Введение

Система программирования Турбо Паскаль, разработанная американской корпорацией Borland, остается одной из самых популярных систем программирования в мире. Этому способствуют, с одной стороны, простота лежащего в ее основе языка программирования Паскаль, а с другой стороны – труд и талант сотрудников Borland во главе с создателем Турбо Паскаля Андерсом Хейлсбергом.

Придуманный, примерно в 1970 году, швейцарским ученым Никласом Виртом как средство для обучения студентов программированию, язык Паскаль стараниями А. Хейлсберга превратился в мощную современную профессиональную систему программирования, которой по плечу любые задачи.

Данное пособие основывается на версию – Турбо Паскаль 7.0

Пакет Турбо Паскаль 7.0 обладает ограниченными возможностями и позволяет работать только в обычном режиме MS DOS. Начинающему программисту целесообразно начать изучение языка и среды именно с этого пакета.

Турбо Паскаль включает в себя как язык программирования – одно из расширений языка Паскаль для ЭВМ типа IBM, так и среду, предназначенную для написания, запуска и отладки программ.

Язык характеризуется расширенными возможностями по сравнению со стандартом, хорошо развитой библиотекой модулей, позволяющей использовать возможности операционной системы, создавать оверлейные структуры, организовывать ввод-вывод, формировать графические изображения и т.д.

Среда программирования позволяет создавать тексты программ, компилировать* их, находить ошибки и исправлять их, компоновать программы из отдельных частей, включая стандартные модули, отлаживать и выполнять отлаженную программу.

В данном пособии описаны основные возможности языка, которые предлагаются учащимся старших классов общеобразовательной школы. Отметим, что программа курса информатики (раздел программирование) казанской школы № 152 соответствует содержанию пособия.

1.Алфавит языка

Программа записывается в соответствии с синтаксисом Паскаля и используется его алфавит.

Алфавит – совокупность допустимых в языке символов.

1.1.Латинские буквы

A,B,...,Z;

a,b,…,z;

Прописные и строчные буквы не различаются.

1.2.Арабские цифры

0,1,2,…, 9;

1.3.Специальные символы

| { } | [ ] | ( ) | ‘ | := | ; | : | = | , | .. | . | ^ | @ | # | $ |

1.4.Знаки операций

Знаки операций предназначены для обозначения арифметических, логических или других действий (см. таблицы темы “”)

1.5.Служебные слова:

and, array,…,while, with, xor

1.6.Неиспользуемые символы

Некоторые символы, например: | % | & | и т.д. не используются в Турбо Паскале, но их можно использовать в комментариях и строках

2.Типы данных (простые)

Программа на Паскале делится на две части - это:

1. Описание данных, над которыми совершаются действия
2. Описание действий над данными

В соответствии с описанием данных компилятор распределяет память, т.е. выделяет ячейки нужной длины

Под типом данных понимается множество допустимых значений этих данных, а также совокупность операций над ними.

В Турбо Паскале имеются следующие группы простых типов:

2.1.Целые типы

Тип	***Диапазон***	***Формат***	***Размер в байтах***
Integer	2^{15}	Знаковый	2
Longint	2^{31}	Знаковый	4
Byte	2^{8}	Беззнаковый	1
Word	2^{16}	Беззнаковый	2
Shortint	2^{7}	Знаковый	1

В программе записывается как последовательность цифр со знаком или без.

Примеры:

+4; 168; -1245; +000;

2.2.Действительный (вещественный) тип Real

Под хранение значений действительного типа отводятся ячейки длиной в 6 байт. В этих 6-ти байтах под хранение порядка отводится столько бит, сколько позволяет записать порядок (до 99). Остальные биты отводятся под хранение мантиссы, что обеспечивает точность в 12 -14 значащих цифр.

В программе может записываться в двух формах:

- **естественная форма**

В виде последовательности цифр со знаком или без, в которой целая и дробная части разделены точкой.

Примеры:

+13.167; -0.0002;

- **форма с порядком**

mEp

m – мантисса – либо целое число, либо действительное в естественной форме.
P – порядок – двузначное целое число (не более 99)
E – десятичное основание степени
Пример:
3.123E-; 1234E12

2.3.Символьный тип
Char

Для хранения данных символьного типа выделяются ячейки длиной в 1 байт. В программе записывается как любой символ клавиатуры, заключенный в апострофы
Пример:
’1’; ’d’; ’#’;

3. Строковый тип

Для хранения данных строкового типа выделяются ячейки длиной 256 байт, 1-й байт содержит признак строки, поэтому длина строки не должна превышать 255 символов
В программе записывается, как последовательность символов, заключенных в апострофы.
Примеры:
’информатика’; ’**********’; ’35545767’;

2.4.Логический тип
Boolean

Для хранения данных логического типа выделяются ячейки длиной 1 байт. В программе записывается:
True - истина;
False - ложь.

2.5.Строковый тип (относится к структурным)
String

Для хранения данных типа String отводятся ячейки длиной 256 байт. 1-й байт содержит признак строки, поэтому длина строки не должна превышать 255 символов. В программе записывается, как последовательность символов, заключенных в апострофы.
Пример:
S:=’информатика’;

Задачи
1.К какому типу в Паскале относятся следующие последовательности символов?
а) –123; б) 0.003; в) ’d’; г)’1.245’; д) ’Учебник по информатике’; е) True
2.Как хранится в памяти компьютера следующая последовательность символов?
а)12; б)-34.02; в) ’s’
Сколько байт выделяется для хранения? Почему?

3.Выражения

Выражения определяют действия и порядок их выполнения при вычислении значений

Выражения в Паскале формируются из следующих элементов:

3.1.Константы

Константы – это величины, которые в ходе выполнения программы не изменяют свое значение. Они могут задаваться:

- **в явном виде** и тогда их вид определяет тип константы

Примеры:

17 - Integer; 3.14 - Real; 'h' - Char; 'Tyrbo Pascal' - String; True - Boolean

- **в виде именованной константы**, тогда имя константы должно быть описано и указано ее значение.

Примеры:

C1=17;
C2=3.14;
C3='h';

3.2.Переменные

Переменные используются для обозначения величин, которые изменяются в ходе выполнения программы.

Для обозначения переменных используются имена **(идентификаторы)**. В качестве имен можно использовать последовательность латинских букв, цифр, знак подчеркивания. Имена должны начинаться с буквы

Примеры:

Max
Summa_1
Y_1

Переменные разделяются на простые и структурированные. У простых переменных одному имени, в любой момент выполнения программы, соответствует только одно значение. У структурированных переменных, одному имени соответствует совокупность значений.

Пример структурированной переменной – таблица.

3.3.Стандартные функции

Стандартные функции служат для облегчения записи и обращения к наиболее часто используемым функциям обработки данных.

Имя функции	***Действие***	***Тип аргумента***	***Тип результата (значение функции)***
Арифметические функции			
PI		_____	Real
ABS(X)	|X|	Integer Real	Integer Real
SQR(X)	X^2	Integer Real	Integer Real

SIN(X)	Sin x	Integer Real	Real
COS(X)	Cos x	Integer Real	Real
EXP(X)	E^x	Integer Real	Real
LN(X)	Ln x	Integer Real	Real
SQRT(X)		Integer Real	Real
ARCTAN(X)	Arctg x	Integer Real	Real
Функции преобразования типов			
ORD(X)	Преобразует любой порядковый тип в целый (можно узнать ASCII-коды символов)	Любой порядковый тип	Integer
CHR(X)	Преобразует ASCII-коды в символ	Integer	Char
ROUND(X)	Округление до ближайшего целого	Real	Longint
TRUNG(X)	Целая часть числа	Real	Longint
Функции для величин порядкового типа			
PRED(X)	Определение предыдущего значения	Любой порядковый тип	Значение функции того же типа, что и аргумент
SUCC(X)	Определение следующего значения	Любой порядковый тип	Значение функции того же типа, что и аргумент
ODD(X)	Проверяет величину X на нечетность	Integer	Boolean
Функции разнообразного назначения			
RANDOM[(X)]	Формирует случайное число	Word X - параметр, указывающий диапазон значений случайного числа	Задан x - Word : 0<=Rez<x Не задан x – Real: 0.0<=Rez<1.0

3.4.Знаки операций

- ***Арифметические операции***

Знак	***Операция***	***Тип операндов***	***Тип результата***
+	Сложение	Целые Хотя бы один действительный	Целый Действительный
-	Вычитание	Целые Хотя бы один действительный	Целый Действительный
*	Умножение	Целые Хотя бы один действительный	Целый Действительный
/	Деление	Целые или действительные	Действительные
div	Деление целых чисел	Целые	Целый
mod	Остаток от деления целых чисел	Целые	Целый

