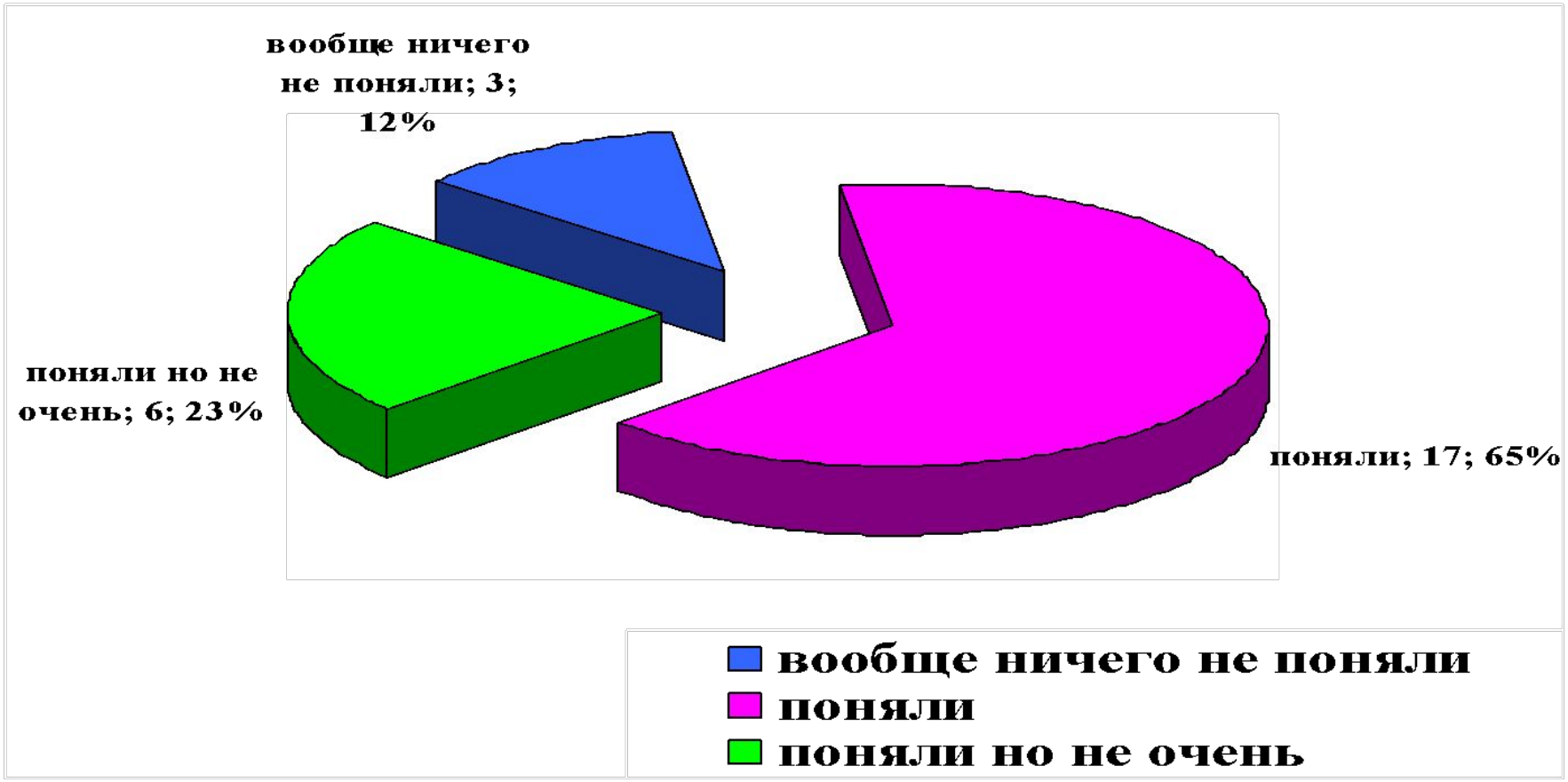
Программирование на языке TURBO PASCAL (Учебник для начинающих)

Проект: Варламовой Кристины

Учитель: Касприк Е.А.

Социологическое исследование 11 «В» класса:

- Вот как ответили ученики моего класса на вопрос : «Поняли ли вы тему «Программирование в среде «TURBO PASCAL»?»



Цель проекта

- Сформировать умения разрабатывать и использовать программы для решения определенных типов задач.

Задачи проекта:

- Познакомиться основными понятиями среды **TURBO PASCAL** ;
- Описать стандартные приемы решения задач и построения программ;
- Создать банк задач, в виде лабораторного практикума, для закрепления основных умений и навыков работы в среде.

Основные этапы проектирования программ

Проектирование и последующая поддержка программного обеспечения включает несколько этапов:

- разработка спецификации (specification);
- проектирование или конструирование программы (design);
- доказательство правильности проекта (verification);
- кодирование, т. е. описание программ на языке программирования (coding);
- отладка и проверка правильности программ (debugging and testing);
- доработка и улучшение программ (refining);
- производство окончательного программного продукта (production);
- поддержка программного продукта в процессе его использования (maintenance).

