Лекция №2

Элементы языка Паскаль

План лекции

1. Язык Паскаль
2. Алфавит языка Паскаль.
3. Структура Паскаль-программы.
4. Комментарии.
5. Идентификаторы.
6. Переменные и типы данных.
7. Оператор присваивания.
8. Константы.
9. Операции и выражения.
   - 9.1. Арифметические операции.
   - 9.2. Логические операции.
   - 9.3. Стандартные математические функции.
   - 9.4. Порядок вычислений. Запись выражений.
10. Операторы ввода-вывода.
11. Простейшие операторы.
12. Пример простейшей программы на языке Паскаль.

Язык Паскаль

Язык 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.

Язык Паскаль

- Причины популярности языка Паскаль
  - Прост для обучения.
  - Отражает фундаментальные идеи алгоритмов в легко воспринимаемой форме, что предоставляет программисту средства, помогающие проектировать программы.
  - Позволяет четко реализовать идеи структурного программирования и структурной организации данных.
  - Использование простых и гибких структур управления: ветвлений, циклов.
  - Надежность разрабатываемых программ.

Алфавит языка Паскаль

Алфавит языка – набор элементарных символов, используемый для составления программ.

Алфавит содержит:

- 52 буквы латинского алфавита (строчные и заглавные);
- арабские цифры (0-9);
- специальные символы;
- знаки математических действий (+ – * /);
- знаки пунктуации (. : , ; " `);
- скобки ([ ] ( ) { });
- знак пробела;
- знаки отношений (< > =).

Структура Паскаль - программ

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.
- раздел описания модулей (uses);
- раздел описания меток (label);

Программный блок

- раздел описания констант (const);
- раздел описания типов данных (type);
- раздел описания переменных (var);

Общая структура программы на языке Паскаль

Program ИМЯ..; {заголовок программы}

Uses ...; {раздел описания модулей}

Var ..; {раздел объявления переменных}

...

Begin {начало исполнительной части программы}

... {последовательность
... операторов}

End. {конец программы}

Пример программы на языке Паскаль

```
Program  z1 (input, output);
{ Программа для вычисления площади прямоугольника
    по заданным сторонам}
Var
     a,b,s : integer;
Begin
     writeln (‘Введите стороны A и B’);
     read (a, b);
     s:=a*b;
     write (‘S=‘,s,’ кв.см.’)
End.
```

Комментарии

Комментарий – это строка (или несколько строк) из произвольных символов, заключенная в фигурные скобки:

{ комментарий }

Другой вариант оформления комментария:

(комментарий *)*

Внутри самого комментария символы } или *) встречаться не должны.

Идентификаторы

Имена, даваемые программным объектам (константам, типам, переменным, функциям и процедурам, да и всей программе целиком) называются **идентификаторами**.

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.

Примеры зарезервированных слов

and	goto	set	
array	implementation	shl	
begin	in	shr	
case	interface	string	
const	label	then	
div	mod	text	
do	nil	to	
downto	not	type	
else	of	unit	
end	or	until	
file	pointer	uses	
far	procedure	var	
for	program	while	
forward	record	with	
function	repeat	xor	

Переменные и типы данных

Переменная – это программный объект, значение которого может изменяться в процессе работы программы.

Тип данных – это характеристика диапазона значений, которые могут принимать переменные, относящиеся к этому типу данных.

Все используемые в программе переменные должны быть описаны в специальном разделе **var** по следующему шаблону:

var
<имя_переменной_1> [, <имя_переменной_2, _>] : <имя_типа_1>;
<имя_переменной_3> [, <имя_переменной_4, _>] : <имя_типа_2>;

О типах данных

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

Простые:
- integer – целых чисел;
- real – действительных чисел;
- char – символьный тип;
- string – строковый;
- boolean – логический;

Сложные:
- record – комбинированный;
- set – множественный

и другие.

Описание переменных

Приведем пример описания переменных:

```
Var
    a : integer;
    b,c : real;
    m : boolean;
```

Оператор присваивания

Оператор присваивания – основной оператор любого языка программирования.

Общая форма записи оператора:

имя величины **:=** выражение

Например, V:=5;
или X:=A+1;

Константы

Константа – это объект, значение которого известно еще до начала работы программы.