- ***Операции отношения***

Знак	***Операция***	***Тип операндов***	***Тип результата***
=	Равно	Любой сравнимый тип (оба операнда должны быть одного типа)	Логический
<>	Не равно	----/-----	----/-----
>	Больше	----/-----	----/-----
>	Меньше	----/-----	----/-----
>=	Больше или равно	----/-----	----/-----
<=	Меньше или равно	----/-----	----/-----

- ***Логические операции***

X	***Y***	***X and Y***	***X or Y***	***X xor Y***
False	False	False	False	False
False	True	False	True	True
True	False	False	True	True
True	True	True	True	False

3.5. Круглые скобки

Круглые скобки используются для заключения в них части выражения, значения которой необходимо выполнить в первую очередь

Порядок вычисления выражений

Группа	***Типы действий***	***Операции или элементы***
1	Вычисления в круглых скобках	( )
2	Вычисление значения функции	Функции
3	Операции подобные умножению	*, / , div , mod, and
4	Операции типа сложения	+, - , or, xor
5	Операции отношения	= , <> , < , > , >= , <=, in

Пример:
(X>0) and (X<5)
В этом примере сначала выполняются 2 операции сравнения (операции 5-й группы), а затем логическая операция (операция 3 группы).
Задачи:
1.Какие из следующих последовательностей символов являются именами (идентификаторами)?
а) X; б)x1; в)Max; г)Symma_1; д)Kor 1; е)10_e; ж)Класс; з)R-1;
2.Найти значение функции:
а)Ord(’a’); б)Chr(125); в)Round(123.6); г)Trung(34.999); д)Pred(’Z’); е(Succ(1); ж)Odd(13);
3.Найти значение выражения:
а)17 Div 5; б)17 Mod 5; в)5>3; г)11<>11;
4.Какие из следующих последовательностей символов, являются выражениями, записанными по правилам Паскаля? К какому типу выражений они относятся? Из каких элементов состоят?
а)1; б)2-(Y Mod X)+Sqr(X); в)X^2+Y^4; г)2xy+15; д)(y>x) or (12=x); е)’11111+2222’;

4.Структура программы

Программа состоит из заголовка программы и блока программы

Заголовок:
Program <имя программы>;
<имя программы> - любой идентификатор.

Блок программы:
Состоит из 6-ти разделов, 5 из которых служат для описания данных, а 6-ой для описания действий над данными:
1. Раздел описания меток;
2. Раздел определения констант;
3. Раздел описания типов;
4. Раздел описания переменных;
5. Раздел описания процедур и функций;
6. Раздел операторов;

В Турбо Паскале порядок следования описательных разделов произвольный, но с обязательным соблюдением правила, что любое имя до его исполнения должно быть описано.
Некоторые из описательных разделов могут отсутствовать за ненадобностью.

4.1.Раздел описания меток
Label <метки>;

В Турбо Паскале в качестве меток разрешается использовать имена (идентификаторы). Метками помечаются те операторы программы, на которые предусмотрена передача управления. Метка, помечающая оператор, отделяется от него двоеточием. Все метки, используемые в разделе операторов, обязательно должны быть описаны в разделе описания меток.

Пример:
:
Label 1,3,15,155,Sum,Max;
:
155:A:=25+B;
Goto 155;
:
Те операторы, на которые нет передачи управления, не нужно снабжать метками.

4.2.Раздел определения констант

Const <записываются имена констант с указанием их значений>;

Пример:
Const g=15.37; Max=10000; Str=’sss’;

4.3.Раздел описания типов

(Рассмотрим ниже)

4.4.Раздел описания переменных

Var <записываются имена всех переменных, используемых в программе>;

Переменные одного типа можно описать общим списком.
Пример1:
Описать переменные квадратного уравнения $ax^2+bx+c=0$
Все переменные действительного типа
Var A,B,C,D,X1,X2:Real;
Пример2:
Если в программе используются переменные различных типов, то их разносят по спискам своего типа

```
Var A,B,C,D,X1,X2:Real;
      I,S,Max:Integer;
      Str:String;
```

4.5.Раздел описания процедур и функций

(Рассмотрим ниже)

4.6. Раздел операторов

Раздел операторов считается обязательным, он всегда последний. Этот раздел начинается со слова:
Begin
и заканчивается:
End.
Внутри записываются операторы.
Операторы размещаются в строке произвольно. Но принято соблюдать ступенчатую запись.
По назначению операторы можно разбить на следующие группы:
1 группа - из одного оператора присваивания.
Служит для вычисления значений выражений и присваивания их переменным

2 группа - операторы ввода-вывода.
Служат для обмена информацией между человеком и компьютером
3 группа – операторы обращения к процедуре.
Служат для структуризации программы
4 группа – операторы управления ходом выполнения программы.
Служат для изменения естественного порядка выполнения программы
По составу операторы разделяются на следующие виды:

- **простой оператор** (присваивания, ввода-вывода, обращения к процедуре);
- **сложный оператор или структурный** – это оператор, который содержит в своем составе другие операторы (операторы цикла, условный оператор, оператор варианта и т.д.);
- **составной оператор** – это группа операторов, заключенная в операторные скобки:

Begin
операторы
End;

Составной оператор может записываться в тех местах программы, в которых по синтаксическим правилам должен быть записан один оператор.
Задачи

1. Описать переменные, которые используются при нахождении площади треугольника по трем сторонам (формула Герона).
2. Описать переменные, которые используются при нахождении расстояния между двумя точками с координатами x1, y1 и x2, y2.

5.Программирование линейных алгоритмов

Линейные программы могут включать в себя операторы присваивания, ввода-вывода, обращения к процедурам.

5.1.Оператор присваивания

A:=B;

A – переменная;
B – выражение;
Типы переменной A и значение переменной B должны соответствовать друг другу.
Если переменная A - целого типа, то выражение B тоже должно иметь значение целого типа.
Задачи.
1.Какие из следующих последовательностей символов являются операторами присваивания?
а)X:=Y; б)Min=K+1; в)Summa+1:=W; г)S:=S; д)A:=’****’; е)Rrr=125+1*A ж)Str:=’120*Y’; з)T:=5>Y;
2.Пусть значения переменных X и У равны, соответственно, 3 и –2, какие значения будут иметь эти переменные после выполнения операторов присваивания?
а) X:=X+2*Y; Y:=Y/2;
в) Y:=-Y; X:=X+Y; Y:=Y+1;
г) X:=1; X:=X+Y;
д) X:=Y; Y:=X;

5.2.Операторы ввода-вывода

В общем случае ввод и вывод значений происходит из файлов привязанных к программе. Ввод с клавиатуры и вывод на экран рассматривается в Паскале, как обмен информацией между программой и стандартными файлами с именами Input (клавиатура) и Output (экран), поэтому при вводе с клавиатуры значений и выводе результатов на экран имена этих файлов не указываются.

5.3.Оператор вывода

Оператор вывода на экран имеет 3 модификации:

Write(B1,B2,…,Bn); - где

B1,B2,…,Bn – выражения типов: Integer, Char, Real, String, Boolean.

При выполнении оператора значения выражений вычисляются и выводятся в одну строку экрана.

Значения типа Integer - выводятся в обычной форме в виде целого числа.

Значения типа Real – в простейшем случае выводятся в форме с порядком, при этом мантисса содержит 7 цифр.

Пример:

Write(123.456); на экране – 1.234560E02

Write(-0.000123); на экране – - 1.230000E-04

Для удобства восприятия вывода существует возможность задания маски. Она задается в следующем виде:

Write(A:N:M);

N – целая константа, обозначающая общее число позиций, отводимых под значение A

M – целая константа, обозначающая число позиций, отводимых под дробную часть.

Пример:

Write(123.456:10:4);

		1	2	3	.	4	5	6	0

Write(123.456:6:2);

1	7	9	.	5	1

Значения типа Char и String выводятся в виде одного или последовательности символов.

Значения типа Boolean выводятся в виде True или False.

При выводе нескольких значений в одну строку возникает потребность отделять их друг от друга с помощью заданного числа пробелов. Для этого можно использовать элемент списка вывода вида - ’ ’:K – где

K – коэффициент повторения количества необходимых пробелов

Пример:

Write(’X1=’,X1:8:3,’ ’:10,’X2=’,X2:8:3);

Writeln(B1,B2,…,Bn);

Отличается от первого тем, что после вывода последнего значения курсор переводится в начало следующей строки экрана.

