e^{-x} & при\ x=-1 \\ \sin(b^2x) & при\ \ x=3 \\ \sqrt{x^2+2a} & при\ \ x=4 \end{cases}$	a=7,1 b=3,2 x=-1; 3; 4; 6
3	$y=\begin{cases} \sin(\ln\lvert x\rvert) & при\ x=-2 \\ (4x+b)^2 & при\ \ x=3 \\ \frac{1}{x^2+a^2} & при\ \ x=5 \end{cases}$	a=2,73 b=1,68 x=-2; 3; 5; 7
4	$y=\begin{cases} x+\frac{\cos(ax)}{x^2+1} & при\ x=5 \\ b\sin\frac{a}{x} & при\ x=8 \\ e^x+\ln\lvert x\rvert & при\ x=2 \end{cases}$	a=3,9 b=4,6 x=1; 2; 5; 8
5	$y=\begin{cases} 2\cos^2(ax^2-b) & при\ x=-2 \\ 3x^2+b & при\ \ x=4 \\ \sqrt{x^2+e^{ax}} & при\ \ x=3 \end{cases}$	a=3,9 b=4,6 x=-2; 1; 3; 4
6	$y=\begin{cases} b-x^2-1 & при\ x=3 \\ \sqrt{\ln(x+a)} & при\ \ x=8 \\ \cos^2(ax^2+3) & при\ \ x=7 \end{cases}$	a=7,1 b=4,2 x=3; 4; 7; 8

7	$y=\begin{cases} a\cos^2 x - b\sin x^2 & при\ x=1 \\ b\ln x + x^3 & при\ x=4 \\ \sqrt{x^2+ab} & при\ x=5 \end{cases}$	a=2,6 b=5,1 x=1; 2; 4; 5
8	$y=\begin{cases} \cos^3(ax)^2 & при\ x=3 \\ \sin^2 x + \dfrac{b}{x} & при\ x=-1 \\ (2-x^2)^3 & при\ x=1 \end{cases}$	a=2,7 b=-3,59 x=-1; 1; 3; 5
9	$y=\begin{cases} (ax+1)^4 & при\ x=3 \\ \dfrac{1}{2x^2+b\ln x} & при\ x=4 \\ a\cos(b+x)^2 & при\ x=6 \end{cases}$	a=1,8 b=3,3 x=1; 3; 4; 6
10	$y=\begin{cases} 1+\sqrt{a+\lvert x\rvert} & при\ x=1 \\ 2+(ax)^2+e^x & при\ x=7 \\ x\sqrt{1+b\ln(a^2x)} & при\ x=5 \end{cases}$	a=6,72 b=4,85 x=1; 5; 4; 7
11	$y=\begin{cases} x\ -ax & при\ x=-2 \\ \dfrac{1}{x^2+2} & при\ x=5 \\ \sqrt[3]{(x+1)^2} & при\ x=3 \end{cases}$	a=1,7 b=6,6 x=1; 2; 3; 5
12	$y=\begin{cases} \dfrac{1}{1+a\lvert x\rvert} & при\ x=0 \\ \cos)bx^2+0{,}5x & при\ x=-2 \\ \sqrt{1+e^{ax}} & при\ x=1 \end{cases}$	a=2,1 b=0,7 x=-2; 0; 1; 2
13	$y=\begin{cases} \ln(\lvert x\rvert+\sqrt{ax^2+1}) & при\ x=-2 \\ \arcsin\dfrac{b}{x^2+1} & при\ x=8 \\ \sqrt{a^2+x^2} & при\ x=1 \end{cases}$	a=4,8 b=0,51 x=-2; 1; 4; 8
14	$y=\begin{cases} e^{\sin x} & при\ x=-2 \\ \ln^2\lvert bx\rvert & при\ x=7 \\ \sqrt{1+(ax)^2} & при\ x=4 \end{cases}$	a=0,19 b=6,1 x=-2; 2; 4; 7
15	$y=\begin{cases} e^x+1 & при\ x=2 \\ \cos^2\sqrt{ax} & при\ x=1 \\ \ln(b+\sqrt{\lvert x\rvert} & при\ x=0 \end{cases}$	a=5,5 b=3,1 x=0; 1; 2; 5

16	$y=\begin{cases} ax+bx^2 & при\ x=4 \\ e^x+x^2 & при\ x=6 \\ \sin^2 bx & при\ x=9 \end{cases}$	a=7,2 b=3,9 x=1; 4; 6; 9
17	$y=\begin{cases} \dfrac{1}{(1+x)^2} & при\ x=-2 \\ x^2+\cos a & при\ x=2 \\ \sin(ax+b) & при\ x=1 \end{cases}$	a=2,7 b=1,5 x=-2; 1; 2; 3
18	$y=\begin{cases} x+\cos ax & при\ x=3 \\ \ln\left\|x+\sqrt{ax}\right\| & при\ x=6 \\ arctg\dfrac{b}{x^2+1} & при\ x=1 \end{cases}$	a=4,8 b=0,64 x=1; 3; 4; 6
19	$y=\begin{cases} \dfrac{a+x}{1+\sqrt{\left\|x\right\|}} & при\ x=3 \\ e^{b+x} & при\ x=12 \\ \ln(ax+bx^2) & при\ x=5 \end{cases}$	a=3,9 b=2,4 x=1; 3; 5;12
20	$y=\begin{cases} \sqrt{ax^2+b} & при\ x=-3 \\ \arccos\dfrac{1}{1+\sqrt{a\left\|x\right\|}} & при\ x=3 \\ \ln\left\|x+\sin\ bx\right\| & при\ x=5 \end{cases}$	a=4,27 b=1,39 x=-3; 2; 3; 5
21	$y=\begin{cases} \dfrac{e^{ax}+e^{-bx}}{2} & при\ x=8 \\ \sin ax+2 & при\ x=6 \\ \cos^2 bx & при\ x=9 \end{cases}$	a=3,6 b=1,7 x=1; 6; 8; 9
22	$y=\begin{cases} \sqrt{x^2+\cos x} & при\ x=2 \\ e^{\sin ax} & при\ x=1 \\ \ln(x^2+b) & при\ x=-2 \end{cases}$	a=6,27 b=5,13 x=-2; 1; 2; 5
23	$y=\begin{cases} \sin^2\sqrt{a+\left\|x\right\|} & при\ x=5 \\ e^{\frac{b}{x}}+1 & при\ x=-3 \\ \sqrt{x^2+ab^3} & при\ x=4 \end{cases}$	a=2,2 b=3,4 x=-3; 2; 4; 5

24	$y=\begin{cases} e^{bx+1} & при\ x=6 \\ \dfrac{1}{ax^3+1} & при\ x=7 \\ \ln\sqrt{1+(ax)^2} & при\ x=2 \end{cases}$	a=4,9 b=1,3 x=1; 2; 6; 7
25	$y=\begin{cases} \sqrt{ax^2+1} & при\ x=3 \\ \ln(bx) & при\ x=4 \\ \cos\dfrac{3x^2}{1+ax} & при\ x=6 \end{cases}$	a=2,7 b=4,4 x=1; 3; 4; 6
26	$y=\begin{cases} ax+\dfrac{b}{x+1} & при\ x=3 \\ \arcsin\dfrac{1}{bx+2} & при\ x=5 \\ e^{\sqrt{a\lvert x\rvert+b}} & при\ x=0 \end{cases}$	a=0,46 b=1,39 x=0; 2; 3; 5
27	$y=\begin{cases} a\,\mathrm{tg}\,(bx) & при\ x=1 \\ \dfrac{1}{x}+\sin bx & при\ x=2 \\ e^{-x+a} & при\ x=-1 \end{cases}$	a=1,24 b=5,17 x=1; 2; 4; 5
28	$y=\begin{cases} b\sqrt{\lvert x\rvert^3} & при\ x=5 \\ a+3^x & при\ x=3 \\ e^{-x} & при\ x=2 \end{cases}$	a=1,76 b=2,34 x=1; 2; 3; 5
29	$y=\begin{cases} 2x+\sqrt{bx+3} & при\ x=-1 \\ \arccos x & при\ x=0 \\ \ln(x+a)^2 & при\ x=1 \end{cases}$	a=0,65 b=1,43 x=-1; 0; 1; 2
30	$y=\begin{cases} \sin bx+1 & при\ x=0 \\ ctg\ x & при\ x=2 \\ ax^3 & при\ x=3 \end{cases}$	a=0,55 b=4,31 x=0; 2; 3; 5

1. Задания к лабораторной работе

Тема: Одномерный массив

4.1. Варианты заданий

Вариант № 1
Дан массив натуральных чисел. Найти сумму элементов, кратных данному К.

Вариант № 2
Дан массив целых чисел, в котором есть нулевые элементы. Создать массив из номеров этих элементов.

Вариант № 3
Дан массив из N целых чисел. Выяснить, какое число встречается в массиве раньше – положительное или отрицательное.

Вариант № 4
Дан массив из N натуральных чисел. Создать массив из чётных чисел этого массива. Если таких чисел нет, то вывести сообщение об этом факте.

Вариант № 5
Дан массив из N чисел. Указать наименьшую длину числовой оси, содержащую все эти числа.

Вариант № 6
Дан массив из N действительных чисел. Заменить все его члены, большие данного Z, этим числом. Подсчитать количество замен.

Вариант № 7
Дан массив действительных чисел, размерность которого N. Подсчитать, сколько в нем отрицательных, положительных и нулевых элементов.

Вариант № 8
Дан массив действительных чисел, размерность которого N. Поменять местами наибольший и наименьший элементы массива.

Вариант № 9
Дан массив A из N целых чисел. Вывести на печать только те числа, для которых выполняется условие $A_i \leq i$, где i – номер элемента массива.

Вариант № 10
Дан массив из N натуральных чисел. Указать те числа, остаток от деления которых на M равен L ($0 \leq L \leq M-1$).

Вариант № 11
В заданном одномерном массиве поменять местами соседние элементы, стоящие на чётных местах, с элементами, стоящими на нечётных.

Вариант № 12
При поступлении в вуз абитуриенты, получившие «двойку» на первом экзамене, ко второму не допускаются. В массиве A[n] записаны оценки экзаменующихся, полученные на первом экзамене. Подсчитать, сколько человек не допущено ко второму экзамену.

Вариант № 13
Дана массив чисел, среди которых имеется один нуль. Вывести на печать все числа включительно до нуля.

