Оглавление

[1.Основные понятия Паскаля. 2](#_Toc344290208)

[2.Структура программы на Паскале. 3](#_Toc344290222)

[3.Правила написания программы. 3](#_Toc344290226)

[4.Правила записи выражений на Паскале 4](#_Toc344290227)

[5.Типы данных: REAL. 4](#_Toc344290228)

[6.Типы данных: INTEGER. 5](#_Toc344290230)

[7. Типы данных: BOOLEAN. 5](#_Toc344290232)

[8.Типы данных: CHAR. 5](#_Toc344290233)

[9.Типы вычислительных процессов. 5](#_Toc344290235)

[10.Оператор присваивания. 7](#_Toc344290239)

[11.Оператор ввода. 7](#_Toc344290240)

[12.Оператор вывода. 7](#_Toc344290241)

[13.Оператор условного перехода 7](#_Toc344290242)

[14.Оператор выбора (CASE). 7](#_Toc344290243)

[15.Оператор цикла с параметром For. 7](#_Toc344290244)

[16.Оператор цикла с предусловием. 8](#_Toc344290245)

[17.Оператор цикла с постусловием. 8](#_Toc344290246)

[18.Вложенные циклы. 8](#_Toc344290247)

[19. Массивы 8](#_Toc344290248)

[20. Множества 9](#_Toc344290249)

[21. Динамические переменные, указатели, процедуры 10](#_Toc344290250)

[22.Процедуры 11](#_Toc344290251)

[23.ФУНКЦИИ 11](#_Toc344290252)

[24.Обращение к подпрограммам 12](#_Toc344290253)

[25. Параметры переменные 12](#_Toc344290254)

[26.Доступ к компонентам записи. Понятие записи. 12](#_Toc344290255)

[27. Файлы 13](#_Toc344290256)

[28.Текстовые файлы 13](#_Toc344290257)

[29. Процедуры при работе с текстовыми файлами 13](#_Toc344290258)

[30. Типизированные файлы 13](#_Toc344290259)

[31.Модуль. 19](#_Toc344290260)

[32.Графика 19](#_Toc344290261)

[33. Delphi. (компонент TEdit). 20](#_Toc344290265)

[34. Компонент TString Greed. 20](#_Toc344290266)

[35.Компонент Tlabel 21](#_Toc344290267)

[36.Компонент TMainMenu 21](#_Toc344290268)

[37.Компонент TPopupMenu 21](#_Toc344290269)

[38. TMemo 21](#_Toc344290270)

[39.TButton 22](#_Toc344290271)

[40.MDI-приложения. 22](#_Toc344290272)

# 1.Основные понятия Паскаля.

СЛОВА:

А)Служебные:

And, array, absolute, begin, case, constructor,

Destructor, div, do, down to, else, end, external,

File, for, function, go to, if, implementation, in,

Label, mod, not, of, or, procedure, program,

Record, repeat, string, then to, type, unit, uses, var,

Virtual, while, with.

Правила:

Служебные слова не искажаются.

Б)Идентификаторы

* Константы;
* Имена переменных;
* Имена процедур и функций;

Правила:

* 1. Состоит из букв латинского алфавита, цифр, знака подчеркивания;
  2. Не может начинаться с цифры;
  3. Не может совпадать не с одним из зарезервированных слов;

ПРИМЕР:

Площадь треугольника: S\_TR

# 2.Структура программы на Паскале.

Program xx; Блок описаний:

Var x: real; х -дробное знач.

Const g = 9.8; ускорение своб.пад.

Label 1;

Procedure Vvod (n,m:Bute;Var: x:mass);

Function Fact(n:Bute):Word;

Type mas(

Begin

Операторный блок

End.