Общая схема построения программы

- Блок 1. Задание исходных данных.
- Блок 2. Решение поставленной задачи.
- Блок 3. Вывод полученных результатов.

Числовые типы данных.

Выражения. Стандартные функции. Решение задач линейной структуры

- ***Описание целых типов***

ЦЕЛЫЕ ТИПЫ		
Название	Длина, байт	Диапазон значений
Byte	1	0 . . 255
ShortInt	1	-128 . . +127
Word	2	0 . . 65535
Integer	2	-32768 . . +32767
LongInt	4	-2 147 483 648 . . +2 147 483 647

Описание действительных типов

Длина, байт	Название	Количество значащих цифр	Диапазон десятичного порядка
4	Single	7..8	-45..+38
6	Real	11 .. 12	-39 . . +38
8	Double	15 . .16	-324 . . +308
10	Extended	19 . .20	-4951 . . +4932
8	Comp	19 . . 20	$-2*10^{63}+1..+2*10^{63}-1$

Выражения и операторы действия

Выражение	Типы операндов	Тип результата	Операция
-A	R, I	R, I	Унарный минус (изменение знака числа)
+A	R, I	R, I	Унарный плюс (изменение знака числа)
A+B	I, I	I	Сложение
	R, R; R, I; I, R	R	
A-B	I, I	I	Вычитание
	R, R; R, I; I, R	R	
A*B	I, I	I	Умножение
	R, R; R, I; I, R	R	
A/B	R, R; R, I; I, R; I, I	R	Деление
A div B	I, I	I	Целочисленное деление
A mod B	I, I	I	Остаток от целочисленного деления

Стандартные математические функции Турбо Паскаля

Обращение	Тип аргумента	Тип результата	Примечание
Abs(x)	Real, integer	Тип аргумента	Модуль аргумента
ArcTan(x)	Real, integer	Real	Арктангенс (значение в радианах)
Cos(x)	Real, integer	Real	Косинус, угол в радианах
Exp(x)	Real, integer	Real	Экспонента
Frac(x)	Real	Real	Дробная часть числа
Int(x)	Real, integer	Real	Целая часть числа
Ln(x)	Real, integer	Real	Логарифм натуральный
Pi	Нет	Real	3,141592653…
Sin(x)	Real, integer	Real	Синус, угол в радианах
Sqr(x)	Real, integer	Тип аргумента	Квадрат аргумента
Sqrt(x)	Real, integer	Real	Корень квадратный
random	Нет	Real	Псевдослучайное число в интервале [0, 1]
Random(I)	Integer	Integer	Псевдослучайное число в интервале [0, I]
Round(x)	Real	Integer	Округление до ближайшего целого
Trunc(x)	Real	Integer	Отбрасывание дробной части числа

Порядок вычислений в выражениях следующий:

- вычисляются подвыражения, заключенные в скобки;
- затем выполняются операции с наибольшим приоритетом; обычно используются следующие уровни приоритетов (в порядке убывания):
- возведение в степень;
- мультипликативные операции: * , / , div , mod;
- унарные операции: + , - , abs , not;
- аддитивные операции: +, -;
- операции отношения: = , <> , < , > , <= , >= ;
- логические операции: and, or, not;
- операции с одинаковым приоритетом выполняются слева направо

Пример заданий на отработку записи основных выражений

- **Математическое выражение**
- x3/2 – 7x + tg (x+2)
- **Выражение на Паскале**
  exp(3*ln(x)/2)-7*x+sin(x+2)/cos(x+2)

Пример: вычислить значение выражения

$$\frac{b+\sqrt{b^2+4ac}}{2a}-a^3c$$

Текст программы:

Program zadacha1 _2;
Var a,b,c,y: real; {описываем переменные, используемые в программе}
Begin {начало основной программы}
Writeln(‘введите значение переменной a, не равное 0’);
Readln (a); {ввод с клавиатуры переменной a}
Writeln (‘введите значения переменных b и c через пробел’);
Readln (b,c); {ввод с клавиатуры переменных b,c}
Y:= (b+sqrt(sqr(b)+4*a*c))/(2*a)-sqr(a)*a*c; {вычисление значения выражения}
Writeln(‘значение искомого выражения равно’, y:8:4); {вывод результата на экран}
End.

Пример: Найти сумму цифр трехзначного натурального числа.

Текст программы:

```
Program zadacha1_3;
Var N: word;
ed, des, sot: 0..9;
s: byte;
begin {начало программы}
writeln(‘введите трехзначное число’);
readln(N);
s:=0;
ed:=N mod 10;
des:= (N div 10) mod 10;
sot:= N div 100;
s:= ed+des+sot;
writeln(‘сумма цифр числа  ‘, N, ‘  равна  ‘, s);
end.
```

Логические выражения.