Вариант № 14
В одномерном массиве размещены: в первых элементах значения аргумента, а в следующих – соответствующие им значения функции. Напечатать элементы этого массива в виде двух параллельных столбцов: аргументы и значения функции.

Вариант № 15
Дан целочисленный массив с количеством элементов N. Напечатать те его элементы, индексы которых являются степенями двойки (1, 2, 4, 8, 16, …).

Вариант № 16
Дан массив из N действительных чисел. Напечатать те его элементы, которые принадлежат отрезку [c, d].

Вариант № 17
Дан массив целых положительных чисел. Найти произведение только тех чисел, которые больше заданного числа M. Если таких нет, то выдать сообщение об этом.

Вариант № 18
Массив из N элементов состоит из нулей и единиц. Поставить в начало этого массива нули, а затем единицы.

Вариант № 19
Дан массив из N действительных чисел, в котором есть только положительные и отрицательные элементы. Вычислить произведение отрицательных элементов P1 и произведение положительных элементов P2. Сравнить модуль P2 с модулем P1 и указать, какое из произведений по модулю больше.

Вариант № 20
Задан массив с количеством элементов N. Сформируйте два массива: в первый включите элементы исходного массива с чётными номерами, а во второй – с нечётными.

Вариант № 21
Составить программу нахождения наибольшего среди тех элементов одномерного массива A, что лежат в интервале [C, D].

Вариант № 22
Составить программу отыскания наименьшего среди тех элементов одномерного массива A, что лежат вне интервала [C, D].

Вариант № 23
Составить программу подсчёта среди элементов одномерного массива B количества чисел, больших C.

Вариант № 24
Составить программу отыскания наименьшего среди элементов одномерного массива A и его индекса.

Вариант № 25
Задан массив Y с количеством элементов N. Сформируйте массив, в котором элементы с чётными индексами будут равны соответствующим элементам исходного массива, а элементы с нечётными индексами будут равны нулю.

Вариант № 26
Составить программу подсчёта в одномерном массиве A суммы элементов с чётными индексами и суммы элементов, значения которых больше нуля.

Вариант № 27
Составить программу подсчёта в одномерном массиве C количества отрицательных и произведения положительных элементов массива.

Вариант № 28
Составить программу подсчёта в одномерном массиве B произведения элементов с нечётными индексами и суммы отрицательных элементов.

Вариант № 29
Заданы два одномерных массива A и B с одинаковым количеством элементов. Составить программу подсчёта суммы элементов с чётными индексами в массиве A и суммы элементов, значения которых больше нуля, в массиве B.

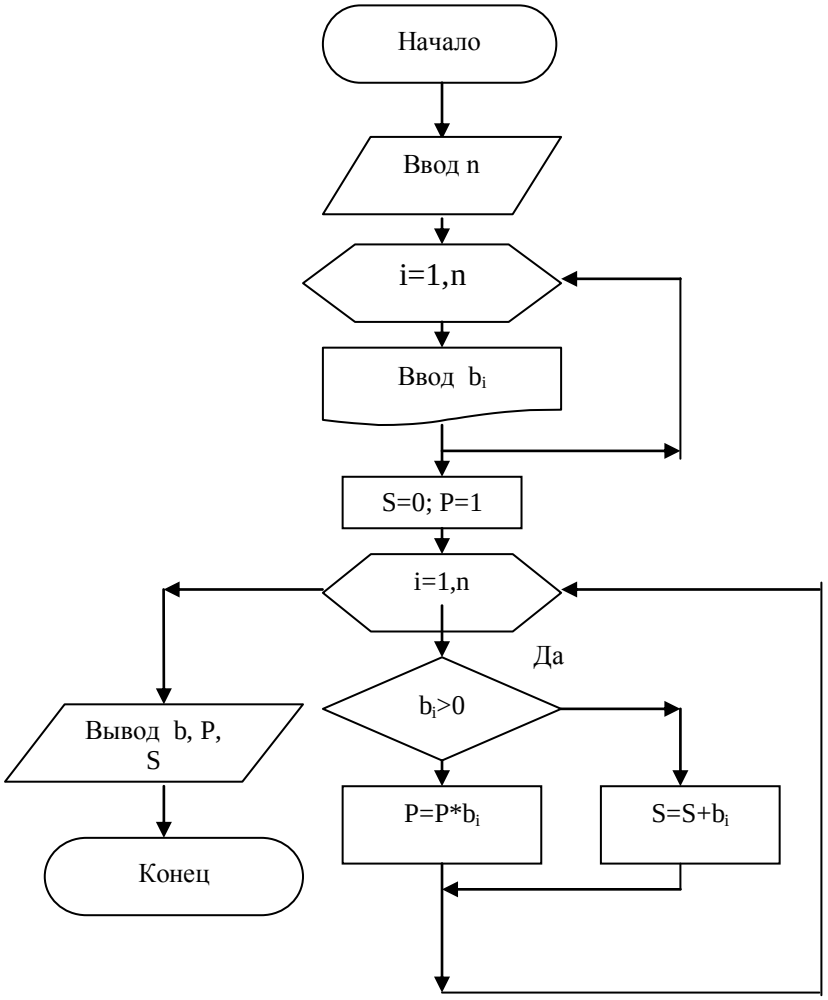
Вариант № 30
Заданы два одномерных массива A и B с одинаковым количеством элементов. Составить программу подсчёта суммы элементов с нечётными индексами в массиве B и произведения отрицательных элементов в массиве A.

4.2. Пример решения задачи

Задание**.** Составить программу подсчёта в одномерном массиве **B** из **n** элементов суммы отрицательных и произведения положительных элементов массива.

Решение**.** 1.Блок-схема решения задачи приведена на рисунке.

Рисунок 4.1



2.Текст программы.

```
var b:array [1..20] of integer;
   i,n,S,P:integer;
begin
write('введите n=');
readln(n);
for i:=1 to n do
begin
write('введите b[',i,']=');readln(b[i]);
end;
S:=0;
P:=1;
for i:=1 to n do
if b[i]>0 then P:=P*b[i]
 else S:=S+b[i];
writeln('     исходный массив b');
```

```
for i:=1 to n do
writeln(b[i]:5);
writeln;
writeln('Сумма S=',s);
writeln('Произведение P=',P);
readln;
end.
```

3.Результат выполнения контрольного примера.

```
 исходный массив b
  0
 -6
 -2
  67
  4
  2
Сумма S=-8
Произведение P=536
```

1. **Задания к лабораторной работе**

Тема: Двумерные массивы

5.1.Варианты заданий

Вариант № 1
Задан двумерный массив C из 4-х строк и 4-х столбцов (квадратная матрица). Составить программу подсчёта суммы всех отрицательных элементов и суммы элементов по главной диагонали.

Вариант № 2
Задан двумерный массив Y из 7-и строк и 3-х столбцов. Составить программу подсчёта суммы произведений элементов строк.

Вариант № 3
Задан двумерный массив A из 5-и строк и 2-х столбцов. Составить программу, которая формирует одномерный массив B, каждый элемент которого есть произведение элементов массива A в строке.

Вариант № 4
Задан двумерный массив B из 4-х строк и 4-х столбцов. Составить программу, которая организует двумерный массив, элементы главной диагонали которого равны соответствующим элементам исходного массива, а остальные элементы равны нулю.

Вариант № 5
Задан двумерный массив A из 2-х строк и 7-и столбцов. Составить программу, которая организует одномерный массив, каждый элемент которого есть сумма элементов исходного в столбце.

Вариант № 6
Задан двумерный массив Y из 5-и строк и 5-и столбцов. Составить программу подсчёта суммы всех положительных элементов и суммы элементов по главной диагонали.

Вариант № 7
Задан двумерный массив A из 4-х строк и 4-х столбцов. Составить программу, которая подсчитывает произведение элементов массива, лежащих вне главной диагонали.

Вариант № 8
Задан двумерный массив C из 6-и строк и 6-и столбцов. Составить программу, которая подсчитывает сумму всех элементов массива. Затем организовать формирование нового массива C, в котором элементы, лежащие на главной диагонали, равны 1, а остальные элементы равны соответствующим элементам исходного массива C.

Вариант № 9
Задан двумерный массив Y из 7-и строк и 3-х столбцов. Составить программу, которая вычисляет значение суммы произведений элементов строк

Вариант № 10
Задан двумерный массив B из 4-х строк и 5-и столбцов. Составить программу, которая организует одномерный массив, каждый элемент которого есть сумма элементов в столбце. Вычислить произведение элементов полученного массива.

Вариант № 11
Задан двумерный массив A из 5-и строк и 4-х столбцов. Составить программу, которая вычисляет значение произведения сумм строк.

Вариант № 12
Задан двумерный массив Y из 4-х строк и 4-х столбцов. Составить программу, которая вычисляет S – сумму элементов побочной диагонали и значение суммы всех элементов массива

Вариант № 13
Задан двумерный массив C из 6-и строк и 3-х столбцов. Составить программу, которая подсчитывает сумму всех элементов массива. Затем организовать формирование нового массива C, в котором элементы, лежащие не на главной диагонали, равны 1, а остальные элементы равны соответствующим элементам исходного массива C Вычислить произведение всех элементов нового массива.

Вариант № 14
Задан двумерный массив A из 6-и строк и 3-х столбцов. Составить программу, которая организует одномерный массив, каждый элемент которого есть произведение элементов массива A в каждой строке. Затем вычислить сумму элементов полученного одномерного массива.

Вариант № 15
Задан двумерный массив C из 3-х строк и 5-и столбцов. Составить программу, которая вычисляет произведение всех элементов массива. Затем организовать новый массив C, в котором значения элементов, лежащих на главной диагонали, равны 1, а остальные элементы равны квадрату соответствующих элементов исходного массива C.

Вариант № 16
Задан двумерный массив B из 4-х строк и 5-и столбцов. Составить программу, которая вычисляет сумму всех элементов массива.
Затем организовать новый массив B, в котором заменить отрицательные элементы исходного массива на 1, а значения остальных элементов оставить без изменения. Подсчитать количество замен.

Вариант № 17
Задан двумерный массив A из 8-и строк и 3-х столбцов. Составить программу, которая подсчитывает общее число неотрицательных элементов в массиве. Затем организовать формирование нового массива B, в котором значения элементов исходного массива заменить на противоположные по знаку.

Вариант № 18
Задан двумерный массив B из 6-и строк и 3-х столбцов. Составить программу, которая организует одномерный массив C, элементы которого равны количеству положительных элементов в строке исходного массива B.