Writeln;

Выводит пустую строку.

Задача 1.
Вывести на экран фразу – «Моя первая программа»
Решение:

```
Program Lin1;
  Begin
    Writeln(’Моя первая программа’);
  End.
```

Задачи:
1.Какие из следующих последовательностей символов являются операторами вывода:
а) Write(X,Y); б)Writeln(X:Y:Z); в)Writeln г) Write(’Xx’,Xx,’Yy’,Yy); д)Writeln(1+2,3+4,Y/X); е) Writeln(’1+2’,’3+4’,’Y/X’); ж)Writeln(X:=Y,Z);
2.Создать на экране рисунок (использовать любые символы).

5.4.Оператор ввода

Оператор ввода значений с клавиатуры имеет вид:

Read(A1,A2,…,An); - где

A1, A2,…,An – имена переменных.
Следует иметь в виду, что вводимые значения не могут быть логическими.
При выполнении этого оператора компилятор приостанавливает свою работу, программист должен набрать значения переменных перечисляемых в списке ввода, учитывая следующие особенности:
- при вводе значений символьных переменных каждый очередной введенный символ становится значением очередной переменной;
- числовые значения должны разделяться пробелами или нажатием клавиши ввода;
- при вводе значений строковых переменных последний вводимый символ полностью включается в значение строковой переменной, поэтому невозможно в одном операторе ввести значение двух строковых переменных так, как нет символа разделяющего одно значение от другого.
-

Readln(B1,B2,…,Bn);

Отличается от первого тем, что после ввода последнего значения курсор переводится в начало следующей строки экрана.

Readln;

Ждет нажатия клавиши ввода.

Задача 2.
Составить программу – «Диалог с компьютером»
Решение:

```
Program Lin2;
  Var A:String;
  Begin
    Writeln(’Я компьютер, а как зовут тебя?’);
```

```
    Readln(A);
   Writeln('Очень приятно, ',A );
  End.
```

Задача 3.

Даны числа A,B,C. Переприсвоить их значения следующим образом: 

Исходные данные:

A=10, B=15, C=20

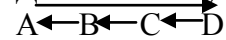
Результат:

A=20, B=10, C=15

Решение:

```
Program Lin3;
  Var A,B,C,D:Integer;
      Begin
      Writeln('Введите значения переменных A, B, C');
Read(A,B,C);
       D:=C;
       C:=B;
       B:=A;
       A:=D;
       Writeln('A=',A,'B=',B,'C=',C);
   End.
```

Задачи:

1. Какие из следующих последовательностей символов являются операторами ввода:
   а) Write(X,Y); б) Readln(X+Y,Y); в)Read(X,Y); г)Read('Xx',Xx,'Yy',Yy)?
2. Даны числа A,B,C,D. Переприсвоить их значения следующим образом:  (составить программу).
3. Какие числа будут выведены в результате выполнения последовательности операторов:
   Read(X); X:=X Mod 2; X:=Sqrt(X+1); Write(X,X*2); если в качестве исходного данного использовалось число:
   а) 2; б)1.2; в)6; г)5.4?
4. Вычислить расстояние между двумя точками с данными координатами X1,Y1 и X2,Y2.
5. На предприятии 2-м рабочим выделена премия (N руб). Разделить эту премию между рабочими в зависимости от стажа их работы на предприятии (прямо пропорциональная зависимость). Стаж вводится с клавиатуры.
6. По стороне основания и боковому ребру определить полную площадь поверхности правильной призмы:
   а) треугольной; б) четырехугольной; в)шестиугольной.
7. Продолжить программу –«Диалог с компьютером».
8. Дано целое число A. Не используя никаких функций и никаких операций кроме умножения получить:
   а) A^8 за три операции; б) A^{10} за четыре операции; в) A^7 за четыре операции;
9. Дано число X. Вычислить выражение $2x^4-3x^3+4x^2-5x+6$.Позаботиться об

экономии операций.

6.Программирование разветвляющихся алгоритмов.

Для организации разветвления в программе, в Паскале используются следующие операторы:

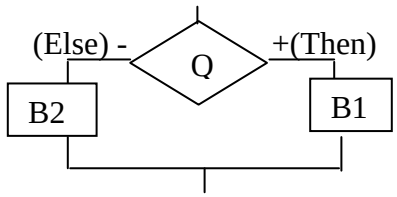
1. Условный оператор.
2. Оператор варианта.
3. Оператор перехода

6.1.Условный оператор

Условный оператор имеет вид:

If Q Then B1 [Else B2];

Блок-схема:



Q - выражение, принимающее логическое значение;
B1, B2 – любые операторы, в единственном числе;
Выполняются:

1. Вычисляется выражение Q, которое имеет логическое значение.
2. Если это значение True, то выполняется оператор B1. Если же Q имеет значение False, то в случае полной формы условного оператора выполняется оператор B1 , а в случае неполной формы управление передается следующему оператору программы.

Так, как операторы B1 и B2 могут быть любыми, а значит и условными, то возникает конструкция двусмысленная:

If Q Then If Q1 Then B1 Else B2;
(1 — If Q1 Then B1; 2 — If Q1 Then B1 Else B2)

Для устранения двусмысленности введено правило, что часть Else относится к ближайшему слева свободному условию. И структура записи следующая:

```
If Q Then
      If Q1 Then
            B1
        Else
          B2;
```

Если по смыслу задачи необходимо, чтобы оператор понимался в смысле 1 , то его надо поставить в операторные скобки:

```
If Q Then
      Begin
        If Q1 Then
```

```
                B1;
        End
    Else
      B2;
```

Задача1:

Даны 2 различных числа. Найти максимальное из этих чисел.

Пример:

Исходные данные: A=10, C=15

Результат: Максимальное число =15

Решение:

```
Program Vet1;
  Var A,C,Max:Integer;
   Begin
     Writeln('Введите 2 различных числа');
      Readln(A,C);
      If A>C Then
                Max:=A
        Else Max:=C;
      Write('Максимальное число = ',Max);
   End.
```

Задачи:

1. Даны 2 числа. Вывести первое число, если оно больше второго и оба числа, если это не так.
2. Даны 2 различных числа. Найти C=Max/Min.
3. Составить программу вычисления значения Y по введенному значению X.

$$Y=\begin{cases}(X+2)^2, & X>=0\\ X-2, & X<0\end{cases}$$

4. Даны числа A, B, X. Составить программу вычисления значения Y.

$$Y=\begin{cases}(X+2)^2, & X<A\\ X-2, & A<=X<=B\\ 2*X, & X>B\end{cases}$$

5. Даны 3 различных числа. Найти максимальное из этих чисел.
6. Даны 3 числа. Определить сколько среди них нулей.
7. Даны 3 числа. Определить сколько среди них отрицательных и сколько положительных чисел.

6.2.Использование составного оператора

Если по смыслу задачи, в зависимости от условий, надо выполнять либо одну, либо другую группу операторов, то их надо их надо превратить в один составной оператор (заключить в операторные скобки):

```
Begin
  операторы;
End;
If Q Then
```

```
        Begin
          A1;
          A2;
           :
           An;
        End
   Else
     Begin
       B1;
       B2;
        :
       Bn;
     End;
```

Задача 2:

Даны 2 различных числа. Вычислить $x=(y+z)^2$; - где y=max*2; z=max/min;

Решение:

```
Program Vet2;
  Var A,C,Max,Y:Integer;
      Z,X:Real;
  Begin
    Writeln('Введите 2 различных числа');
    Readln(A,C);
    If A>C Then
                Begin
                  Y:=A*2;
                  Z:=A/C;
                End
       Else
          Begin
            Y:=C*2;
            Z:=C/A;
          End;
    X:=SQR(Y+Z);
    Write('X = ',X);
  End.
```

Задача 3.

Даны числа A, C. Решить линейное уравнение A*X=C с полным анализом.

Решение:

```
Program Vet3;
  Var A,C,X:Real;
  Begin
    Writeln('Введите 2 числа');
    Readln(A,C);
    If A=0 Then
            If B=0 Then
                    Writeln('X -любое')
```

```
            Else
              Writeln('Решений нет')
        Else
          Begin
            X:=C/A;
             Writeln('X=',X);
          End;
  End.
```

Задачи:

1. Даны X,Y (X<>Y). Меньшее из этих двух чисел заменить полусуммой, а большее их удвоенным произведением
2. Даны числа A, B, C. Решить уравнение $AX^2+BX+C=0$ с полным анализом.
3. Даны 3 числа. Найти разность большего и меньшего из этих чисел
4. Если сумма трех различных чисел X,Y,Z меньше единицы, то меньшее из X,Y заменить полусуммой Y и Z, иначе большее из X и Y заменить произведением X*Y.