В языке Pascal существует три вида констант:

- неименованные константы (цифры и числа, символы и строки, множества);
- именованные нетипизированные константы;
- именованные типизированные константы.

Неименованные константы

Неименованные константы не имеют имен, и потому их не нужно описывать.

Примерами использования неименованных констант могут послужить следующие операторы:

```
I := -10;
r := 12.075 + x;
c := 'z';
s := 'abc' + st;
s5 := [1,3,5] * s6;
b := true;
```

Нетипизированные константы

Именованные константы, как следует из их названия, должны иметь имя.

Эти имена необходимо сообщить компилятору, то есть описать в специальном разделе **const**.

Если не указывать тип константы, то по ее внешнему виду компилятор сам определит, к какому (базовому) типу ее отнести.

Нетипизированные константы

Вот несколько примеров описания нетипизированных именованных констант:

```
const
  n = -10;
  m = 1000000000;
  mmm = n*100;
  x = 2.5;
  c = 'z';
  s = 'string';
  b = true;
```

Типизированные константы

Типизированные именованные константы представляют собой ***переменные(!)*** с начальным значением, которое к моменту старта программы уже известно.
Типизированные константы нельзя использовать для определения других констант, типов данных и переменных.
Их значения можно изменять в процессе работы программы.

Описание типизированных констант производится по следующему шаблону:

```
const
<имя_константы> : <тип_константы> = <начальное_значение>;
```

Типизированные константы

Пример описания типизированных констант

```
const
    n: integer = -10;
    x: real = 2.5;
    c: char = 'z';
    b: boolean = true;
```

Операции и выражения. Арифметические операции.

Операции общей арифметики (арифметические оперции)

+ сложение
– вычитание
* умножение
/ деление

Пример арифметического выражения :

$$y = \frac{2k + 5}{7 - x}$$ y:=(2*k+5)/(7–x);

Арифметические операции

Операции целочисленной арифметики применимы, как легко догадаться, только к целым типам.

a **div** b – деление a на b нацело

Пример:

```
x:=13;
y:=5;
z:=x div y;
```

В результате переменная **z** получит значение 2.

Арифметические операции

a **mod** b – взятие остатка при делении a на b нацело.

Пример:

```
x:=13;
y:=5;
z:=x mod y;
```

В результате переменная **z** получит значение 3.

Логические операции

Операции сравнения

= равно

<> неравно

> больше

< меньше

<= меньше либо равно

>= больше либо равно

Применимы ко всем базовым типам.

Результатом является значение **истина** (true) или **ложь** (false)

Логические операции

Пример операций сравнения:

5>=4	true
7<=7	true
(2+7)<3	false
true=false	false
a>b	зависит от значений a и b

Логические операции

and	логическое «и» (конъюнкция)
or	логическое «или» (дизъюнкция)
not	логическое «не» (инверсия)
xor	логическое «или исключающее»

Логические операции

Обозначим **true** за **1** , а **false** за **0**.
Рассмотрим таблицу истинности для конъюнкции

X	Y	X and Y
0	0	0
0	1	0
1	0	0
1	1	1

Пример использования операции **and** :

(x>2) and (x<5)

Логические операции

Обозначим **true** за **1** , а **false** за **0**.
Рассмотрим таблицу истинности для дизъюнкции

X	Y	X or Y
0	0	0
0	1	1
1	0	1
1	1	1

Пример использования операции **or** :

(x<2) or (x>5)

Логические операции

Обозначим **true** за **1** , а **false** за **0**.
Рассмотрим таблицу истинности для «или исключающего»

X	Y	X xor Y
0	0	0
0	1	1
1	0	1
1	1	0

Пример использования операции **xor** :

(x<2) xor (x>5)

Логические операции

Обозначим **true** за **1** , а **false** за **0**.
Рассмотрим таблицу истинности для инверсии

X	not X
0	1
1	0

Пример использования операции **not** :

not ((x<=2) xor (x>=5))

Стандартные математические функции

Функция	*Описание*
abs(x)	Абсолютное значение (модуль) числа
arctan(x)	Арктангенс (в радианах)
cos(x)	Косинус (в радианах)
exp(x)	Экспонента (e^x)
frac(x)	Взятие дробной части числа
int(x)	Взятие целой части числа
ln(x)	Натуральный логарифм (по основанию e)
odd(x)	Проверка нечетности числа
pi	Значение числа π
round(x)	Округление к ближайшему целому
trunc(x)	Округление "вниз" - к ближайшему меньшему целому
sin(x)	Синус (в радианах)
sqr(x)	Возведение в квадрат
sqrt(x)	Извлечение квадратного корня