Вариант № 19
Задан двумерный массив D из 5-и строк и 5-и столбцов. Составить программу, которая организует одномерный массив, элементы которого равны элементам массива D , лежащим на побочной диагонали, а затем вычисляет сумму элементов полученного одномерного массива.

Вариант № 20
Задан двумерный массив B из 4-х строк и 5-и столбцов. Составить программу, которая подсчитывает количество положительных, отрицательных и нулевых элементов в массиве B и организует одномерный массив из полученных значений.

Вариант № 21
Задан двумерный массив A из 3-х строк и 5-и столбцов. Составить программу, которая организует одномерный массив, каждый элемент которого есть количество положительных элементов исходного массива в столбце. Вычислить произведение элементов полученного массива.

Вариант № 22
Задан двумерный массив C из 2-х строк и 4-х столбцов. Составить программу, которая организует одномерный массив, каждый элемент которого есть количество отрицательных элементов исходного массива в строке. Вычислить сумму элементов полученного массива.

Вариант № 23
Задан двумерный массив B из 4-х строк и 4-х столбцов. Составить программу, которая организует одномерный массив, каждый элемент которого есть количество отрицательных элементов исходного массива в столбце. Вычислить сумму элементов полученного массива.

Вариант № 24
Задан двумерный массив B из 4-х строк и 4-х столбцов. Составить программу, которая организует двумерный массив, одна строка которого содержит количество ненулевых элементов исходного массива в столбце, а вторая – количество нулевых. Организовать проверку правильности формирования массива путём вычисления суммы элементов полученного массива.

Вариант № 25
Задан двумерный массив D из 5-и строк и 5- столбцов. Составить программу, которая организует новый массив B путём деления всех элементов заданной матрицы на элемент, наибольший по абсолютной величине.

Вариант № 26
Задан двумерный массив A из 3-х строк и 5-и столбцов. Составить программу, которая организует одномерный массив, каждый элемент которого есть наибольший элемент среди элементов в строке исходного массива.

Вариант № 27
Задан двумерный массив X из 3-х строк и 7-и столбцов. Составить программу, которая организует одномерный массив, каждый элемент которого есть наименьший по абсолютной величине элемент среди элементов в столбце исходного массива.

Вариант № 28
Задан двумерный массив A из 3-х строк и 5-и столбцов. Составить программу, которая организует одномерный массив, каждый элемент которого есть наибольший элемент среди элементов в столбце исходного массива.

Вариант № 29
Задан двумерный массив X из 3-х строк и 7-и столбцов. Составить программу, которая организует одномерный массив, каждый элемент которого есть наименьший по абсолютной величине элемент среди элементов в строке исходного массива.

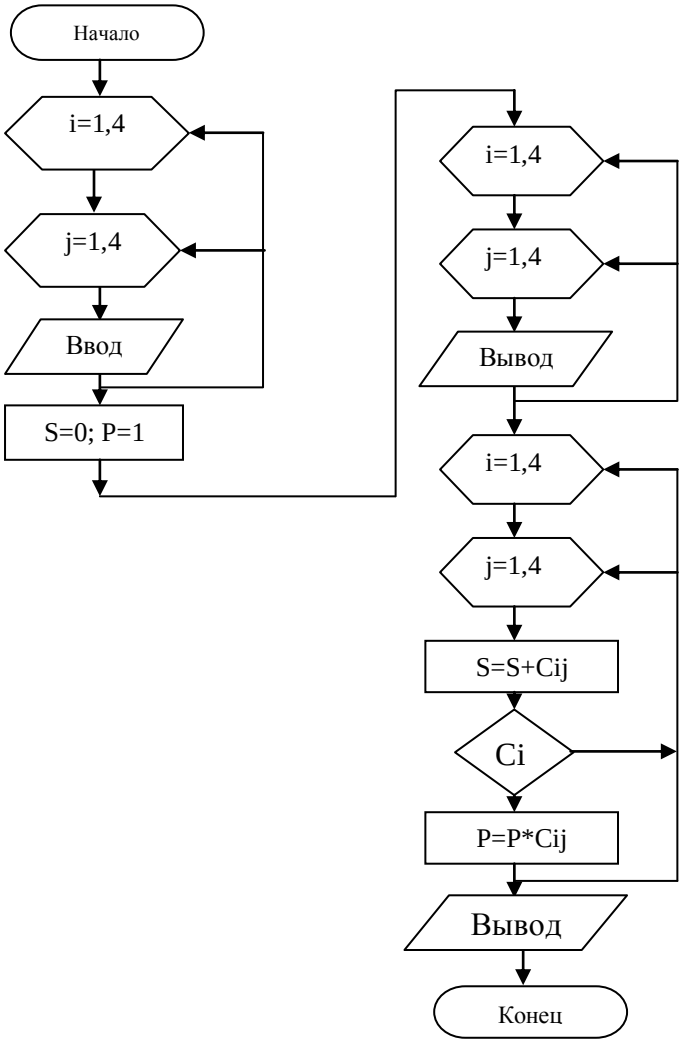
Вариант № 30
Задан двумерный массив A из 6-и строк и 6-и столбцов и одномерный массив Y из 6 строк. Составить программу, которая подсчитывает количество элементов, больших 1, а затем вычисляет произведение всех элементов массива.

5.2.Пример решения задачи

Задание. Задан двумерный массив С из 4-х строк и 4-х столбцов (квадратная матрица). Составить программу подсчёта суммы всех элементов массива и произведения отрицательных элементов.

Решение. 1.Блок-схема решения задачи приведена на рисунке.

Рисунок 5.1



2. Текст программы.

```
program p4;
const n=4;
var c:array [1..4,1..4] of real;
   S,P:real;
   i,j:integer;
begin
for i:=1 to n do
 for j:=1 to n do
begin
write('Введите c[',i,',',j,']=');
readln(c[i,j]);
end;
S:=0;
P:=1;
writeln('Исходный массив C');
for i:=1 to n do begin
 for j:=1 to n do
```

```
   write(c[i,j]:6:2);
writeln;
end;
for i:=1 to n do
 for j:=1 to n do begin
 S:=S+c[i,j];
 if c[i,j]<0 then P:=P*c[i,j];
end;
writeln('Сумма S=',s:7:2,' Произведение P=',P:7:2);
readln;
end.
```

3.Результат выполнения контрольного примера.

```
Исходный массив C
   2.00  -5.90   5.00 -65.00
  -3.98   0.00   8.00  76.40
  23.70   1.34  -9.00   1.00
   4.00   7.40   1.00   8.00
Сумма S=    53.96 Произведение P= 13736.97
```

Тест по теме:
«Информация. Кодирование информации»

Вариант 1

1. За минимальную единицу измерения информации принят….
   1) 1 бод;

   2) 1 пиксель;

   3) 1 байт;

   4) 1 бит.

2. В рулетке общее количество лунок равно 32. Какое количество информации мы получаем в зрительном сообщении об остановке шарика в одной из лунок.
   1) 8 бит;

   2) 5 бит;

   3) 2 бита;

   4) 1 бит.

3. Какое количество информации получит второй игрок при игре в крестики-нолики на поле 4×4 после первого хода первого игрока, играющего крестиками?
   1).5 бит;

   2) 4 бита;

   3) 3 бита;

   4) 2 бита.

4. Черно-белое (без градаций серого) растровое графическое изображение имеет размер 10×10 точек. Какой объем памяти займет это изображение?
   1) 100 бит;

   2) 100 байт;

   3) 10 Кбайт;

   4) 1000 бит.

5. Во сколько раз увеличится информационный объем страницы текста (текст не содержит управляющих символов форматирования) при его преобразования из кодировки MS-DOS (таблица кодировки содержит 256 символов) в кодировку Unicode (таблица кодировки содержит 65536 символов)?
   1) в 2 раза;

   2) в 8 раз;

   3) в 16 раз;

   4) в 256 раз.

6. В процессе преобразования растрового графического изображения количество цветов уменьшилось с 65536 до16. Во сколько раз уменьшится объём, занимаемый им памяти?
   1) в 2 раза;

   2) в 4 раза;

   3) в 8 раз;

   4) в 16 раз.

7. Как записывается десятичное число 11_{10} в двоичной системе счисления?
   1) 1111;

   2) 1101;

   3) 1011;

   4) 1001.

8. Преобразовать число 37_8 в шестнадцатеричную систему счисления…
   1) 37;

   2) 1F;

   3) 9A;

   4) F1.

9. Сложить числа E_{16} и 6_8. Сумму представить в двоичной системе счисления.
   1) 11110;

2) 10100;

3) 10110;

4) 10010.

Ответы теста

№ вопроса	1	2	3	4	5	6	7	8	9
№ прав. ответа	4	2	2	1	1	2	3	1	2

Вариант 2

1. Чему равен 1 байт?
   1) 8 бит;

   2) 2 бит;

   3) 10 бит;

   4) 10 бит.

2. Производится бросание симметричной четырехгранной пирамидки. Какое количество информации мы получаем в зрительном сообщении о её падении на одну из граней?
   1) 1 бит;

   2) 4 бита;

   3) 1 байт;

   4) 2 бита.

3. Какое количество информации получит второй игрок при игре в крестики-нолики на поле 8×8 после первого хода первого игрока, играющего крестиками?
   1) 4 бита;

   2) 5 бит;

   3) 6 бит;

   4) 7 бит.

4. Какое количество информации содержит один разряд восьмеричного числа?
   1) 1 байт;

   2) 3 бита;

   3) 4 бита;

   4) 1 бит.

5. Во сколько раз уменьшится информационный объем страницы текста (текст не содержит управляющих символов форматирования) при его преобразования из кодировки MS-DOS (таблица кодировки содержит 65536 символов) в кодировку Windows CP-1251 (таблица кодировки содержит 256 символов)?
   1) в 256 раз;

   2) в 8 раз;

   3) в 4 раза;

   4) в 2 раза.

6. Звуковая плата реализует 16-ти битное двоичное кодирование аналогового звукового сигнала. Это позволяет воспроизводить звук с …
   1) 8 уровнями интенсивности;

   2) 256 уровнями интенсивности;

   3) 16 уровнями интенсивности;

   4) 65536 уровнями интенсивности.

7. Как записывается десятичное число 12_{10} в двоичной системе счисления?
   1) 1111;

   2) 1110;

   3) 1011;

   4) 1001.

8. Преобразовать число AF_{16} в двоичную систему счисления…
   1) 10101111;

2) 10110101;

3) 10111000;

4) 10011111.

9. Сложить числа 1101_2 и 5_8. Сумму представить в двоичной системе счисления.
   1) 11110;

   2) 10100;

   3) 10110;

   4) 10010.