6.3.Использование составных логических условий (связки "и"; "или").

При записи составных логических условий следует помнить порядок выполнения операций.

Задача 3.

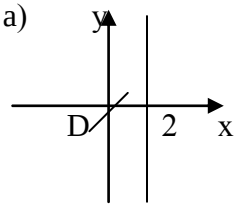
Даны числа X,Y,Z. Проверить, можно ли построить треугольник с данными сторонами.

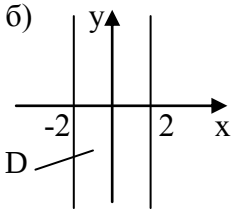
Решение:

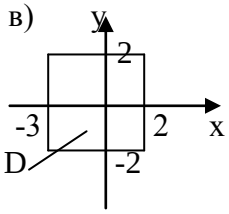
```
Program Vet3;
  Var X,Y,Z:Integer;
   Begin
     Writeln('Введите 3 числа');
      Readln(X,Y,Z);
      If (X+Y>Z) and (X+Z>Y) and (Y+Z>X) Then
                                          Writeln('Треугольник существует.')
         Else
           Writeln('Треугольник не существует.');
  End.
```

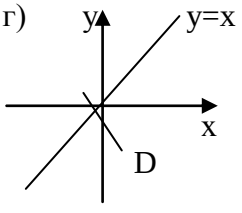
Задачи

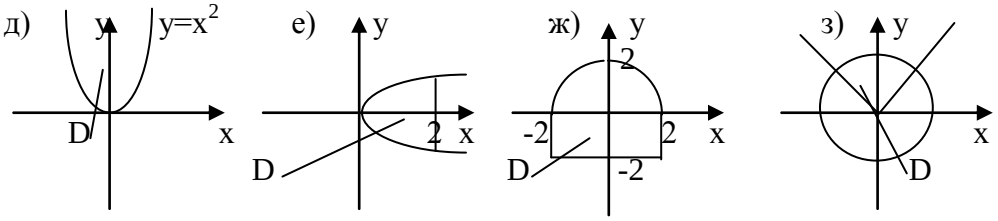
1. Даны числа A, B, C. Определить есть ли среди них отрицательные числа.
2. По введенной температуре определить здоров человек или болен.
3. Даны числа A, B, C. Определить какое из чисел лежит между двумя другими
4. Даны числа A, B, C. Вывести их на экран упорядоченными по возрастанию.
5. Дана точка M(X,Y). Определить, лежит ли данная точка внутри области D:

а) 

б) 

в) 

г) 



6.4.Оператор варианта

Case B of
<список меток 1>:S1;
<список меток 2>:S2;
:
<список меток n>:Sn;
[Else S;]
End;

S1, S2,…, Sn – любые операторы в единственном числе;

B – выражение, дающее значение порядкового типа, оно называется переключателем;

<список меток n> - это перечень констант того же типа, что и значение переключателя B. Разделяются запятой или (..).

Примеры:

1..50; 1,2,4; ’a’,’d’;

Выполняется:

1. Вычисляется значение переключателя B;
2. Отыскивается список меток, включающий это значение и выполняется соответствующий оператор;
3. Если значение переключателя не входит ни в один из списков меток, то в случае полной формы оператора Case, выполняется оператор S, а в случае неполной формы управление передается следующему оператору программы.

Здесь перед Else (;) ставится.

Задача 4:

По номеру дня недели вывести соответствующее ему название.

Решение:

```
Program Vet4;
  Var Z:Integer;
  Begin
     Writeln(’Введите номер дня недели’);
     Readln(Z);
     Case Z of
        1:Writeln(’Понедельник’);
        2:Writeln(’Вторник’);
        3:Writeln(’Среда’);
        4:Writeln(’Четверг’);
        5:Writeln(’Пятница’);
        6:Writeln(’Суббота’);
```

```
      7:Writeln('Воскресенье');
      Else
        Writeln('Дня с таким номером не существует.');
   End;
 End.
```

Задача 5:

По заданному символу определить, является ли он цифрой, латинской буквой или другим знаком.

Решение:

```
Program Vet5;
 Var Z:Char;
  Begin
    Writeln('Введите символ');
    Readln(Z);
    Case Z of
      '1','2','3','4','5','6','7','8','9','0':Writeln('Символ - цифра');
      'a'..'z':Writeln('Символ – латинская буква');
      Else
        Writeln('Другой знак.');
   End;
End.
```

Задачи:

1. Определить по введенному номеру месяца, количество дней в нем.
2. Определить, принадлежит ли введенное целое число к десяткам, сотням или тысячам.
3. По номеру квартала определить принадлежащие ему названия месяцев.
4. По номеру четверти координатной плоскости вывести знаки X и У.

6.5.Оператор перехода

Оператор перехода имеет вид:

Goto M;

M- метка. Все метки должны быть описаны в разделе описания меток;

В рассмотренных выше программах операторы выполнялись в том порядке, в каком они были записаны. Изменить этот порядок можно с помощью оператора перехода. Он прерывает естественную последовательность операторов: следом за ним выполняется оператор, помеченный указанной меткой.

Оператор Goto Met; передает управление на оператор с меткой Met.

Пусть программа содержит последовательность операторов:

```
X:=2; A:=X; Goto Met;
1:A:=A*2; B:=A; Met:Writeln(B);
```

В этом случае сначала выполняются операторы X:=2; A:=X; затем следует переход к оператору, помеченному меткой Met т.е. к оператору Writeln(B);

Задача 6.

Даны 3 отрицательных числа. Найти предыдущее и следующее значение максимального из этих чисел.

Решение:

```
Program Vet6;
  Label 1;
  Var A,B,C:Integer;
  Begin
  1:Writeln('Введите отрицательные числа');
   Readln(A,B,C);
   If (A>0) or (B>0) or (C>0) Then Goto 1;
   {далее задачу решаем самостоятельно}
   End;
```

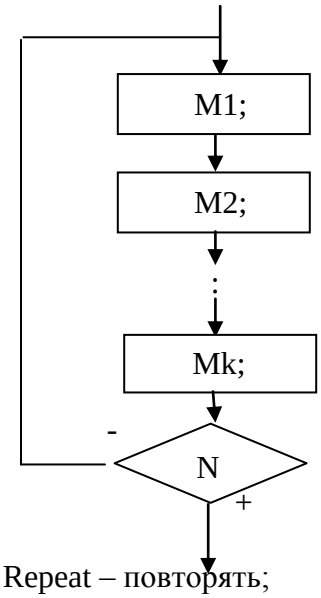
7.Программирование циклических алгоритмов

В Паскале существуют 3 оператора цикла:

7.1.Цикл с постусловием («До»)

Repeat
M1;
M2;
:
Mk;
Until N;

Блок - схема



Repeat – повторять;
Until – до;
M1, M2,…, Mk – группа операторов, они называются операторами тела цикла.
N – выражение, имеющее логическое значение (условие выхода из цикла)
Выполняется:

1. Выполняются операторы M1, M2,…, Mk;
2. Вычисляется значение N;
3. Если N – True, то оператор цикла заканчивает свою работу, иначе перейти к пункту 1

Среди операторов тела цикла обязательно должен присутствовать оператор, изменяющий значение выражения N;
Задача 1.
Найти произведение чисел от 1 до 10.
Решение:

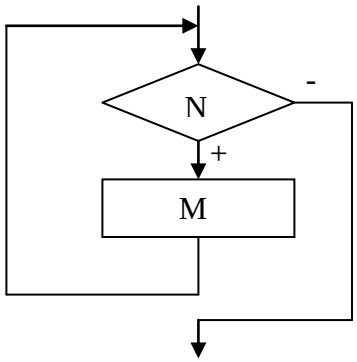
```
Program Ci1;
  Var K:Integer;
        P:Real;
  Begin
     K:=0;P:=1;
     Repeat
        K:=K+1;
        P:=P*K;
     Until K=10;
     Writeln('P= ',P:10:5);
 End.
```

Задачи:

1. Даны числа от 1 до N. Найти:а) сумму всех чисел; б) произведение всех чисел; в) среднее арифметическое.
2. Вычислить сумму всех двузначных чисел.
3. Вычислить сумму чисел: S=1+3+5+…+R где R-нечетное.
4. Вычислить произведение чисел: P=2+4+6+…+R где R-четное.
5. Найти значения функции $y=x^2$ при x=1,3,5,…,15.
6. Найти значения функции $p=(p^2+15)$ при p=2,4,6,…,20.