# 3.Правила написания программы.

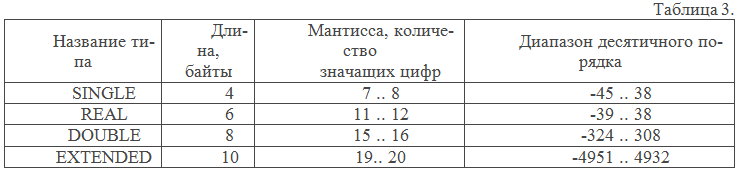
* При написании программы не следует создавать большие программные модули (оформлять в виде подпрограмм)
* Не следует создавать большие текстовые файлы
* Программы пишутся в строке с отступом вправо-влево по отношению к другим операторам
* Не следует в одной строке объединять несколько операторов, за исключением простейших (А:=0; B:=0; P:=0; и т.п.)
* Делайте пробелы для улучшения читаемости программы. Широкое использование пробелов существенно облегчает чтение программы
* Идентификаторы программы должны быть выразительными, т.е. обозначать сущность или сокращенное наименование параметров задачи
* Рекомендуется широко использовать написание комментариев. Каждая процедура, функция должна иметь комментарии, в которых сообщается ее назначение
* Следует использовать при написании программы возможность расцвечивания разными цветами различных её элементов
* Правильное использование скобок существенно улучшает читаемость программы. Основное правило такое: в сомнительных случаях всегда ставить скобки
* Во избежание неоднозначностей не следует локальным и глобальным параметрам давать одинаковые имена
* Последовательности операторов , выполняющих какое-то законченное действие, целесообразно отделять от предыдущих и последующих операторов пустыми строками

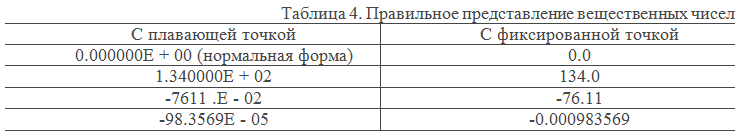
# 4.Правила записи выражений на Паскале

|  |  |  |
| --- | --- | --- |
| Математическая запись | Запись на Паскале | комментарии |
| π  Abs(x)  Arctg  Cos  E^x  Ln(x) | PI  Abs  Arctan(x)  Cos(x)  Exp(x)  Ln(x)  Random(x)  Trunc(x)  Sin(x)  Sqr(x)  Sqrt(x) | π=3.1415926536Е+00  модуль аргумента  Арктангенс x(радианы)  Косинус  Экспонента  Натуральный логарифм  Псевдослучайное число в интервале[0,X)  Ближайшее целое, не превышающее x по модулю  Синус  Квадрат x  Корень квадратный из x |

# 5.Типы данных: REAL.

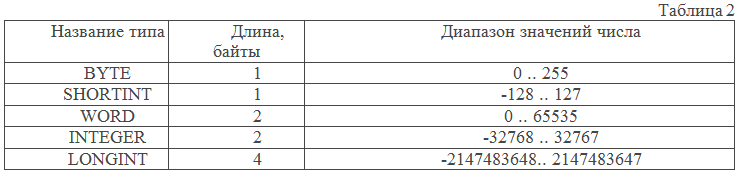
REAL- вещественный тип REAL- вещественный тип.





# 6.Типы данных: INTEGER.

INTEGER- целый тип.



# 7. Типы данных: BOOLEAN.

BOOLEAN-логический тип.

|  |  |  |
| --- | --- | --- |
| идентификатор | длина | Диапазон(множество)значений |
| BOOLEAN | 1 | True, false |

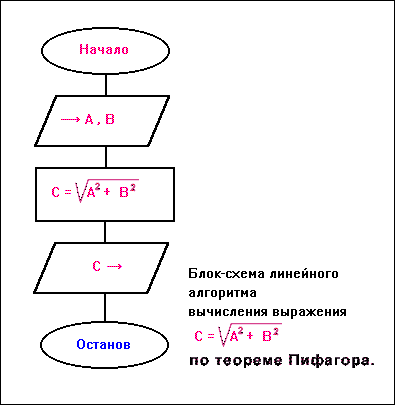
# 8.Типы данных: CHAR.

Char - символьный тип.

|  |  |  |
| --- | --- | --- |
| идентификатор | длина | Диапазон(множество)значений |
| CHAR | 1 | Все символы ASCLL |

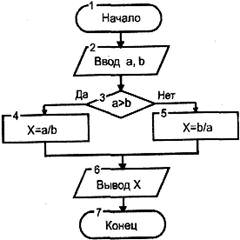
# 9.Типы вычислительных процессов.