Ответы теста

№ вопроса	1	2	3	4	5	6	7	8	9
№ ответа	1	4	3	2	4	4	2	1	4

ВХОДНОЙ ТЕСТ ПО КУРСУ «ИНФОРМАТИКА»

ВАРИАНТ №1

1.За единицу измерения количества информации принято

- Мбайт
- бит
- байт
- Кбайт

2.**Производительность работы компьютера (быстрота выполнения операций) зависит от**

- размера экрана дисплея
- частоты процессора
- напряжения питания
- быстроты нажатия на клавиши

3.Какое устройство может оказывать вредное воздействие на здоровье человека?

- принтер
- монитор
- системный блок
- модем

4.Файл – это

- единица измерения информации
- программа в оперативной памяти
- текст, распечатанный на принтере
- программа или данные на диске

5.**Модель есть замещение изучаемого объекта другим объектом, который отражает**

- все стороны данного объекта
- некоторые стороны данного объекта
- существенные стороны данного объекта
- несущественные стороны данного объекта

6.**Минимальным объектом, используемым в текстовом редакторе, является**

- слово
- точка экрана (пиксел)
- абзац
- символ (знакоместо)

7.**Количество различных кодировок букв русского алфавита составляет**

- одну
- две (MS-DOS, Windows)
- три (MS-DOS, Windows,Macintosh)
- пять (MS-DOS, Windows,Macintosh, КОИ-8, ISO)
- пять (MS-DOS, Windows,Macintosh, КОИ-8, ISO)

8.**Инструментами в графическом редакторе являются**

- линия, круг, прямоугольник
- выделение, копирование, вставка
- карандаш, кисть, ластик
- наборы цветов (палитры)

9.**В состав мультимедиа-компьютера обязательно входят**

- проекционная панель
- CD-ROM дисковод и звуковая плата
- модем
- плоттер

10.**В электронных таблицах выделена группа ячеек A1:B3. Сколько ячеек входит в эту группу?**

- 6
- 5
- 4
- 3

11.**Основным элементом базы данных является**

- поле
- форма
- таблица
- запись

12.**Гипертекст – это**

- очень большой текст
- структурированный текст, в котором могут осуществляться переходы по выделенным меткам
- текст, набранный на компьютере
- текст, в котором используется шрифт большого размера

13.Какое устройство обладает наименьшей скоростью обмена информацией?

- CD-ROM дисковод
- жесткий диск
- дисковод для гибких дисков
- микросхемы оперативной памяти

14.**Заражение компьютерными вирусами может произойти в процессе**

- печати на принтере
- работы с файлами
- форматирования дискеты
- выключения компьютера

15.**Задан полный путь к файлу C:\DOC\PROBA.TXT Каково имя каталога, в котором находится файл PROBA.TXT?**

- DOC
- PROBA.TXT
- C:\DOC\PROBA.TXT
- TXT

16.**Генеалогическое дерево семьи является**

- табличной информационной моделью
- иерархической информационной моделью
- сетевой информационной моделью
- предметной информационной моделью

17.**Минимальным объектом, используемым в растровом графическом редакторе, является**

- точка экрана (пиксел)
- объект (прямоугольник, круг и т.д.)
- палитра цветов
- символ (знакоместо)

18.Наибольший информационный объем будет иметь файл, содержащий

- страницу текста
- черно-белый рисунок 100*100
- аудиоклип длительностью 1 мин
- видеоклип длительностью 1 мин

19.**В электронных таблицах формула не может включать в себя**

- числа
- имена ячеек
- текст
- знаки арифметических операций

20.**Информационной (знаковой) моделью является**

- анатомический муляж
- макет здания
- модель корабля
- диаграмма

ВАРИАНТ №2

1.**Чему равен 1 Кбайт?**

- 1000 бит
- 1000 байт
- (•) 1024 бит
- 1024 байт

2.**Какое устройство обладает наибольшей скоростью обмена информацией?**

- CD-ROM дисковод
- (•) жесткий диск
- дисковод для гибких дисков
- микросхемы оперативной памяти

3.**В целях сохранения информации гибкие диски необходимо оберегать от**

- холода
- (•) загрязнения
- магнитных полей
- перепадов атмосферного давления

4.**Системная дискета необходима для**

- первоначальной загрузки операционной системы
- систематизации файлов
- хранения важных файлов
- “лечения” компьютера от вирусов

5.**Информационной моделью организации учебного процесса в школе является**

- правила поведения учащихся
- список класса
- расписание уроков
- перечень учебников

6.Каково будет значение переменной X после выполнения операций присваивания: X:=5 X:=X+1

- 5
- 6
- 1
- 10

7.**В текстовом редакторе при задании параметров страницы устанавливаются**

- гарнитура, размер, начертание
- отступ, интервал
- поля, ориентация
- стиль, шаблон

8.**Чтобы сохранить текстовый файл (документ) в определенном формате необходимо задать**

- размер шрифта
- тип файла
- параметры абзаца
- размеры страницы

9.**В электронных таблицах нельзя удалить**

- столбец
- строку
- имя ячейки
- содержимое ячейки

10.**Тип поля (числовой, текстовой и др.) в базе данных определяется**

- названием поля
- шириной поля

- количеством строк
- типом данных

11.**Процессор обрабатывает информацию**

- в десятичной системе счисления
- в двоичном коде
- на языке Бейсик
- в текстовом виде

12.**Задан полный путь к файлу C:\DOC\PROBA.TXT Каково расширение файла, определяющее его тип?**

- C:\DOC\PROBA.TXT
- DOC\PROBA.TXT
- PROBA.TXT
- TXT

13.**В текстовом редакторе основными параметрами при задании шрифта являются**

- гарнитура, размер, начертание
- отступ, интервал
- поля, ориентация
- стиль, шаблон

14**В процессе форматирования текста изменяется**

- размер шрифта
- параметры абзаца
- последовательность символов, слов, абзацев
- параметры страницы

15.**Растровый графический редактор предназначен для**

- создания чертежей
- построения графиков
- построения диаграмм
- создания и редактирования рисунков

16.**В электронных таблицах имя ячейки образуется**

- из имени столбца
- из имени строки
- из имени столбца и строки
- произвольно

17.Какое действие не рекомендуется производить при включенном компьютере?

- (•) вставлять/вынимать дискету
- ( ) отключать/подключать внешние устройства
- ( ) перезагружать компьютер, нажимая на кнопку RESET
- ( ) перезагружать компьютер, нажимая на клавиши CTRL – ALT – DEL

18.**В текстовом редакторе выполнение операции Копирование становится возможным после**

- ( ) установки курсора в определенное положение
- ( ) сохранения файла
- ( ) распечатки файла
- (•) выделения фрагмента текста

19.В текстовом редакторе выполнение операции Копирование становится возможным после

- ( ) установки курсора в определенное положение
- ( ) сохранения файла
- ( ) распечатки файла
- (•) выделения фрагмента текста

19.**К основным операциям, возможным в графическом редакторе, относятся**

- ( ) линия, круг, прямоугольник
- ( ) карандаш, кисть, ластик
- (•) выделение, копирование, вставка
- ( ) наборы цветов (палитра)

20.**Запись и считывание информации в дисководах для гибких дисков осуществляется с помощью**

- ( ) магнитной головки
- ( ) лазера
- ( ) сенсорного датчика
- (•) термоэлемента

Определите правильную структуру программы. Структура программы имеет вид:

(выберете один или несколько вариантов ответа)

1)
```
program 12345;
uses crt;
var x:integer;
begin
   begin
   ……..
   End:
End.
```

2).
```
program lab1;
uses crt;
var x:integer;
begin
   begin
   ……..
   End:
End.
```

3).
```
program 1_lab;
uses crt;
var x:integer;
begin;
  const n=10; ……..
End.
```

4).
```
program lab_1;
uses crt;
const n=5;
var x:integer;
begin;
   ……..
End.
```

1. Выберете правильное написание формулы для вычисления y=(x^2+e^{x-2}*$\frac{cosx}{sinx}$)-2 $\sqrt{4{,}18+x}$+cosx+x^2, с использованием оператора присваивания:

   1.y=(x*x+e(x-2)*cosx/sinx)-2*sqr(4,18+x)+cos(x)+sqrt(x)

   2.y:=(x*x+exp(x-2)*(cos(x)/sin(x)))-2*sqrt(4.18+x)+cos(x)+sqr(x)
   3.y:=(sqr(x)+ exp(x-2)*(cos(x)/sin(x)))-2*sqrt(4.18+x)+cos(x)+sqr(x))
   4.y:=sqr(x)+e*(x-2)*cos(x)/sin(x) -2*sqr(4.18+x)+cos(x)+sqrt(x)
2. Выберете правильное написание формулы для вычисления y=2 $\sqrt{4{,}18+x}$+cosx+x^2, с использованием оператора присваивания:

   1.y=2*sqr(4,18+x)+cos(x)+sqrt(x)

   2.y:=2*sqrt(4.18+x)+cos(x)+sqr(x)
   3.y:=2*sqrt(4.18+x)+cos(x)+sqr(x))
   4.y:=2*sqr(4.18+x)+cos(x)+sqrt(x)

3. Целый тип данных определяется как:
   1.Integer, longint, byte, word
   2.integer,real, longint, word, byte
   3.real,longint, string,bête, word, char
   4.byte, integer,real, longint, word, char, string

4. Над целыми типами определены такие операции:
   1. «+», «-», «*», «/», div, mod
   2. «+», «-», «*», «/», round, trunk, div, mod
   3. «+», «-», «*», «/», round, trunk
   4. «+», «-», «*», «/», round, trunk, div, mod, sqrt, sqr

Тест по теме «Арифметические выражения»,

«Типы данных»

1. Выберете правильное написание формулы для вычисления y=(x^2+e^{x-2}*$\frac{cosx}{sinx}$)-2 $\sqrt{4{,}18+x}$+cosx+x^2, с использованием оператора присваивания:

   1.y=(x*x+e(x-2)*cosx/sinx)-2*sqr(4,18+x)+cos(x)+sqrt(x)

   2.y:=(x*x+exp(x-2)*(cos(x)/sin(x)))-2*sqrt(4.18+x)+cos(x)+sqr(x)
   3.y:=(sqr(x)+ exp(x-2)*(cos(x)/sin(x)))-2*sqrt(4.18+x)+cos(x)+sqr(x))
   4.y:=sqr(x)+e*(x-2)*cos(x)/sin(x) -2*sqr(4.18+x)+cos(x)+sqrt(x)