7.2.Оператор цикла с предусловием «Пока»

While N do M;



While – пока;
Do – выполнять;
N – выражение, имеющее логическое значение (условие входа в цикл);
M – любой оператор в единственном числе. Если по смыслу задачи надо повторять группу операторов, то их надо превратить в один составной оператор (заключить в операторные скобки). Этот оператор или группа операторов называется телом цикла;
Выполняется:

1. Вычисляется значение N;
2. Если N – True, то выполняется оператор M (группа операторов), иначе оператор цикла заканчивает свою работу;
3. Перейти к пункту 1.

Среди операторов тела цикла обязательно должен присутствовать оператор, изменяющий значение выражения N;

Задача 2.

Даны числа от 1 до N (N – четное). Найти: S= $\frac{1}{2}+\frac{1}{4}+\frac{1}{6}+.....+\frac{1}{N}$,

Решение:

```
Program Ci2;
  Var K,N:Integer;
        S:Real;
  Begin
     Writeln('Введите четное число');
     Readln(N);
     K:=0;S:=0;
     While  K<N do
      Begin
        K:=K+2;
        S:=S+1/K;
      End;
     Writeln('S= ',S);
  End.
```

Задачи:

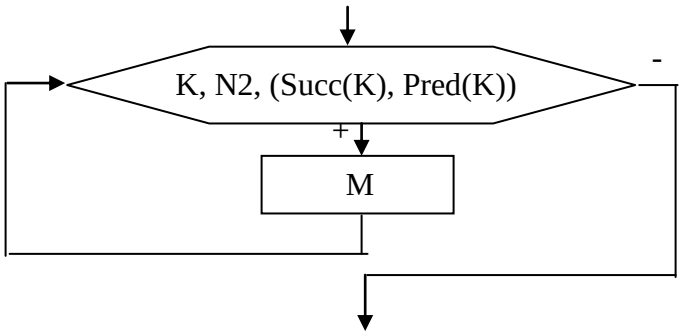
1. Даны числа от 1 до N. Найти: P= $\frac{1}{1}*\frac{1}{2}*\frac{1}{3}*.....*\frac{1}{N}$,
2. Даны числа от 1 до N (N – нечетное). Найти: S= $\frac{1}{1}+\frac{1}{3}+\frac{1}{5}+.....+\frac{1}{N}$,
3. Найти значение X: X=(1*3*5*…*N)/(1+3+5+…+N) (N – нечетное).
4. Даны числа от 1 до N. Вычислять: S= $\frac{1}{1}+\frac{1}{2}+\frac{1}{3}+.....+\frac{1}{N}$, вплоть до слагаемого, меньшего заданного E (E<1).
5. Вычислить среднее арифметическое чисел K,K+1,K+2,…,K+N; (K<N).
6. Вычислить сумму чисел: S=N+(N+1)+(N+2)+(N+3)+…+(N+R); (N<R).
7. Вычислить произведение чисел: P=N*(N+1)*(N+2)*(N+3)*…*(N+R); (N<R).
8. Вычислить: P=1+(1*2)+(1*2*3)+(1*2*3*…*R).
9. Вычислить: P=1*(1+2)*(1+2+3)*(1+2+3+…+R).
10. Найти значения функции M=N^2-N при N={2,4,6,…,T} (T-четное).
11. Найти значения функции X=Y^2 *(Y^2 -Y)/Y при Y={N,N*1,N*2,…,N*M} (N<M).

7.3.Цикл с параметром

Оператор цикла с параметром имеет 2 модификации:

For K:=N1 to N2 do M; (с возрастанием параметра)

For K:=N1 downto N2 do M; (с убыванием параметра)



For – для;
To – до;
Downto – уменьшая до;
K – переменная порядкового типа (Integer, Char, Boolean), называется параметром цикла;
N1, N2 – выражение, имеющее значение того же типа, что и параметр K;
M – любой оператор в единственном числе;
Если по смыслу задачи надо повторять группу операторов, то их надо превратить в один составной оператор (заключить в операторные скобки):
Выполняется:

1. Вычисляется значение N1, N2;
2. K:=N1;
3. Проверяется условие:

- K<=N2 (to)
- K>=N2 (downto)

4. Если условие не выполнено, то выполнение оператора цикла заканчивается. Иначе выполняется оператор M;
5. K – получает приращение:

- K:=succ(K) (to);
- K:=pred(K) (downto);

6. Переход к пункту 3;

В турбо Паскале в отличие от Бейсика цикл с параметром реализован как цикл «пока» и значит оператор M может ни разу не выполнится.

Задача 3.
Вывести коды всех строчных символов латинского алфавита.
Решение:

```
Program Ci3;
  Var S:Char;
   Begin
     For S:=’a’ to ’z’ do
       Writeln(’У символа - ’,S,’ код ’,Ord(S));
   End.
```

Задача 4.
Вывести на экран цифры в обратной последовательности
Решение:

```
Program Ci4;
  Var R:Integer;
  Begin
    For R:=9 downto 1 do Write(R,' ':4);
  End.
```

Задача 5

Вывести на экран 10 случайных чисел в диапазоне от 0 до 100.

Решение:

```
Program Ci4;
   Const X=100;
   Var Y,R:Integer;
   Begin
   Randomize; {инициализирует генератор случайных чисел}
For R:=1 to 10 do
       Begin
        Y:=Random(X);
        Writeln(R,'=',Y);
       End;
   End.
```

8.Массивы (структурный тип)

8.1.Таблицы и табличные величины

При решении задач человек очень часто пользуется таблицами. При записи исходных данных, получении справочной информации и т.п. Таблицы бывают разные, но наиболее часто встречающиеся линейные и прямоугольные таблицы. Каждая таблица имеет свое название. Значения, образующие линейную таблицу, располагаются при записи на бумаге в строку или в столбец. Каждому значению или элементу таблицы, соответствует его порядковый номер (индекс), и наоборот: стоит задать порядковый номер (индекс), и сразу ясно, о каком элементе таблицы идет речь.

Пример:

На метеостанции каждый час измеряется температура воздуха и значения измерений за сутки записываются в таблицу:

Temp - название таблицы.

Время измерения, ч	1	2	3	4	5	6	……	23	24
Температура, С	12	13	11	15	10	12		11	11

Эта линейная таблица содержит 24 элемента, занумерованные от 1 до 24. Например, второй элемент таблицы имеет значение – 13, а пятый – значение 10.

Для прямоугольной таблицы должны быть указаны границы индексов, как по вертикали, так и по горизонтали (строки и столбцы). Каждому значению или элементу прямоугольной таблицы соответствует определенный индекс строки и столбца.

Элементы таблицы могут быть любого типа (числовые, символьные, строковые и т.д.).

Пример:
Составим таблицу размещения пассажиров в первых 6-ти вагонах на местах с 15 по 19 (элементами таблицы будут имена пассажиров).
Vagon

	1 вагон	2 вагон	3 вагон	4 вагон	5 вагон	6 вагон
15	Катя	Наташа	Юля	Римма	Рома	Лиля
16	Олег	Марат	Ирек	Адель	Петя	Эльвина
17	Вася	Таня	Люба	Костя	Луиза	Лиза
18	Лена	Иля	Наташа	Артур	Рустем	Руслан
19	Коля	Мансур	Антон	Саша	Раиль	Марат

Например, элемент таблицы, стоящий в строке с индексом 17 и в столбце – 1 вагон, имеет значение – Вася.

8.2. Описание массива (в разделе Var)

В Турбо Паскале существует возможность работы с таблицами, они имеют название - массивы.
Массив – это совокупность элементов одного типа, объединенных под общим именем. Каждый элемент массива имеет свой индекс (порядковый номер), который определяет его относительную позицию. Число элементов массива задается при описании и в дальнейшем не изменяется.
Массив объявляется в следующей форме:
A:Array[B1,B2,…,Bn] of M;

Array – массив;
Of – из;
Здесь - имя массива (правильный идентификатор);
B1, B2,…,Bn – списки индексных типов, их можно задавать, указав любой перечисляемый тип (кроме Longint) Количество списков (n) определяет размерность массива, они разделяются запятыми и заключаются в квадратные скобки;
M - тип элементов, любой тип Турбо Паскаля;
Доступ к каждому элементу массива в программе осуществляется с помощью индекса - целого числа (точнее, выражения порядкового типа). При упоминании в программе любого элемента массива сразу за именем массива должен следовать индекс элемента в квадратных скобках.