Логический тип. Условный оператор. Оператор выбора. Программирование разветвляющихся алгоритмов

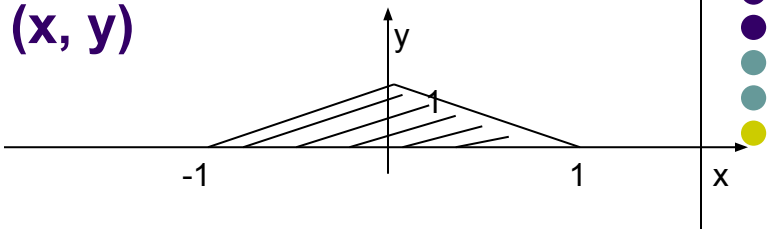
- **Логические выражения (условия)** – это выражения, которые могут принимать лишь одно из двух значений: **true** (истина) или **false** (ложь). Для построения логических выражений используются **операции отношения**, которые обозначаются знаками: = (отношение на равенство), <> (отношение на неравенство), < (отношение меньше), > (отношение больше), <= (отношение меньше или равно), >= (отношение больше или равно).
- Сложные условия составляются из простых с помощью **логических операций**: **and** (логическое «И»), **or** (логическое «ИЛИ») и **not** (логическое «НЕ»). При составлении сложных условий операнды логического выражения берутся в скобки ***(это важно!)***.
- Примеры логических выражений:
- 5>3; 2<=6; (x<2) and (x>=0); 2*x+5<>0 и т.п.
- При вычислении логических выражений операции выполняются в следующем порядке: not, and, or, операции отношения, арифметические операции. Если порядок выполнения операций нужно изменить, то применяют скобки.

Условные операторы позволяют выбирать для выполнения те или иные части программы в зависимости от некоторых условий

Полный условный оператор:

- **if B then S1 else S2**
- Здесь **if** (если), **then** (то) и **else** (иначе) являются служебными словами, **B** – логическое выражение, а **S1** и **S2** – операторы.
- Выполнение такого условного оператора сводится к выполнению одного из входящих в него операторов S1 или S2: *если заданное в операторе условие выполняется (логическое выражение B принимает значение true), то выполняется оператор S1, в противном случае выполняется оператор S2*.

Пример: составить программу, которая определяет, принадлежит точка с координатами (x, y) заштрихованной области или нет.



- **Текст программы**
- **Program** zadacha2_2;
- Var x,y: real;
- **begin**
- **Witeln** (‘введите координаты точки x и y через пробел’);
- **Readln** (x, y);
- **if** (y>=0) **and** (x+y<=1) **and** (y-x<=1)
- **then**
- **Writeln** (‘Точка принадлежит области’)
- **else**
- **writeln**(‘Точка не принадлежит области’);
- **end.**

Циклические структуры. Программирование циклических алгоритмов.

- **Простой арифметический цикл (с параметром)**

For V:= E1 to E2 do S,
где **for** (для), **to** (увеличиваясь к) и **do** (выполнять, делать) – служебные слова,
V – переменная порядкового типа, называемая параметром цикла,
E1 и **E2** – выражения того же типа, что и параметр цикла,
S – оператор, который и выполняется многократно в цикле, называемый телом цикла.

Оператор цикла с постусловием

Для задания таких вычислительных процессов в Паскале и служит оператор цикла с постусловием. Этот оператор имеет вид:
Repeat S1; S2;…;Si until B,
где **repeat** (повторять) и **until** (до) – служебные слова, через **Si** обозначен любой оператор Паскаля, а через **B** – логическое выражение.
При выполнении этого оператора цикла *последовательность операторов, находящихся между словами repeat и until, выполнится один или более раз. Этот процесс завершается, когда после очередного выполнения заданной последовательности операторов логическое выражение B примет (впервые) значение true*

Циклические структуры. Программирование циклических алгоритмов.

- **Оператор цикла с предусловием**

Этот оператор цикла имеет в Паскале следующий вид:

While B do S,

где **while** (пока), **do** (делать, выполнять) – служебные слова, **B** – логическое выражение, **S** – оператор. Здесь *оператор S выполняется ноль или более раз, но перед каждым очередным его выполнением вычисляется значение выражения B, и оператор S выполняется только в том случае, когда значение выражения B true. Выполнение оператора цикла завершается, когда выражение B впервые принимает значение false. Если это значение выражение B принимает при первом же его вычислении, то оператор S не выполнится ни разу*.

Пример: **вычислить сумму членов гармонического ряда с общим членом an=1/n, превышающих заданное число ε (0<ε<1).**

- **Текст программы**
- **Program** zadacha3_2;
- **Var** E,s: real;
- N: word;
- **Begin**
- **Writeln** (‘введите E’);
- **Readln** (E);
- S:=0; N:=1;
- **While** 1/n>E do
- **Begin** {так как в цикле должны выполняться два оператора, то их нужно заключить в операторные скобки}
- S:=S+1/n;
- N:=n+1;
- **End**; {конец тела цикла}
- **Writeln**(‘искомая сумма S=’, S)
- **End.**

Общие выводы:

- Данное пособие представляет собой учебник для людей, начинающих программировать в нем отобраны наиболее популярные задачи для отработки навыков решения задач и работе в среде программирования **TURBO PASCAL**

- Спасибо за внимание!