2. Выберете правильное написание формулы для вычисления y= 2 $\sqrt{4{,}18+x}$+cosx+x^2+tg(x^3), с использованием оператора присваивания:

   1.y=2*sqr(4,18+x)+cos(x)+sqrt(x)+tan(x*x*x)

   2.y:=2*sqr(4.18+x)+cos(x)+sqr(x)+tan(x*x*x)
   3.y:=2*sqrt(4.18+x)+cos(x)+sqr(x)+sin(x*x*x*)/cos(x*x*x)
   4.y:=2*sqr(4.18+x)+cos(x)+sqr(x) +sin(x*x*x*)/cos(x*x*x)

3. Целый тип данных определяется как:
   1.Integer, longint, byte, word
   2.integer,real, longint, word, byte
   3.real,longint, string,bête, word, char
   4.byte, integer,real, longint, word, char, string

4. Определите значение переменной A:=n mod 10, если n=5678
   1. 567
   2. 678
   3. 5
   4. 8

5. Определите значение переменной A:=n div 10, если n=5678
   1. 567
   2. 678

3. 5
4. 8

6. Определите значение переменной A:=n mod (n div 10), если n=48
   1. 8
   2. 6
   3. 4
   4. 48

7. Над целыми типами определены такие операции:
   1. «+», «-», «*», «/», div, mod
   2. «+», «-», «*», «/», round, trunk, div, mod
   3. «+», «-», «*», «/», round, trunk
   4. «+», «-», «*», «/», round, trunk, div, mod, sqrt, sqr

Тест по теме «БАЗА ДАННЫХ»

Вариант №1

1. Один из основных типов информационных структур:
   A. логическая;
   B. база данных;
   C. строковая;
   D. дерево;
   E. числовая.
2. В реляционной БД информация организована в виде:
   A. сети;
   B. иерархической структуры;
   C. файла;
   D. дерева;
   E. прямоугольной таблицы.
3. Записью реляционной базы данных является:
   A. корень дерева;
   B. столбец таблицы;
   C. строка таблицы;
   D. ветви дерева;
   E. дерево.
4. Первичный ключ в реляционной базе данных служит для:
   A. организации новой структуры данных;
   B. указания типа поля;
   C. связи между различными структурами данных;
   D. связи между различными таблицами в реляционной базе данных;
   E. однозначного выделения записи в базе данных.
5. В реляционной базе данных связь между таблицами организована через:
   A. запросы;
   B. общие строки;
   C. условия поиска;
   D. поля, связанные по смыслу;
   E. условия сортировки.
6. Полем реляционной БД является:
   A. строка таблицы;
   B. корень дерева;
   C. дерево;
   D. столбец таблицы;
   E. ветви дерева.
7. Структура записей реляционной БД определяется в режиме:
   A. поиска;
   B. создания индексов;
   C. просмотра БД;
   D. сортировки записей;
   E. создания и редактирования БД.

1-D
2-E
3-C
4-E
5-D
6-D
7-E

ТЕСТ ПО ТЕМЕ «БАЗА ДАННЫХ»
Вариант №2

1. Базы данных — это:

A. информационные структуры, хранящиеся во внешней памяти;
B. программные средства, позволяющие организовать информацию в виде таблиц;
C. программные средства, обрабатывающие табличные данные;
D. программные средства, осуществляющие поиск информации;
E. информационные структуры, хранящиеся в оперативной памяти.

2. БД содержит информацию об учениках компьютерной школы: имя, номер группы, балл за тест, балл за задание, общее количество баллов. Какого типа должно быть поле ОБЩЕЕ КОЛИЧЕСТВО БАЛЛОВ?

A. символьного;
B. логического;
C. числового;
D. любого типа;
E. числового или логического.

3. БД содержит информацию о собаках из клуба собаководства: кличка, порода, дата рождения, пол, количество медалей за участие в выставках. Какие типы должны иметь поля?

A. текстовое, текстовое, числовое, текстовое, числовое;
B. текстовое, текстовое, дата, текстовое, числовое;
C. текстовое, текстовое, дата, числовое, числовое;
D. текстовое, текстовое, числовое, логическое, числовое;
E. текстовое, текстовое, дата, логическое, текстовое.

Используется реляционная база данных, заданная таблицей:

НАЗВАНИЕ	КАТЕГОРИЯ	КИНОТЕАТР	НАЧАЛО СЕАНСА
Буратино	Х/Ф	Рубин	14
Кортик	Х/Ф	Искра	12
Винни-Пух	М/Ф	Экран	9
Дюймовочка	М/Ф	Россия	10
Буратино	Х/Ф	Искра	14
Ну, погоди!	М/Ф	Экран	14
Два капитана	Х/Ф	Россия	16

Записи пронумерованы от 1 до 7 соответственно их порядку в таблице.

4. Какие записи будут выбраны по условию отбора
(НАЗВАНИЕ ="Буратино") И (КИНОТЕАТР="Россия" ИЛИ КИНОТЕАТР="Рубин")?

A. 1, 5, 7
B. 7
C. 1, 5

D. 1
E. 5

5. Сформулировать условие поиска, дающее сведения о всех художественных фильмах, начинающихся в период времени с 12 до 16.

A. КАТЕГОРИЯ = "Х/Ф" ИЛИ (НАЧАЛО СЕАНСА>=12 ИЛИ НАЧАЛО СЕАНСА<=16)
B. КАТЕГОРИЯ = "Х/Ф" И (НАЧАЛО СЕАНСА<=12 И НАЧАЛО СЕАНСА>=16)
C. КАТЕГОРИЯ = "Х/Ф" ИЛИ (НАЧАЛО СЕАНСА>=12 И НАЧАЛО СЕАНСА<=16)
D. КАТЕГОРИЯ = "Х/Ф" И (НАЧАЛО СЕАНСА>= 12 ИЛИ НАЧАЛО СЕАНСА<=16)
E. КАТЕГОРИЯ = " Х/Ф" И (НАЧАЛО СЕАНСА>=12 И НАЧАЛО СЕАНСА<=16)

6. В каком порядке будут идти записи, если их отсортировать по двум ключам НАЗВАНИЕ + КИНОТЕАТР в порядке возрастания?

A. 1, 5, 3, 4, 7, 2, 6
B. 5, 1, 3, 7, 4, 2, 6
C. 6, 2, 4, 7, 3, 1, 5
D. 6, 2, 7, 4, 3, 1, 5
E. 2, 5, 4, 7, 1, 3, 6

7. Какие записи будут выбраны по условию отбора?
НАЧАЛО СЕАНСА = 14 ИЛИ НЕ (КАТЕГОРИЯ = "Х/Ф")

A. 1, 5
B. 1, 3, 4, 5, 6
C. 5
D. 3, 4, 6
E. нет таких записей

ОТВЕТЫ
1-A
2-C
3-B
4-D
5-E
6-B
7-B

ТЕСТ ПО ТЕМЕ «БАЗА ДАННЫХ»
Вариант 3

1. В реляционной БД информация организована в виде:

A. сети;
B. иерархической структуры;
C. файла;
D. дерева;
E. прямоугольной таблицы.

2. Записью реляционной базы данных является:

A. корень дерева;
B. столбец таблицы;
C. строка таблицы;
D. ветви дерева;
E. дерево.

3. Полем реляционной БД является:

A. строка таблицы;
B. корень дерева;
C. дерево;
D. столбец таблицы;
E. ветви дерева.

4. Структура записей реляционной БД определяется в режиме:

A. поиска;
B. создания индексов;
C. просмотра БД;
D. сортировки записей;
E. создания и редактирования БД.

5.При закрытии таблицы СУБД MS Access **не предлагает** выполнить сохранение внесенных данных, потому что данные сохраняются …

- ✔ автоматически сразу же после ввода в таблицу
- только после закрытия всей базы данных
- автоматически при закрытии таблицы базы данных
- после ввода пользователем специальной команды *Сохранение данных*

6.В таблицу базы данных «Магазин», содержащую 10 столбцов информации о товаре (наименование, поставщик, количество, дата окончания срока

хранения, цена, вес, температура хранения), внесена информация о 125 видах товара.
Количество полей в таблице равно …

- 10
- 125
- 1250
- 135

7.В СУБД MS Access **не существует** запрос на _____ данных.

- создание
- обновление
- удаление
- добавление

8.В нижней части окна конструктора запросов MS Access располагается бланк запроса. Каждая строка этого бланка выполняет определенную функцию. Наиболее важная часть бланка запроса, в которой вводятся ограничения поиска (критерии поиска), называется «_____».

- Условие отбора
- Схема данных
- Вывод на экран
- Сортировка

9.Основными понятиями иерархической структуры являются …

- уровень, узел, связь
- отношение, атрибут, кортеж
- таблица, столбец, строка
- таблица, поле, запись

10. В таблицу базы данных «Аптека», содержащую 7 столбцов информации о товаре (наименование, поставщик, количество, дата окончания срока хранения, цена, вес, температура хранения), внесена информация о 15 видах товара.
Количество полей в таблице равно …

- 7

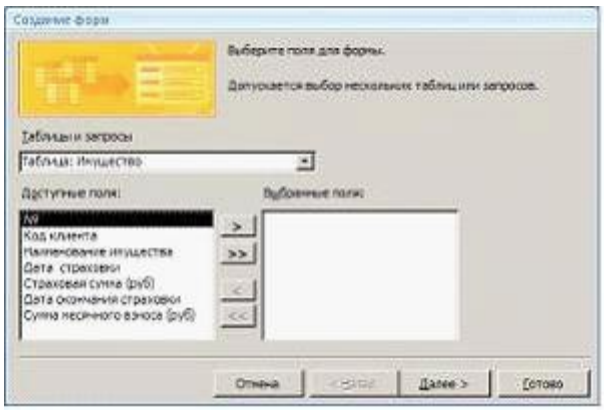
- 15
- 105
- 22

Ответы
1-E 2-C 3-D 4-E

Тест по теме «БАЗА ДАННЫХ»
Вариант №4

1. Один из основных типов информационных структур:
   A. логическая;
   B. база данных;
   C. строковая;
   D. дерево;
   E. числовая.
2. Первичный ключ в реляционной базе данных служит для:
   A. организации новой структуры данных;
   B. указания типа поля;
   C. связи между различными структурами данных;
   D. связи между различными таблицами в реляционной базе данных;
   E. однозначного выделения записи в базе данных.
3. В реляционной базе данных связь между таблицами организована через:
   A. запросы;
   B. общие строки;
   C. условия поиска;
   D. поля, связанные по смыслу;
   E. условия сортировки.
4. Базы данных — это:
   A. информационные структуры, хранящиеся во внешней памяти;
   B. программные средства, позволяющие организовать информацию в виде таблиц;
   C. программные средства, обрабатывающие табличные данные;
   D. программные средства, осуществляющие поиск информации;
   E. информационные структуры, хранящиеся в оперативной памяти.
5. БД содержит информацию об учениках компьютерной школы: имя, номер группы, балл за тест, балл за задание, общее количество баллов. Какого типа должно быть поле ОБЩЕЕ КОЛИЧЕСТВО БАЛЛОВ?
   A. символьного; B. логического;
   C. числового; D. любого типа;
   E. числового или логического.
6. БД содержит информацию о собаках из клуба собаководства: кличка, порода, дата рождения, пол, количество медалей за участие в выставках. Какие типы должны иметь поля?
   A. текстовое, текстовое, числовое, текстовое, числовое;
   B. текстовое, текстовое, дата, текстовое, числовое;
   C. текстовое, текстовое, дата, числовое, числовое;
   D. текстовое, текстовое, числовое, логическое, числовое;
   E. текстовое, текстовое, дата, логическое, текстовое.