8.3.Одномерные массивы

Рассмотрим работу с одномерными массивами (в алгебре они называются векторами):
Дадим таблице имя - Tab

1	2	3	4	5	6	7	8
15	20	33	13	11	4	12	8

Каждое число в таблице имеет тип Integer. Это - тип злементов (M).
Индексы данной таблицы изменяются от 1 до 8 . У одномерных массивов один список (B1).
Таким образом, описание этого массива (в разделе Var) выглядит так:

Tab:Array[1..8] of Integer;

Задав конкретные значения индексов, можно выбрать определенный элемент массива. Например, оператор:

N:=A[5];

Присвоит переменной N значение элемента массива, имеющего индекс 5, т.е. число 11.

Задача 1.

Задан одномерный массив из N чисел. Изменить все элементы массива, увеличив их на единицу.

Работу с массивом можно разделить на 4 этапа:

1. Описание массива (выделить место в памяти компьютера для данного массива);
2. Ввод элементов массива (с клавиатуры; с помощью оператора присваивания; с помощью функции Random);
3. Работа с массивом (изменить элементы массива в соответствии с заданием);
4. Вывод массива (на экран или принтер);

Последовательный доступ к каждому элементу массива удобно осуществлять в цикле.

Решение:

```
Program Mass1;
{1}Var Tab:Array[1..10] of Integer;
       I,N:Integer;
   Begin
     Writeln('Введите число элементов массива (N<=10)');
      Readln(N);
   {2}For I:=1 to N do   Readln(Tab[I]);{ввод элементов с клавиатуры}
   {3}For I:=1 to N do  Tab[I]:= Tab[I]+1;
   {4}For I:=1 to N do  Writeln('Tab[',I,']=',Tab[I]);
   End.
```

Задачи:

1. Задан одномерный массив из N чисел. Изменить значения элементов массива: а) все уменьшить на 100; б)если элемент больше 100 – на 100; в) если элемент отрицательный – на квадрат.
2. Задан одномерный массив из N чисел. Сосчитать количество элементов массива:
   а) положительных; б) отрицательных; в) нулевых.
3. Заполнить одномерный массив элементами, значения которых равны их удвоенным порядковым номерам.
4. Задан одномерный массив из N чисел. Найти те элементы, значения которых совпадают с их индексом.
5. Задан одномерный массив из N чисел. Сосчитать: сумму, произведение и среднее арифметическое всех элементов массива.
6. Задан одномерный массив из N чисел. Определить сумму положительных и отрицательных элементов массива.
7. Задан одномерный массив A из N чисел. Создать новый массив путем деления всех элементов массива A на число X.

8.4. Нахождение максимального элемента массива

Задача 2.

Задан одномерный массив из N чисел. Найти максимальный элемент массива.

Решение:

```
Program Mass2;
 Const X=100;
 Var A:Array[1..10] of Integer;
       I,N,Max:Integer;
 Begin
    Writeln('Введите число элементов массива (N<=10)');
    Readln(N);
    Randomize;
    For I:=1 to N do A[I]:=Random(X);{заполнение массива случайными числами}
    Max:=A[1];
    For I:=2 to N do
      If A[I]> Max Then Max:= A[I];
    For I:=1 to N do  Writeln('A[',I,']=',A[I]);
    Writeln('Max=',Max);
    End.
```

Задачи:

1. Задан одномерный массив из N чисел. Найти минимальный элемент массива.
2. Задан одномерный массив из N чисел. Найти разницу между наибольшим и наименьшим элементами массива.
3. Задан одномерный массив из N чисел. Проверить, есть ли в массиве отрицательные элементы и если есть, то найти среди них наименьший.

8.4.Сортировка одномерного массива

Задача 3.

Задан одномерный массив из N чисел. Отсортировать все элементы массива по возрастанию.

Решение:

```
Program Mass3;
  Var A:Array[1..10] of Integer;
       I,P,N:Integer;
       F:boolean;
  Begin
    Writeln('Введите число элементов массива (N<=10)');
    Readln(N);
    For I:=1 to N do   Readln(A[I]);
    F:=False;
    While F=False do
      Begin
        F:=True;
        For I:=1 to N-1 do
          If A[I]>A[I+1] Then
                          Begin
```

```
                    P:=A[I];
                    A[I]:=A[I+1];
                    A[I+1]:=P;
                    F:=False;
                  End;
    End;
  For I:=1 to N do Writeln('A[',I,']=',A[I]);
 End.
```

Задачи:

1. Задан одномерный массив из N строк, имеющих вид фамилий. Отсортировать все элементы массива по алфавиту.
2. В массиве каждый элемент равен 0, 1 или 2. Переставить элементы массива так, чтобы вначале массива расположились все нули, затем все единицы и, наконец все двойки.
3. Задан одномерный массив из N чисел. Переставить элементы массива так, чтобы отрицательные элементы предшествовали всем неотрицательным.
4. Задан одномерный массив из N чисел и число B. Упорядочить массив по возрастанию и поместить переменную B в соответствующее место массива.

8.4.Решение задач, используя одномерные массивы.

1. Задан одномерный массив из N чисел. Составить новый массив, состоящий из тех же чисел, но идущий в обратном порядке.
2. Задан одномерный массив из N чисел и число K. Напечатать «Да», если K совпадает хотя бы с одним из элементов массива и «Нет» в противном случае.
3. Задан одномерный массив из N чисел. Найти количество элементов массива, больших среднего арифметического всех его элементов.
4. Задан одномерный массив из N чисел Определить в массиве число соседств из двух чисел одного знака.
5. Задан одномерный массив из N чисел Найти сумму произведений всех троек соседних чисел.
6. Задан одномерный массив из N чисел. Подсчитать наибольшее число одинаковых элементов, идущих в массиве подряд.
7. Задан одномерный массив из N чисел. Заменить все элементы массива на сумму предыдущего и последующего их значений.
8. Задан одномерный массив из N чисел. Подсчитать количество неповторяющихся элементов массива.
9. Задан одномерный массив из N чисел. Составить массив У, где Y[I]:=Min(X[1],X[2],…,X[N]).

8.4.Двумерные массивы

Рассмотрим работу с двумерными массивами (в алгебре они называются матрицами):

Дадим таблице имя - Tab

	1	2	3	4
1	20	33	15	11
2	11	-1	4	7
3	5	0	3	6

Каждое число в таблице имеет тип Integer. Это - тип злементов (M).
Индексы данной таблицы изменяются от 1 до 8 . У двумерных массивов два списка (B1,B2).
Таким образом, описание этого массива (в разделе Var) выглядит так : **Tab:Array[1..3,1..4] of Integer;**
Задав конкретные значения индексов, можно выбрать определенный элемент массива. Например оператор:

N:=A[1,3];

Присвоит переменной N значение элемента массива, имеющего индекс строки 1, индекс столбца 3, т.е. число 15.
Задача 4.
Задан массив из целых чисел, размерностьюN*M (N-строки, M-столбцы). Вывести на экран:
а) N-ю строку; б) M-й столбец; в) весь массив.
Решение:

```
Program Mass4;
  Var Tab:Array[1..10,1..10] of Integer;
      N,M,X,Y:Integer;
  Begin
    Writeln('Введите количество строк массива (N<=10)');
    Readln(N);
    Writeln('Введите количество столбцов массива (M<=10)');
    Readln(M);
    For X:=1 to N do
      For Y:=1 to M do Readln(Tab[X,Y]);
    Writeln('Вывод ',N,' строки');
    For X:=1 to M do Write('Tab[',N,',',X,']=',Tab[N,X],' ':2);
    Writeln;
    Writeln('Вывод ',M,' столбца');
    For X:=1 to N do Writeln('Tab[',X,',',M,']=',Tab[X,M]);
    Writeln('Вывод массива');
    For X:=1 to N do
     Begin
      For Y:=1 to M do Write('Tab[',X,',',Y,']=',Tab[X,Y],' ':2);
      Writeln;
     End;
```

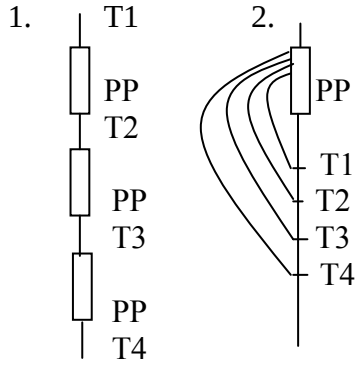
Задача 5.
Задан массив из строк, размерностьюN*M (N-строки, M-столбцы). Отсортировать все элементы массива по возрастанию.
Решение:

```
Program Mass5;
  Var A:Array[1..10,1..10] of String;
      I,N,M,X,Y:Integer;
      P:String;
      F:boolean;
  Begin
```

```
{Ввести двумерный массив}
 F:=False;
   While F=False do
     Begin
       F:=True;
       For X:=1 to N  do
         For Y:=1 to M-1 do
           Begin
             If A[X,Y]>A[X,Y+1] Then
                                    Begin
                                    {Поменять местами 2 соседних элемента строки}
                                    End;
          If (X<N)and(Y=(M-1))and(A[X,Y+1]>A[X+1,1])Then
                                                          Begin
                                                            P:=A[X,Y+1];
                                                            A[X,Y+1]:=A[X+1,1];
                                                            A[X+1,1]:=P;
                                                            F:=False;
                                                          End;
      End;
    End;
   {Вывести отсортированный массив}
End.
```

9.Подпрограммы.