А)Линейный

* Действия выполняются друг за другом;
* Линии идут сверху вниз;
* Терминатор, процесс, данные.

Б) Разветвляющийся

В процессе решения задач возникает ситуация, когда нужно выбрать дальнейший путь(направление ветвей).

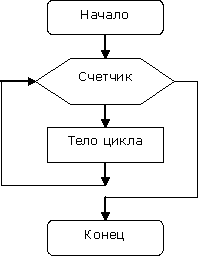
Содержит несколько ветвей.

В) Циклический

Цикл - многократное повторение последовательностей действий.

Цикл состоит из :

* Начальное значение (одно);
* Тело цикла (основные действия);
* Новое значение (повтор);
* Нужно ли повторять цикл?



# 10.Оператор присваивания.

<переменная>:=<выражение>;

Золотое правило: обязательное соответствие типов

## 11.Оператор ввода.

Read(список ввода);

Readln ( список ввода);

## 12.Оператор вывода.

Write(список вывода);

Write( список вывода);

13.Оператор условного перехода.

**Условный оператор.**

If<условие>then<оператор>; да

If<условие>then <оператор1>

Else<оператор2>;

Оператор 1

да

Опер-р 2

оператор

нет

# 14.Оператор выбора (CASE).

Обобщенная форма оператора if используется , если в условии больше 3-х.

CASE <выражение> of

<значение1>:=<оператор1>;

<значение2>:=<оператор2>;

…..

End;

## 

## 15.Оператор цикла с параметром For.

For <пер.>:=<начальное значение> to <конечное значение> DO

Down to Begin

End;

## 16.Оператор цикла с предусловием.

Начальное значение

While<условие> do (пока условие верно!)

Begin X:=x+шаг;

End;

## 17.Оператор цикла с постусловием.

Repeat

Начальное значение

Repeat

(шаг)x:=x+h; ТЕЛО ЦИКЛА

Until<условие>;

# 18.Вложенные циклы.

Внешний цикл это, когда в теле, содержит цикл. Внутренний, который соединяется внутри цикла.

Тела другого цикла внешний нач. за и после.

Работа при начал. Внешнего цикла перебирает все

**I=j**

**j=(1)n**

Работа при начал. Внешнего цикла перебирает все значения цикла. Далее при новом значении пар. внешнего цикла. И до так до тех пор пока не переберет все значения параметра внешнего цикла.

## 19. Массивы

Одномерные:

Ввод:

Writeln('введите размерность');

Readln(n);

For i:=1 to n DO

BEGIN

Writeln('введите значение',i:3,’элемента’);

Readln(a[i]);

END;

**Сумма:**

S:=0;

FOR I:=1 to n DO

S:=S+a[i];

WRITELN(‘S= ’,S:7);

**Произведение:**

PR:=1;

FOR I:=1 to n DO

PR:=PR\*a[i];

WRITELN('PR=',PR:8);

**Максимум:**

max:=-MaxInt;

FOR i:=1 to n DO

IF a[i]>max then

Begin

max :=a[i];

imax:=i;

end;

WRITELN(‘max=’,max:4);

WRITELN(‘инд=’,imax:);

**Минимум**

min :=MaxInt;

FOR i:=1 to n DO

IF a[i]<min then

Begin

min :=a[i];

imin :=i;

end;

writeln(‘min=’,min:4);

writeln(‘инд=’,min:3);

**Вывод**

FOR i:=1 to n DO

WRITE(a[i]:6:2);

WRITELN;

**Двумерные**

**B: ARRAY [1..10,1..10] of REAL;**

**ВВОД**

WRITELN('количество строк');

READLN(n);

WRITELN('количество столбцов');

READLN(m);

FOR i=1 to n do

FOR j=1 to m do

READLN(b[i,j]);

**ВЫВОД**

FOR I:=1 to n DO

BEGIN

FOR J:=1 to m DO

WRITE(b[i,j]:3);

WRITELN

END;