6.Данное окно является окном _____ СУБД MS Access.



-мастера форм
-конструктора форм
-создателя форм
-построителя форм

7.Дан фрагмент базы данных «Автомобилисты».
Записи, удовлетворяющие запросу
Дата регистрации > 13.03.05 И Дата регистрации < 09.09.07,
имеют номера …

Автомобилисты : таблица

№ п/п	Владелец	Марка	Номер авто	Регион	Дата регистрации	Дата техосмотра
1	Колесников А.П.	Волга	в137ао	71	25.06.2006	04.05.2008
2	Петров А.И.	Хонда	м652ан	66	31.12.2006	03.08.2007
3	Крутов И.В.	Форд	к372оа	77	15.02.2005	12.04.2008
4	Ваина И.К.	Хонда	с356рв	77	13.03.2005	22.12.2007
5	Симонов А.П.	Жигули	о258ав	77	08.08.2007	14.01.2008
6	Антонов Ф.Г.	Хонда	с987км	78	03.11.2006	31.03.2008
7	Кошелев А.В.	Жигули	к145вт	66	25.01.2005	21.09.2007
8	Сидорова О.С.	Волга	м453нм	77	12.12.2007	02.06.2008

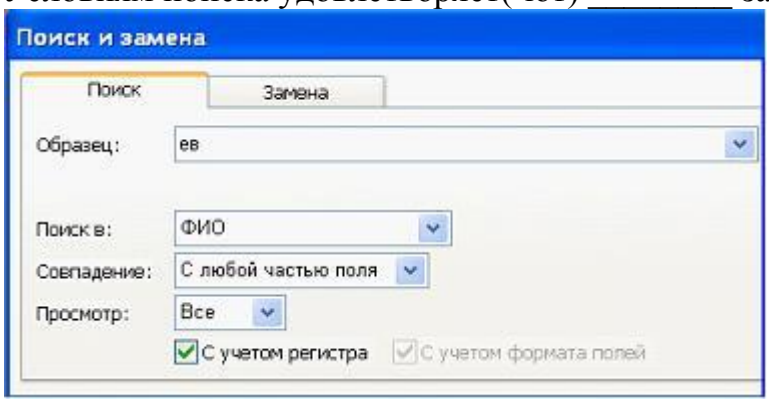
- 1, 2, 5, 6
- 1, 2, 5, 6, 7, 8
- 1, 2, 4, 5, 6
- 2, 6, 7

8.Представлена база данных «Тестирование».

Тестирование : таблица

Номер	ФИО	Пол	Математика	Физика	Информатика
1	Аганян Л.Г.	ж	82	59	52
2	Аксенов И.Н.	м	56	46	48
3	Васильева Л.И.	ж	43	38	32
4	Кондратьев О.Г.	м	74	54	63
5	Сергеева Т.В.	ж	62	62	60
6	Прокопьев И. В.	м	63	63	58
7	Черепанова О.С.	ж	72	70	59
8	Яшина Н.А.	ж	60	62	48
9	Севрюгин Н. А.	м	63	58	48
10	Евсюкова А. И.	ж	56	56	56

Условиям поиска удовлетворяет(-ют) _____ записей



-5

-4

-2

-6

9.В нижней части окна конструктора запросов MS Access располагается бланк запроса. Каждая строка этого бланка выполняет определенную функцию. Наиболее важная часть бланка запроса, в которой вводятся ограничения поиска (критерии поиска), называется «_____».

- ✔ Условие отбора
- Схема данных
- Вывод на экран
- Сортировка

10.Для эффективной работы с базой данных система управления базами данных (СУБД) должна обеспечивать _____ данных.

- ✔ непротиворечивость
- достоверность
- объективность
- кодирование

ОТВЕТЫ
1-D 2-E 3-D 4-A

ТЕСТОВЫЕ ЗАДАНИЯ
ПО ТЕМЕ «ИСТОРИЯ РАЗВИТИЯ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ»

ВАРИАНТ 1

1. Основным носителем информации, а также и средством ее хранения в конце XX века:
а) являлась бумага (изобретена в Китае во II веке нашей эры, в Европе бумага появилась в XI веке);
б) являлись кино и фотопленка (изобретены в XIX столетии);
в) являлась магнитная лента (изобретена в XX веке);
г) являлись дискета, жесткий диск (появились в 80-е годы XX века);
д) являлись лазерные компакт-диски (появились в последнем десятилетии XX века).

2. Первый арифмометр выполнявший четыре арифметических действия, сконструировал в XVII веке:
а) Ч. Бэббидж;
б) Б. Паскаль;
в) Г. Холлерит;
г) В. Лейбниц;
д) Дж. Буль.

3. Идея использования двоичной системы счисления в вычислительных устройствах принадлежит:
а) Ч. Бэббиджу;
б) Б. Паскалю;
в) Г. Лейбницу;
г) Дж. Булю;
д) Дж. Фон Нейману.

4. Решающий вклад вклад в возможность формальных преобразований логических выражений внес:
а) А. Тьюринг;
б) Г. Лейбниц;
в) Дж. Буль;
г) Ч. Бэббидж;
д) Н. Винер.

5. Одна из первых электронно-вычислительных машин ENIAC была создана под руководством:
а) Дж. Маучли и Дж. П. Эккерта;
б) Г. Айкена;
в) Д. Анастасова;
г) К. Цузе.
д) С.А. Лебедева

6. Первая отечественная ЭВМ, разработанная под руководством С. А. Лебедева, называлась:
а) БЭСМ;
б) Стрела;
в) МЭСМ;
г) Урал;
д) Киев.

7. Время появления первой ЭВМ в нашей стране.
а) XIX век;
б) первая половина XX века;
в) 1951 год;г) 60-е годы XX века.

8. Поколения ЭВМ отличаются друг от друга по:
а) автору создания вычислительной машины;
б) программным средствам;
в) элементной базе;
г) периоду создания вычислительной машины.

9. ЭВМ первого поколения:
а) имели в качестве элементной базы полупроводниковые элементы; программировались с использованием алгоритмических языков;
б) имели в качестве элементной базы электронные лампы; характеризовались малым быстродействием, низкой надежностью; программировались в машинных кодах;
в) имели в качестве элементной базы интегральные схемы, отличались возможностью доступа с удаленных терминалов;
г) имели в качестве элементной базы — большие интегральные схемы, микропроцессоры, отличались способностью обрабатывать различные виды информации
д) имели в качестве элементной базы — сверхбольшие интегральные схемы, обладали способностью воспринимать видео- и звуковую информацию.

10.Какая из отечественных ЭВМ была лучшей в мире ЭВМ второго поколения?
а) МЭСМ;
б) БЭСМ;
в) БЭСМ-6;
г) Минск-22.

11.Какое поколение машин позволяет нескольким пользователям работать с одной ЭВМ?
а) первое;
б) второе;
в) третье;
г) четвертое.

12.Что представляет собой большая интегральная схема?

а) на одной плате расположены различные транзисторы;
б) это набор программ для работы на ЭВМ;
в) это набор ламп, выполняющих различные функции;
г) это кристалл кремния, на котором размещаются от десятков до сотен логических элементов.

ВАРИАНТ 2

1. **Первым средством передачи информации на большие расстояния принято считать:**

а) радиосвязь;
б) электрический телеграф;
в) телефон;
г) почту;
д) компьютерные сети.

2. **Кто впервые сконструировал счетное устройство?**

а) Дж. Непер;
б) Б. Паскаль;
в) Ч. Бэббидж;
г) Дж. фон Нейман.

3. **Идея программного управления вычислительными процессами была впервые сформулирована:**

а) Н. Винером;
б) Дж. Маучли;
в) А. Лавлейс;
г) Ч. Бэббиджем;
д) Дж. Фон Нейманом.

4. **Кем были разработаны основные принципы цифровых вычислительных машин?**

а) Б. Паскаль;
б) Г. Лейбниц;
в) Ч. Бэббидж;
г) Дж. Фон Нейман.

5. **В каком году появилась первая ЭВМ?**

а) 1823;
б) 1946;
в) 1951;
г) 1949.

6. **Первая ЭВМ, разработанная в нашей стране называлась:**

а) БЭСМ;
б) Стрела;
в) МЭСМ;
г) Урал;

д) Киев.

7. Кто является основоположником отечественной вычислительной техники?
а) С. А. Лебедев;
б) М.В. Ломоносов;
в) П.Л. Чебышев;
г) Н.И. Лобачевский.

8. Что понимают под термином « поколение ЭВМ»?
а) все счетные машины;
б) все типы и модели ЭВМ, построенные на одних и тех же научных и технических принципах;
в) совокупность машин, предназначенных для обработки, хранения и передачи информации,
г) все электронные машины.

9. ЭВМ второго поколения:
а) имели в качестве элементной базы полупроводниковые элементы; программировались с использованием алгоритмических языков;
б) имели в качестве элементной базы электронные лампы; характеризовались малым быстродействием, низкой надежностью; программировались в машинных кодах;
в) имели в качестве элементной базы интегральные схемы, отличались возможностью доступа с удаленных терминалов;
г) имели в качестве элементной базы — большие интегральные схемы, микропроцессоры, отличались способностью обрабатывать различные виды информации
д) имели в качестве элементной базы — сверхбольшие интегральные схемы, обладали способностью воспринимать видео- и звуковую информацию.