При решении многих задач возникает необходимость проведения одних и тех же вычислений на различных этапах решения задачи и при различных значениях исходных данных.



При составлении программы по первому алгоритму приходится задавать одну и ту же группу операторов (PP) для каждого из повторяющихся фрагментов.

Для сокращения текста программы в Паскале введено понятие подпрограммы, при этом повторяющаяся группа операторов записывается отдельно (второй алгоритм) один раз по отношению к некоторым формальным параметрам, а в

соответствующих местах программы помещается лишь обращение к подпрограмме с указанием нужных в данный момент фактических параметров.

Использование подпрограмм позволяет применять принципы структурного программирования, когда исходная задача разбивается на группы простых подзадач, каждая подзадача программируется отдельно, а программа исходной задачи составляется только из обращений к программам подзадач. Кроме того, использование подзадач позволяет вводить в новую программу, программы составленные ранее.

В Паскале подпрограммы реализуются 2-х видов - процедуры и функции. И те, и другие вводятся в программу с помощью описания. Для этого существует специальный раздел программы

9.1.Описание процедур

Структура описания процедур аналогична самой программе. Она состоит из заголовка процедуры и блока процедуры. Блок процедуры включает в себя те же 6 разделов, что и блок программы.

Заголовок процедуры:

Procedure Xxx[(P1:T1;P2:T2;…Pn:Tm)];

Xxx - имя процедуры (любой идентификатор);

P1,P2,…,Pn - формальные параметры;

T1,T2,…,Tm – типы формальных параметров

Формальные параметры – это перечень имен для обозначения исходных данных и результатов работы процедуры. Формальные параметры нужны для того, чтобы указать, куда должны быть подставлены фактические параметры при обращении к процедуре. Формальные параметры в программе не описываются.

Формальные параметры могут отсутствовать и, в этом случае, процедура описывается без параметров.

Обращение к процедуре:

Чтобы исполнить процедуру, в нужном месте программы записывается оператор обращения к ней, который имеет вид:

Xxx[(B1,B2,…,Bn)];

B1,B2,…,Bn – список фактических параметров. Он отсутствует, если процедура описывалась без параметров.

При вызове процедуры устанавливается взаимное однозначное соответствие между фактическими и формальными параметрами, а затем управление передается процедуре.

Соответствие между фактическими и формальными параметрами:

1. Число фактических параметров должно быть равно числу формальных параметров;
2. Соответствующие фактические и формальные параметры должны совпадать по порядку следования и по типу.

9.3.Формальные параметры – переменные

Перед ними ставится слово Var.

Для формального параметра – переменной используется именно та ячейка, которая содержит соответствующий фактический параметр и поэтому результаты работы процедуры могут быть переданы в программу только через параметр –

переменную. В фактических параметрах – переменных допускается использовать только переменные.

Пример 1:

```
:
Procedure Xxx(Var A,B,C:Integer; Var D:Real);{A,B,C,D – формальные параметры – переменные; }
:
Xxx(M,N,X,Y);{M,N,X,Y – переменные}
```

Рассмотрим на примере принцип работы параметров-переменных:

Пример 2:

```
Program Pr1;
Var A,B:Integer;
Procedure Zamena(Var Y,X:Integer);
  Begin
    X:=X+1;Y:=Y+10;
  End;
Begin
  A:=0;B:=0;
 Zamena(A,B);
  Writeln(A,B);
End.
```

Результат: 1,10 {Результат работы процедуры передается в программу – переменные A и B изменили свое значение}

Задача.

Составить программу вычисления максимального из 4-х чисел, используя процедуру вычисления максимального из 2-х чисел.

Решение:

```
Program Pr3;
  Var A,B,C,D,M1,M2,M3:Integer;
   Procedure Max(Var X,Y,M:Integer);
     Begin
       If X>Y Then M:=X Else M:=Y;
     End;
   Begin
      Readln(A,B,C,D);
       Max(A,B,M1);
       Max(C,D,M2);
       Max(M1,M2,M3);
       Writeln('Max=',M3);
   End.
```

Задачи:

1. Правильно ли описана процедура и обращение к ней (если есть ошибки, то исправить):

```
Procedure Str(Var (A,B,C): Real;)
  C:=A+B+C;
:
```

```
Str(X,Y,Z,S);
:
```

2. В чем отличие процедур:

   а)
   ```
   Procedure Str1;
     Begin
       P:=(A+B+C)/2;
       S:=sqrt(P*(P-A)*(P-B)*(P-C));
     End;
   ```

   в)
   ```
   Procedure Str1(Var A,B,C,S:Real);
     Begin
       P:=(A+B+C)/2;
       S:=sqrt(P*(P-A)*(P-B)*(P-C));
     End;
   ```

   Используя эти процедуры вычислить площадь выпуклого четырехугольника, заданного длинами четырех сторон и диагональю.

3. Для каждого из приведенных ниже описаний процедур сформулировать назначение соответствующей процедуры:

   а)
   ```
   Procedure M(Var X,Y,M:Real);
     Begin M:=X+Y; End;
   ```

   б)
   ```
   Procedure M1(Var X,Y,Z,T:Real);
     Begin Z:=X+Y; T:=X*Y; End;
   ```

   в)
   ```
   Procedure M2(Var X,Y,R:Real; Var P:Integer);
     Begin If sqr(X)+sqr(Y)<=sqr(R) Then P:=1 Else P:=0; End;
   ```

4. Процедура описана следующим образом: `Procedure M2(Var X,Y:Integer);` Допустимо ли обращение к процедуре, имеющее вид M2(A,B-1);
5. Даны числа X,Y,Z. Используя процедуру нахождения максимального из двух чисел, вычислить U=(Max(X,Y)+Max(X,Z))/(Max(Y,Z));
6. Одинаковы ли последствия обращения к процедурам:

   а)
   ```
   Procedure P;
     Begin X:=X+Y; Y:=X-Y; End;
   ```

   б)
   ```
   Procedure P;
     Begin Y:=X-Y; X:=X+Y; End;
   ```

9.4.Формальные параметры – значения

Слово Var не ставится.

Используются только для передачи исходных данных в процедуру. В ходе выполнения процедуры эти значения изменяться не могут и, следовательно, параметры – значения не могут выполнять роль результата работы. Все формальные параметры, кроме тех, которые присваивают результаты работы, рекомендуется объявлять параметрами – значениями. В фактических параметрах – значениях допускается ставить выражения.

Пример:

```
:
Procedure Xxx(A,B,C:Integer; Var D:Real);{A,B,C – формальные параметры – значения; D – формальный параметр – переменная; }
:
Xxx(M,N,X,Y);{M,N,X – могут быть выражениями; Y – только переменная}
```

Рассмотрим на примерах различия при использовании параметров-значений и параметров-переменных :

Пример1.

```
Program A1;
  Var X:Integer;
   Procedure Zamena(Y:Integer);
     Begin
       Y:=1;
     End;
   Begin
     X:=0; Zamena(X);
     Writeln('X=',X);
   End.
```

Результат: X=0

Пример2.

```
Program Pr2;
  Var X:Integer;
   Procedure Zamena(Var Y:Integer);
     Begin
       Y:=1;
     End;
   Begin
     X:=0; Zamena(X);
     Writeln('X=',X);
   End.
```

Результат: X=1

Задача:

Составить программу обращения к процедуре вычисления натуральной степени числа A ($Z=A^k$)

```
Program Proc2;
  Var A,K,Z:Integer;
   Procedure Nstep(X:Real;N:Integer;Var Y:Real);{Y=X^n}
      Var I:Integer;
      Begin
         Y:=1;
         For I:=1 to N do
           Y:=Y*X;
      End;
   Begin
     Readln(A,K);
     Nstep(A,K,Z);
     Writeln('Z=',Z);
   End.
```

Переменная I, описанная в процедуре, называется локальной по отношению к процедуре. Локальная – местная, имеющая местное значение. Как только процедура выполнится, значение локальной переменной I забудется.