**СУММА**

S:=0;

FOR i:=1 to n DO

FOR J:=1 to m DO

S:=S+b[I,j];

Writeln(‘S= ‘,S:6:2);

**Сумма по строкам**

For i:=1 to n Do

Begin

S:=0;

For j:=1 to m Do

S:=S+b[i,j];

Writeln(‘S= ‘,S:6:2);

End;

**Сумма по столбцам**

For j:=1 to n Do

Begin

S:=0;

For i:=1 to m Do

S:=S+b[i,j];

Writeln(‘S= ‘,S:6:2);

End;

**ПРОИЗВЕДЕНИЕ**

PR:=1;

FOR i:=1 to n DO

FOR j:=1 to m DO

PR:=PR\*b[i,j];

WRITELN('PR= ',PR:10:6);

**Максимум**

max:=-MaxInt;

FOR i:=1 to n DO

FOR j:=1 to m DO

IF b[i,j]>max then

Begin

b[i,j]:=max;

end;

imax:=i;

imax:=j;

end;

WRITELN(‘максимальное значение =’,max:4);

WRITELN('номер максимального значения i:= ',imax:4);

WRITELN('номер максимального значения j:= ',jmax:4);

**МИНИМУМ**

min:=MaxInt;

FOR i:=1 to n DO

FOR j:=1 to m DO

IF b[i,j]<min then

Begin

b[i,j]:=min;

imin:=i;

jmin:=j;

writeln('минимальное значение = ',max:4);

writeln('номер минимального значения i:= ', imin:4);

writeln(‘номер минимального значения j:= ’,jmin:4);

end;

**ОБМЕН**

**min с 1**

b[imin,jmin]:=b[1,1];

b[1,1]:=imin;

**max с 5**

b[imax,jmax]:=B[n,m];

b[5,5]:=imax;

**min с max**

b[imin,jmin]:=max;

b[imax,jmax]:=min;

**Сумма главной диагонали**

S=0;

For i:=1 to n Do

For j:=1 to m Do

If i=j then

Begin

S:=S+b[i,j];

Writeln(‘ cумма = ’,S:4);

End;

**Сумма побочной диагонали**

S=0

For i:=1 to n Do

For j:=1 to m Do

If(i+j)=(n+1) then

Begin

S:=S+b[i,j];

Writeln(‘сумма= ’,S:4);

End;

## 20. Множества

**Элементами множества являются величины базового типа ,кроме REAL**

**Var**

**<имя\_множества> : set of <базовый тип>;**

**Операции над множествами:**

**1.Присваивание**

<пер>:=<множественное выражение>;

Примеры:

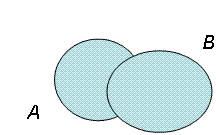
A:=[50,100,150,200];

**2.Объединение**

**Объединением** двух множеств A и B называется множество, состоящее из элементов, входящих хотя бы в одно из множеств A или B. Знак операции объединения в Паскале «+»

Пример:

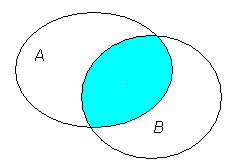
1) [1, 2, 3, 4] + [3, 4, 5, 6] => [1, 2, 3, 4, 5, 6];



**3.Пересечение**

**Пересечением** двух множеств A и B называется множество, состоящее из элементов, одновременно входящих во множество A и во множество B.

Знак операции пересечения в Паскале «\*»

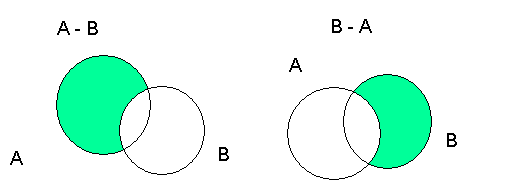


Пример:

1) [1, 2, 3, 4] \* [3, 4, 5, 6] => [3, 4]

**4.Разность**

**Разностью** двух множеств A и B называется множество, состоящее из элементов множества A, не входящих во множество B.



Примеры:

1) [1, 2, 3, 4] - [3, 4, 5, 6] => [1, 2]

**5.Операции отношения:**

A=B Множества A и B совпадают;