10. В каком поколении машин появились первые программы?
а) в первом;
б) во втором;
в) в третьем;
г) в четвертом.

11. На какой серии вычислительных машин остановилось развитие отечественной электронной промышленности?
а) ЕС;
б) IBM;
в) Pentium;
г) Wax.

12. Целью создания “пятого поколения ЭВМ” является:
а) реализация новых принципов построения компьютера;
б) создание дешевых компьютеров;
в) достижение высокой производительности персональных компьютеров (более 10 млрд. операций в 1 с);

г) реализация возможности моделирования человеческого интеллекта (искусственного интеллекта);
д) создание единого человеко-машинного интеллекта.

ВАРИАНТ 3

1. Первоначальный смысл английского слова «компьютер»?
а) вид телескопа;
б) электронный аппарат;
в) электронно-лучевая трубка;
г) человек, производящий расчеты.

2. Когда было сконструировано первое в мире счетное устройство?
а) 1614 год;
б) 1642 год;
в) 1822 год;
г) 1886 год.

3. Состав и назначение частей (функциональных элементов) автоматического вычислительного устройства впервые сформулировал:
а) Дж.фон Нейман;
б) Ч. Бэббидж;
в) А.Лавлейс;
г) А. Тьюринг;
д) К. Шеннон.

4. Принцип хранимой программы был предложен:
а) Джоном фон Нейманом;
б) Чарльзом Бэббиджем;
в) Дж. П. Эккертом;
г) Аланом Тьюрингом;
д) Клодом Шенноном.

5. Как называлась первая ЭВМ?
а) Минск;
б) БЭСМ;
в) ENIAC
г) IBM

6. Первая советская ЭВМ – это:
а) Урал-1;
б) М-20;
в) МЭСМ;
г) БЭСМ.

7. Кто является основоположником отечественной вычислительной техники?
а) Д.Н. Лозинский;
б) С.А. Лебедев;

в) А.А. Макаров;
г) Д. Атанасов.

8. Человек, создавший первую программу:
а) Джон фон Нейман;
б) Чарльз Бэббидж;
в) Ада Лавлейс;
г) Алан Тьюринг;
д) Клод Шеннон

9. ЭВМ третьего поколения:
а) имели в качестве элементной базы полупроводниковые элементы; программировались с использованием алгоритмических языков;
б) имели в качестве элементной базы электронные лампы; характеризовались малым быстродействием, низкой надежностью; программировались в машинных кодах;
в) имели в качестве элементной базы интегральные схемы, отличались возможностью доступа с удаленных терминалов;
г) имели в качестве элементной базы — большие интегральные схемы, микропроцессоры, отличались способностью обрабатывать различные виды информации;
д) имели в качестве элементной базы — сверхбольшие интегральные схемы, обладали способностью воспринимать видео- и звуковую информацию.

10. Для машин какого поколения потребовалась специальность «оператор ЭВМ»?
а) для первого поколения;
б) для второго поколения;
в) для третьего поколения;
г) для четвертого поколения.

11. В каком поколении машин появились первые операционные системы?
а) в первом;
б) во втором;
в) в третьем;
г) в четвертом.

12. Общим свойством машины Беббиджа, современного компьютера и человеческого мозга является способность обрабатывать:
а) числовую информацию;
б) текстовую информацию;
в) звуковую информацию;
) графическую информацию.

ОТВЕТЫ К ТЕСТАМ

Вариант 1

1-а, 2-г, 3-в, 4-в, 5-а, 6-в, 7-в, 8-в, 9-б, 10-в, 11-в, 12-г

Вариант 2

1-г, 2-б, 3-г, 4-г, 5-б, 6-в, 7-а, 8-б, 9-г, 10-б, 11-а, 12-г

Вариант 3

1-г, 2-б, 3-б, 4-в, 5-в, 6-в, 7-б, 8-б, 9-в, 10-б, 11-в, 12-а

ТЕСТЫ по теме «СИСТЕМЫ СЧИСЛЕНИЯ»

1.1. Используя Правило Счета, запишите первые 20 целых чисел в десятичной, двоичной, троичной, пятеричной и восьмеричной системах счисления.

1.2. Какие целые числа следуют за числами:

а) 1_2;
б) 101_2;
в) 111_2;
г) 1111_2;
д) 101011_2;
е) 18;
ж) 78;
з) 378;
и) 1778;
к) 77778;
п) F16;
м) 1F16;
н) FF16;
о) 9AF916;
п) CDEF16 ?

1.3. Какие целые числа предшествуют числам:

а) 10_2;
б) 1010_2;
в) 1000_2;
г) 10000_2;
д) 10100_2;
е) 10_8;
ж) 20_8;
з) 100_8;
и) 110_8;
к) 1000_8;
л) 10_{16};
м)20_{16};
н) 100_{16};
о) $A10_{16}$;

п) 1000_{16} ?

1.4. Какой цифрой заканчивается четное двоичное число? Какой цифрой заканчивается нечетное двоичное число? Какими цифрами может заканчиваться четное троичное число?

1.5. Какое наибольшее десятичное число можно записать тремя цифрами:

- а) в двоичной системе;
- б) в восьмеричной системе;
- в) в шестнадцатеричной системе?

1.6. В какой системе счисления 21 + 24 = 100?
Решение.

Пусть x — искомое основание системы счисления.

Тогда $100_x = 1 \cdot x^2 + 0 \cdot x^1 + 0 \cdot x^0$, $21_x = 2 \cdot x^1 + 1 \cdot x^0$, $24_x = 2 \cdot x^1 + 4 \cdot x^0$.

Таким образом, $x^2 = 2x + 2x + 5$ или $x^2 - 4x - 5 = 0$.

Положительным корнем этого квадратного уравнения является x = 5. Ответ. Числа записаны в пятеричной системе счисления.

1.7. В какой системе счисления справедливо следующее:

- а) 20 + 25 = 100;
- б) 22 + 44 = 110?

1.8. Десятичное число 59 эквивалентно числу 214 в некоторой другой системе счисления.

Найдите основание этой системы.

1.9. Переведите числа в десятичную систему, а затем проверьте результаты, выполнив обратные переводы:

а) 1011011_2;
б) 10110111_2;
в) 011100001_2;
г) $0{,}1000110_2$;
д) $110100{,}11_2$;
е) 517_8;
ж) 1010_8;
з) 1234_8;
и) $0{,}34_8$;
к) $123{,}41_8$;
л) $1F_{16}$;
м) ABC_{16};
н) 1010_{16};
о) $0{,}A4_{16}$;
п) $1DE{,}C8_{16}$.

1.10. Переведите числа из десятичной системы в двоичную, восьмеричную и шестнадцатеричную, а затем проверьте результаты, выполнив обратные переводы:

а) 125_{10};

б) 229_{10};

в) 88_{10};

г) $37{,}25_{10}$;

д) $206{,}125_{10}$.

1.11. Переведите числа из двоичной системы в восьмеричную и шестнадцатеричную, а затем проверьте результаты, выполнив обратные переводы:

а) $1001111110111{,}0111_2$;
б) $1110101011{,}1011101_2$;
в) $10111001{,}101100111_2$;
г) $1011110011100{,}11_2$;
д) $10111{,}1111101111_2$;
е) $1100010101{,}11001_2$.

1.12. Переведите в двоичную и восьмеричную системы шестнадцатеричные числа:

а) $2CE_{16}$;

б) $9F40_{16}$;

в) $ABCDE_{16}$;

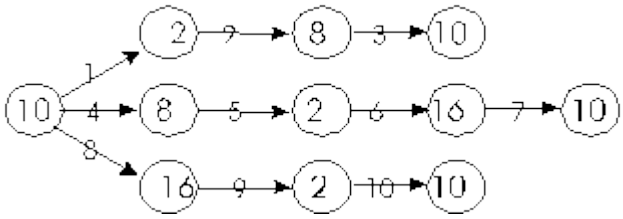
г) $1010{,}101_{16}$;

д) $1ABC{,}9D_{16}$.

1.13. Выпишите целые числа:

- а) от 101101_2 до 110000_2 в двоичной системе;
- б) от 202_3 до 1000_3 в троичной системе;
- в) от 14_8 до 20_8 в восьмеричной системе;
- г) от 28_{16} до 30_{16} в шестнадцатеричной системе.

1.14. Для десятичных чисел 47 и 79 выполните цепочку переводов из одной системы счисления в другую:



1.15. Составьте таблицы сложения однозначных чисел в троичной и пятеричной системах счисления.

1.16. Составьте таблицы умножения однозначных чисел в троичной и пятеричной системах счисления.

1.17. Сложите числа, а затем проверьте результаты, выполнив соответствующие десятичные сложения:

а) 1011101_2 и 1110111_2;

б) $1011{,}101_2$ и $101{,}011_2$;
в) 1011_2, 11_2 и $111{,}1_2$;
г) 1011_2 , $11{,}1_2$ и 111_2;
д) 37_8 и 75_8;
е) 165_8 и 37_8;
ж) $7{,}5_8$ и $14{,}6_8$;
з) 6_8, 17_8 и 7_8;
и) A_{16} и F_{16};
к) 19_{16} и C_{16};
л) $A{,}B_{16}$ и $E{,}F_{16}$;
м) E_{16}, 9_{16} и F_{16}.

1.18. В каких системах счисления выполнены следующие сложения? Найдите основания каждой системы:

а)
$$\begin{array}{r} 98 \\ +\ 89 \\ \hline 121 \end{array}$$

б)
$$\begin{array}{r} 1345 \\ +\ 2178 \\ \hline 3523 \end{array}$$

в)
$$\begin{array}{r} 10101 \\ +\ 1111 \\ \hline 1011 \\ \hline 20000 \end{array}$$

г)
$$\begin{array}{r} 765 \\ +\ 576 \\ \hline 677 \\ \hline 2462 \end{array}$$

д)
$$\begin{array}{r} 98 \\ +\ 56 \\ \hline 79 \\ \hline 167 \end{array}$$

1.19. Найдите те подстановки десятичных цифр вместо букв, которые делают правильными выписанные результаты (разные цифры замещаются разными буквами):

а)
$$\begin{array}{r} ABCD \\ +\ ABCD \\ \hline BDCEC \end{array}$$

б)
$$\begin{array}{r} A \\ +\ AB \\ \hline ABC \\ \hline BCB \end{array}$$

в)
$$\begin{array}{r} ABCDA \\ +\ FLCDA \\ \hline FLCLMN \end{array}$$

г)
$$\begin{array}{r} ABCD \\ +\ EFBCA \\ \hline GHGCIJ \end{array}$$

д)
$$\begin{array}{r} ABCD \\ +\ ABCEF \\ \hline EGDHIG \end{array}$$

1.20. Вычтите:

а) 111_2 из 10100_2;
б) $10{,}11_2$ из $100{,}1_2$;
в) $111{,}1_2$ из 10010_2;
г) 10001_2 из $1110{,}11_2$;
д) 15_8 из 20_8;
е) 47_8 из 102_8;
ж) $56{,}7_8$ из 101_8;

з) $16,54_8$ из $30,01_8$;
и) $1A_{16}$ из 31_{16};
к) $F9E_{16}$ из $2A30_{16}$;
л) $D,1_{16}$ из $B,92_{16}$;
м) ABC_{16} из 5678_{16}.