Задачи

1. В программе описана процедура P с формальными параметрами X,Y и процедура Q с формальными параметрами S,T. Среди операторов программы встречаются операторы процедуры P(1,A) и Q(B,D+F). Какие из формальных параметров процедур P и Q заведомо являются параметрами-значениями?
2. Даны числа A, B, C, D. Найти значение выражения K=M*N+K, где M=Max(A,B); N=Max(B,C,D); K=Max(A,B,C,D);. Использовать процедуру нахождения максимального из 2-х чисел.
3. Даны натуральные числа A, B, C, D, X. Найти значение выражения $M=(X*A)_A+B_C*(D*X)_C$. Использовать процедуру вычисления натуральной степени числа.
4. Составить программу обращения к процедуре вычисления целой степени числа A ($Z=A^k$) (K-целое число)
5. Даны натуральные числа K, M. Требуется вывести на экран рамку из звездочек высота которой – K строк, ширина – M знаковых позиций. Описать процедуру Zvezda(S,N), обращение к которой дает вывод данной рамки.

9.5.Описание функций

Функция – это подпрограмма, результатом выполнения которой является одно единственное значение. Это значение присваивается имени функции. Таким образом, функция не требует введения формальных параметров, играющих роль результата, так как эту роль играет имя самой функции.

Функция состоит из заголовка и блока:

Заголовок функции:

Function Xxx(P1:T1;P2:T2;…Pn:Tm):Q;

Xxx - имя функции (любой идентификатор);

P1,P2,…,Pn - формальные параметры;

T1,T2,…,Tm – типы формальных параметров;

Q – тип значений, которые способна принимать функция в результате ее выполнения;

Особенности раздела операторов функции:

Раздел операторов функции должен содержать оператор присваивания, в котором слева помещено имя функции.

Обращение к функции:

Обращение к функции не является оператором, оно входит в состав выражения (например, в правой части оператора присваивания).

Z:=Xxx[(B1,B2,…,Bn)];

Задача.

Составить программу вычисления максимального из 4-х чисел, используя функцию вычисления максимального из 2-х чисел.

```
Program func1;
  Var A,B,C,D,M1,M2,M3:Integer;
```

```
Function Max(X,Y:Integer):Integer;
  Begin
    If X>Y Then Max:=X Else Max:=Y;
  End;
Begin
   Readln(A,B,C,B);
    m1:=Max(A,B);
    m2:=Max(C,D);
    m3:=Max(M1,M2);
    Writeln('Max=',M3);
End.
```

Задачи:

Из параграфа 9.4 решить задачи №№ 2,3,4,5, используя обращение к функции.

10.Стандартные модули

Модули можно использовать для создания библиотек стандартных подпрограмм и данных. В Турбо Паскале в настоящее время имеется большое количество стандартных подпрограмм и данных, объединенных в несколько стандартных модулей:

10.1.Модуль System

Модуль автоматически подключается к любой программе

В модуль System входят следующие подпрограммы:

- арифметические функции; {рассмотрены в параграфе 3.3}
- функции для величин порядкового типа;
- функции преобразования типов;
- процедуры и функции работы со строками;
- процедуры и функции работы с файлами;

10.2.Функции работы со строками

Function Concat(S1[S2,...,Sn]:String):String;

Объединяет несколько строк в одну.

S1,S2,...,Sn - объединяемые строки;

Пример1.

```
Program Ct1;
  Var S,S1,S2:String;
  Begin
    Writeln('Введите 1 строку');
    Readln(S1);
    Writeln('Введите 2 строку');
    Readln(S2);
    S:=Concat(S1,S2);
    Writeln('S= ',S);
  End.
```

Function Copy(S:String;I,C:Integer):String;

Создает подстроку строки S
S - исходная строка
I - номер первого выделяемого символа строки
C - число выделяемых символов
Пример2.

```
Program Ct2;
  Var S,S1:String;
  Begin
     Writeln('Введите  строку');
     Readln(S1);
     S:=Copy(S1,3,4);{копирует 4 символа, начиная с 3-го}
     Writeln('S= ',S);
  End.
```

Function Length(S:String):Integer;

Возвращает текущий размер строки

S - строка, у которой определяют размер

Пример3.

```
Program Ct3;
  Var S:String;
        Y:Integer;
  Begin
     Writeln('Введите  строку');
     Readln(S);
     Y:= Length(S);
     Writeln('Количество символов в строке = ',Y);
  End.
```

Function Pos (S1,S2:String):Byte;

Поиск последовательности S1 в строке S2 (результат равен номеру первого символа строки S2, с которого начинается искомая последовательность, или 0, если такой последовательности в строке нет)

S1 – искомая последовательность;

S2– строка, в которой ищется последовательность;

Пример4.

```
Program Ct4;
  Var S2,S1:String;
        Y:Integer;
  Begin
     Writeln('Введите  строку');
     Readln(S2);
```

```
    Writeln('Введите  искомые символы');
    Readln(s1);
    y:= Pos(S1,S2);
    Writeln('Искомая последовательность начинается с символа ',Y);
  End.
```

10.3.Процедуры работы со строками

Procedure Delete(Var S:String;I,C:Integer);

Удаляет подстроку из строки S
S - исходная строка
I - номер первого удаляемого символа
C - число удаляемых символов
Пример5.

```
Program Ct5;
  Var S:String;
   Begin
    Writeln('Введите  строку');
    Readln(S);
    Delete(S,5,3);{удаляет из строки 3 символа, начиная с 5-го}
    Writeln(S= ',S);
  End.
```

Procedure Insert(C:String;Var S:String;I:Integer);

Помещает подстроку C в строку S
S - исходная строка
C - подстрока, помещаемая в строку
I - номер позиции исходной строки, начиная с которой, помещается подстрока
Пример6.

```
Program Ct6;
  Var C,S:String;
   Begin
    Writeln('Введите  исходную строку');
    Readln(S);
    Writeln('Введите подстроку');
    Readln(C);
    Insert(C,S,5);{в строку S помещается подстрока C, раздвигая ее, с 5 позиции}
    Writeln('S= ',S);
  End.
```

Procedure Str(X[:M[:N]];Var S:String);

Преобразует число в последовательность символов.
X - выражение вещественного или целого типа
S - строка, в которую записывается символьное представление числа

M,N - формат вывода
Пример7.

```
Program Ct7;
  Var S:String;
        Y:Integer;
   Begin
     Writeln(’Введите число’);
     Readln(Y);
     Str(Y,S);
     Writeln(строка, имеющая вид числа = ’,S);
  End.
```

Procedure Val (S:String;Var X; Var C:Integer);

Преобразует символьное представление числа в число.
S – строка с символьным представление числа;
X -переменная вещественного или целого типа для записи числа;
C – номер неправильного символа (0 – если изображение числа правильное)

Пример8.

```
Program Ct8;
  Var S:String;
        Y:Integer;
   Begin
     Writeln(’Введите строку из цифр’);
     Readln(S);
     Val(S,Y,C);
     Writeln(’число = ’,Y*10);{можно производить арифметические действия}
  End.
```

Задача: Составить программу, которая подсчитывает количество слов в предложении.
Пример8.

```
Program Ct8;
  Var S:String;
        K,Y:Integer;
   Begin
     Writeln(’Введите строку’);
     Readln(S);K:=1;
     For Y:=1 to Length(S) do
       If Copy(S,Y,1)=’ ’ Then K:=K+1;
     Writeln(’В предложении ’,K,’ слов);
   End.
```

Задачи для самостоятельного решения
Дана строка, имеющая вид предложения.
Задача1_1. Вставит в предложение с 3-го символа слово «кит».
Задача2_1. Удалить из предложения 3 символа, начиная с 6-го.

Задача3_1. Скопировать из предложения 3 символа, начиная со 2-го.
Задача1_2. Подсчитать сколько раз встречается буква «м», предлог «не» в предложении.
Задача2_2. Выяснить, есть ли в предложении хотя бы одна пара одинаковых символов.
Задача3_2. Удалить из предложения все пробелы.
Задача1_3. Вставить в предложение пробелы после каждой буквы «а»
Задача2_3. Проверить, есть ли в предложении запятые
Задача3_3.