A<>B Множества A и B не совпадают;

A<=B Все элементы A принадлежат B;

A>=B Все элементы B принадлежат A;

**6.Операция вхождения:**

Операции вхождения - это операция, устанавливающая связь между множеством и скалярной величиной, тип которой совпадает с базовым типом множества. Если X-такая скалярная величина,

а М-множество[1,3,2,4] то операция вхождения записывается так:

4In M-true

5In M-false

# 21. Динамические переменные, указатели, процедуры

**Динамические переменные** - это переменные, память под которые выделяется во время выполнения программы.

**Память**

Статистическая Динамическая

Заполняется во время компиляции    выделение памяти под объект осуществляется

Во время выполнения программы

**Работа с динамическими переменными складываются из двух шагов:**

1.Найти адрес поля

2.Извлечь из этого корня значение

**Размер поля определяется типом величины:**

X: Byte – 1 байт;

X: Integer – 2 байта;

X: longint – 4 байта;

**Объявления ссылочного типа**

VAR

< > :=< ссылочный тип >;

**Type**

**Mas=array [1..100] of Real;**

**Var**

## 22.Процедуры

PROGRAM xx;

VAR

**PROCEDURE**<имя> [( список формальных параметров)];

**VAR** локальные переменные

**BEGIN**

**END;**

BEGIN

Обращение (вызов)

<имя>

[( список формальных параметров )];

END.

**Пример**

PROGRAM xx;

VAR

**PROCEDURE KORNI** (a , b ,c:Integer ; VAR x1,x2:REAL);

**BEGIN**

**END;**

BEGIN

KORNI (a1, b1, c1, y1, y2);

Обращение

<имя> [(список)]

END.

## 23.ФУНКЦИИ

PROGRAM xx;

VAR

**FUNCTION** <имя> [( список параметров )]:<тип>;

**VAR**

**BEGIN**

**<Имя>:=<Вычисленное значение> ;**

**END;**

BEGIN

END.

Пример

Написать функцию вычисления факториала

**FUNCTION FACT** (n:BYTE):Integer;

**VAR**

I: Byte;

y: Integer;

**BEGIN**

y:=1

FOR i:=1 to n Do

y:=y\*i;

Fact: =y;

**END;**

## 24.Обращение к подпрограммам

Обращение к подпрограммам осуществляется указанием имени подпрограммы и списка фактических параметров. Количество, тип и порядок следования фактических и формальных параметров должны совпадать.

**Параметры значения**

Служат для передачи исходных значений из основной программы в подпрограмму.

Для этого примера a,b,c- параметры значения

procedure KORNI(a, b, c: real; **var** x1, x2: real);

## 25. Параметры переменные

Служат для передачи результатов из подпрограммы в основную программу

Для этого примера x1,x2-параметры переменные**.**

**procedure KORNI(a, b, c: real; var x1, x2: real);**

## 26.Доступ к компонентам записи. Понятие записи.

Структура записи определяется типом **RECORD**

PROGRAM XX;

TYPE

<ИМЯ>=RECORD

<ИМЯ\_ПОЛЯ\_1>:<ТИП>;

<ИМЯ\_ПОЛЯ\_2>:<ТИП>;

… … … …

<ИМЯ\_ПОЛЯ\_n>:<ТИП>;

END;

**ОПЕРАЦИИ:**

1.Создание (появляется новый файл данных на диске)

2.Просмотр

3.Добавление (добавим несколько записей в конец файла)

4.Редактирование

5.Удаление

|  |
| --- |
| 27. Файлы **Файл-**это поименованная область на диске |

# 28.Текстовые файл

VAR

F: Text; - объявление текстового файла

# 29. Процедуры при работе с текстовыми файлами

1.**ASSIGN(F,' ПОЛНОЕ ИМЯ ');**

2.**REWRITE(F);**

3.**RESET(F);**

**Append(F);**

4.**READ(F;< ИМЯ ЗАПИСИ** >);

5 **READLN(F,<имя записи>);**

. **WRITE (F;<имя записи>);**

11.**CLOSE (F)**;