1.21. Перемножьте числа, а затем проверьте результаты, выполнив соответствующие десятичные умножения:

а) 101101_2 и 101_2;
б) 111101_2 и $11,01_2$;
в) $1011,11_2$ и $101,1_2$;
г) 101_2 и $1111,001_2$;
д) 37_8 и 4_8;
е) 16_8 и 7_8;
ж) $7,5_8$ и $1,6_8$;
з) $6,25_8$ и $7,12_8$.

1.22. Разделите 10010110_2 на 1010_2 и проверьте результат, умножая делитель на частное.

1.23. Разделите 10011010100_2 на 1100_2 и затем выполните соответствующее десятичное и восьмеричное деление.

1.24. Вычислите значения выражений:

- а) $256_8 + 10110,1_2 \cdot (60_8 + 12_{10}) - 1F_{16}$;
- б) $1AD_{16} - 100101100_2 : 1010_2 + 217_8$;
- в) $1010_{10} + (106_{16} - 11011101_2)\ 12_8$;
- г) $1011_2 \cdot 1100_2 : 14_8 + (100000_2 - 40_8)$.

1.25. Расположите следующие числа в порядке возрастания:

- а) 74_8, 110010_2, 70_{10}, 38_{16};
- б) $6E_{16}$, 142_8, 1101001_2, 100_{10};
- в) 777_8, 101111111_2, $2FF_{16}$, 500_{10};
- г) 100_{10}, 1100000_2, 60_{16}, 141_8.

1.26. Запишите уменьшающийся ряд чисел +3, +2, ..., -3 в однобайтовом формате:

- а) в прямом коде;
- б) в обратном коде;
- в) в дополнительном коде.

1.27. Запишите числа в прямом коде (формат 1 байт):

а) 31;

б) -63;

в) 65;

г) -128.

1.28. Запишите числа в обратном и дополнительном кодах (формат 1 байт):

а) -9;

б) -15;

в) -127;

г)-128.

1.29. Найдите десятичные представления чисел, записанных в дополнительном коде:

а) 1 1111000;

б) 1 0011011;

в)11101001;

г)10000000.

1.30. Найдите десятичные представления чисел, записанных в обратном коде:

а) 1 1101000;

б)10011111;

в)10101011;

г)10000000.

1.31. Выполните вычитания чисел путем сложения их обратных (дополнительных) кодов в формате 1 байт. Укажите, в каких случаях имеет место переполнение разрядной сетки:

а) 9 - 2;
б) 2 - 9;
в) -5 - 7;
г) -20 - 10;
д) 50 - 25;
е) 127 - 1;
ж) -120 - 15;
з) -126 - 1;
и) -127 - 1.

ОТВЕТЫ НА ТЕСТЫ

по теме «СИСТЕМЫ СЧИСЛЕНИЯ»

1.1.

в) троичная: 0, 1, 2, 10, 11, 12, 20, 21, 22, 100, 101, 102, 110, 111, 112, 120, 121, 122, 200, 201;

г) пятеричная: 0, 1, 2, 3, 4, 10, 11, 12, 13, 14, 20, 21, 22, 23, 24, 30, 31, 32, 33, 34.

1.2.

а) 10_2; **б)** 110_2; **в)** 1000_2; **г)** 10000_2; **д)** 101100_2; **е)** 2_8; **ж)** 10_8; **з)** 40_8;

и) 200_8; **к)** 10000_8; **л)** 10_{16}; **м)** 20_{16}; **н)** 100_{16}; **о)** $9AFA_{16}$; **п)** $CDF0_{16}$.

1.3.

а) 1_2; **б)** 1001_2; **в)** 111_2; **г)** 1111_2; **д)** 10011_2; **е)** 7_8; **ж)** 17_8; **з)** 77_8;

и) 107_8; **к)** 777_8; **л)** F_{16}; **м)** $1F_{16}$; **н)** FF_{16}; **о)** $A0F_{16}$; **п)** FFF_{16}.

1.4.

Четное двоичное число оканчивается цифрой 0, нечетное двоичное — цифрой 1, четное троичное — цифрами 0, 1 или 2.

1.5.

а) 7; **б)** 511; **в)** 4091.

1.7.

а) ни в какой;

б) в шестеричной.

1.8. Основание 5.

1.9.

а) 91; **б)** 183; **в)** 225; **г)** $^{35}/_{64}$; **д)** 52,75; **е)** 335; **ж)** 520; **з)** 668; **и)** $^{7}/_{16}$; **к)** $83^{33}/_{64}$;

л) 31; **м)** 2748; **н)** 4112; **о)** $^{41}/_{64}$; **п)** $478^{25}/_{32}$.

1.10.

а) 1111101_2; 175_8; $7D_{16}$; **б)** 11100101_2; 345_8; $E5_{16}$;

в) 1011000_2; 130_8;58_{16}; **г)** $100101,01_2$; $45,2_8$; $25,4_{16}$;

д) $11001110,001_2$; $316,1_8$; $CE,2_{16}$.

1.11.

а) $11767,34_8$; $13F7,7_{16}$; **б)** $1653,564_8$; $3AB,BA_{16}$; **в)** $271,547_8$; $B9,B38_{16}$;

г) $13634,6_8$; $179C,C_{16}$; **д)** $27,7674_8$; $17,FBC_{16}$; **е)** $1425,62_8$; $315,C8_{16}$.

1.12.

а) 1011001110_2; 1316_8; **б)** 1001111101000000_2; 117500_8

в) 10101011110011011110_2; 2536336_8;

г) $1000000010000,000100000001_2$; $10020,0401_8$;

д) $1101010111100,10011101_2$; $15274,472_8$.

1.13.

а) 101101_2, 101110_2, 101111_2, 110000_2;

б) 202_3, 210_3, 211_3, 212_3, 220_3, 221_3, 222_3, 1000_3;

в) 14_8, 15_8, 16_8, 17_8, 20_8;

г) 28_{16}, 29_{16}, $2A_{16}$, $2B_{16}$, $2C_{16}$, $2D_{16}$, $2E_{16}$, $2F_{16}$, 30_{16};

1.14.

а) 47_{10} - 101111_2 - 57_8 - 47_{10} - 57_8 - 101111_2 - $2F_{16}$ - 47_{10} - $2F_{16}$ - 101111_2 - 47_{10};

б) 79_{10} - 1001111_2 - 117_8 - 79_{10} - 117_8 - 1001111_2 - $4F_{16}$ - 79_{10} - $4F_{16}$ - 1001111_2 - 79_{10}.

1.15.

+	0	1	2	3	4
0	0	1	2	3	4

+	0	1	2
0	0	1	2
1	1	2	10
2	2	10	11

1	1	2	3	4	10
2	2	3	4	10	11
3	3	4	10	11	12
4	4	10	11	12	13

1.16.

x	0	1	2
0	0	0	0
1	0	1	2
2	0	2	11

x	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	11	13
3	0	3	11	14	22
4	0	4	13	22	31

1**.17.**
а) 11010100_2; **б)** $10001,0_2$; **в)** $10101,1_2$; **г)** $11001,1_2$; **д)** 134_8; **е)** 224_8; **ж)** $24,3_8$; **з)** 34_8; **и)** 19_{16}; **к)** 25_{16}; **л)** $19,A_{16}$; **м)** 26_{16}.

1.18.

а) в 16-й; **б)** в 10-й; **в)** в 3-й; **г)** в 8-й; **д)** в 16-й.

1.19.

в) A=9, B=4, C=5, D=3, F=1, L=0, M=7, N=8;

г) A=3, B=6, C=2, D=5, E=9, F=7, G=1, H=0, I=4, J=8;

д) A=9, B=3, C=4, D=2, E=1, F=8, G=0, H=7, I=6.

1.20.

а) 1101_2; **б)** $1,11_2$; **в)** $1010,1_2$; **г)** $-10,01_2$; **д)** 3_8; **е)** 33_8; **ж)** $22,1_8$; **з)** $11,25_8$;

и) 17_{16}; **к)** $1A92_{16}$; **л)** $-1,7E_{16}$; **м)** $4BBC_{16}$.

1.21.

а) 11100001_2; **б)** $11000110,01_2$; **в)** $1000000,101_2$; **г)** $1001011,101_2$;

д) 174_8; **е)** 142_8; **ж)** 15.26_8; **з)** 55.2222_8.

1.22.

1111_2.

1.23

1100111_2; 103_{10}; 147_8.

1.24.

а) 1493_{10}; **б)** 542_{10}; **в)** 1420_{10}; **г)** 11_{10}.

1**.25.**
а) 110010_2, 38_{16}, 74_8, 70_{10}; **б)** 142_8, 100_{10}, 1101001_2, $6E_{16}$;
в) 101111111_2, 500_{10}, 777_8, $2FF_{16}$; **г)** 1100000_2, 60_{16}, 141_8, 100_{10}.

1.26.

а) 00000011, 00000010, 00000001, 00000000, 10000001, 10000010, 10000011;

б) 00000011, 00000010, 00000001, 00000000, 11111110, 11111101, 11111100;

в) 00000011, 00000010, 00000001, 00000000, 11111111, 11111110, 11111101.

1.27.

а) 00001111; **б)** 10111111; **в)** 01000001; **г)** невозможно.

1.28.

Обратный: **а)** 11110110, **б)** 11110000, **в)** 10000000, **г)** невозможно.
Дополнительный: **а)** 11110111; **б)** 11110001; **в)** 10000001; **г)** 10000000.

1.29.

а) -8; **б)** -101; **в)** -23; **г)** -128.

1.30.

а) -23; **б)** -96; **в)** -84; **г)** -127.