# 30. Типизированные файлы

**Просмотр**

ASSIGN (F,’D:\ИМЯ ПАПКИ\SPR.DAN’);

RESET(F);

WHILE NOT EOF (F) DO

**Создание**

assign(fstud, ‘D:\имя папки\spr.dan’);

rewrite(fstud);

Close(fstud);

**Добавление**

Assign(fstud,’D:\имя папки\spr.dan’);

Reset(fstud);

Seek(fstud, filesize(f));

While true do

With stud do

Begin

Writeln(‘введите номер’);

Readln(N);

Writeln(‘введите Ф.И.О’);

Readln(FIO);

Writeln(‘введите год рождения’);

Readln(GR);

Writeln(‘введите пол’);

Readln(POL);

Write (fstud,stud);

Writeln(‘хотите продолжить ввод?’);

Readln(answer);

If(answer=’N’)or(answer=’n’)then

Begin

Close(fstud);

Exit

End;

End;

End.

**УДАЛЕНИЕ**

PROGRAM UD;

TYPE

ZAP=RECORD

NP: BYTE;

FIO: STRING [30];

GR: WORD;

POL: CHAR;

END;

VAR

STUD: ZAP

f: FILE OF ZAP;

ANSWER: CHAR;

BEGIN

ASSIGN (f,'D:\pvm 21-41\f.DAN');

RESET (f);

ASSIGN (fc,'D:\pvm 21-41\fc.DAN');

REWRITE(fc);

WITELN('введите номер удаляемой записи');

READLN (D);

WHILE NOT EOF (f) DO

WITH STUD DO

BEGIN

READ (f,STUD);

IF D<>N THEN

WRITE (fc,STUD);

END;

CLOSE (f);

CLOSE (fc);

ERACE (f);

RENAME (fc,’D:\pvm 21-41\f.DAN’);

END.

**Корректировка**

Program xx;

Type

Zap=record

N:byte;

FIO:strring[30];

GR:1985..1988;

SEX:char;

End;

Var

Stud:zap;

F:file of zap;

K:byte;

Begin

Assign(f,’d:\pvm 21-41)f.dan’);

Reset(f);

While true do

Begin

Writeln(‘введите номер корректированной записи’);

Readln(n);

If k=0 then

Begin

Close(f);

Exit

End

Else

Begin

Seek (f,k-1);

With stud do

Begin

Writeln(‘введите номер 1’);

Readln(n);

Writeln(‘введите номер Ф.И.О.’);

Readln(FIO);

Writeln(‘введите год рождения’);

Readln(GR);

Writeln(‘введите пол’);

Readln(sex);

Write(f,stud)

End;

End;

End;

End.

# 31.Модуль.

Unit<имя модуля>;

Interface

Procedure<имя процедурs>;

Implementation

Procedure<имя процедурs>;

Begin

End;

[begin]

End.

# 32.Графика

Графический модуль содержит около 80 процедур и функций.

1.Инициализация графической системы:

INITGRAPH

( Var GD: Integer; Var GH :Integer; PATH TO GD : String);

CLOSEGRAPH

2.Процедуры параметров изображения:

GRAPHDEFAULTS

SETCOLOR

SETBKCOLOR

SETLINESTYLE

SETFILLSTYLE

3.Процедуры для получения изображения:

OUTTEXT

PUTPIXEL ( X, Y, COLOR)

LINE ( X1, Y1, X2, Y2)

LINETO ( X, Y)

LINEREL ( DX, DY)-

MOVETO (X1, Y1)-

MOVEREL ( DX, DY)

ARC (X, Y, ST, END, R)

ELLIPCE (X, Y, ST, END, XR, YR)

FILLELLIPSE (X, Y, XR, YR)

CIRCLE (X, Y, R)

PIESLICE (X, Y, ST, END, R)

RESTANGLE (X1, Y1, X2, Y2)

BAR (X1, Y1, X2, Y2)

BAR3D (X1, Y1, X2, Y2, D, P)

DRAWPOLY (N, KOORDS)

FILLPOLY (N, KOORDS)

FLOODFILL (X, Y, BORDER)