Порядок вычислений

Приоритеты операций языка Pascal

	Операции	**Приоритет**
Унарные операции	not	Первый(высший)
Операции, эквивалентные умножению	*, /, div, mod, and	Второй
Операции, эквивалентные сложению	+,-, or, xor	Третий
Операции сравнения	=, <>, >, <, <=, >=	Четвертый

Запись выражений

Пример записи выражения на языке Pascal

$$b = \sqrt{\frac{2 + x_1 - |x_2|}{\sin^2 \alpha}} + 2k$$

```
b:=sqrt((2+x1-abs(x2))/sqr(sin(alpha)))+2*k;
```

Операторы ввода-вывода

Операторы ввода
Для того чтобы получить данные, вводимые пользователем вручную (то есть с клавиатуры), применяются команды

read(<список_ввода>) и **readln**(<список_ввода>).

Список ввода – это последовательность имен переменных, разделенных запятыми.
Например, при помощи команды

```
read(k,x,c,s);
```

программа может получить с клавиатуры данные сразу для четырех переменных.

Операторы ввода-вывода

Операторы вывода

Для того чтобы вывести на экран какое-либо сообщение, используют процедуру

write(<список_вывода>) или **writeln**(<список_вывода>).

Примеры операторов вывода:

```
write (a);
writeln (s,m,k);
write ( ‘Длина окружности ’, L, ’ см.’ );
```

Простейшие операторы

a**:=** b; – **присваивание.** В данном примере переменной a присваивается значение переменной b.

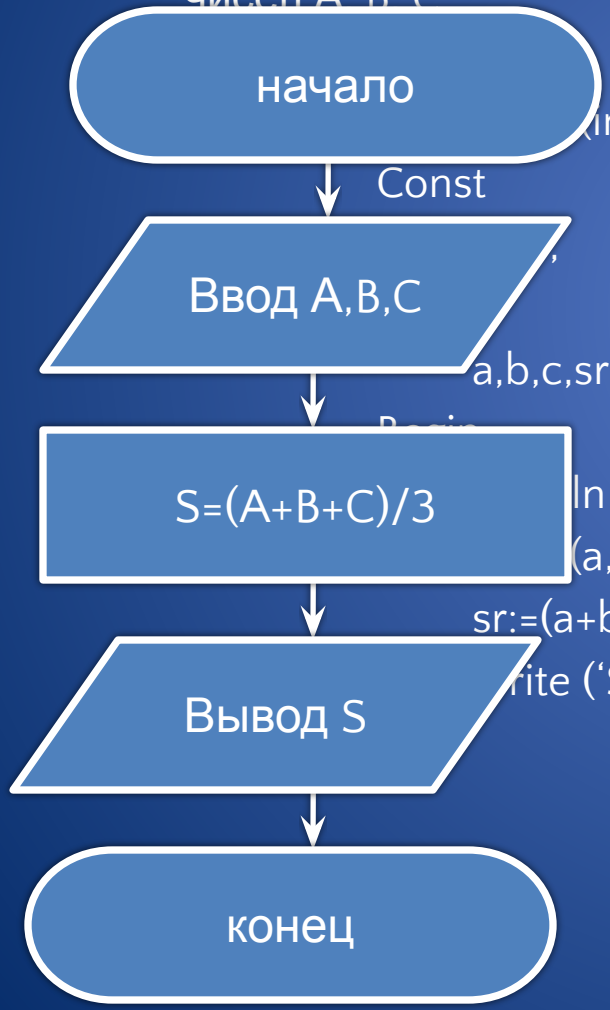
; – **пустой оператор**, который можно вставлять куда угодно, а также вычеркивать откуда угодно, поскольку на целостность программы это никак не влияет.

Операторные скобки превращают несколько операторов в один:

```
begin
    <несколько операторов>
end;
```

Пример реализации линейного алгоритма на языке Pascal

Задача . Составить алгоритм нахождения среднего арифметического трех чисел A, B, C




```
input, output);
Const
,
a,b,c,sr :real;
Begin
ln ('Введите A, B и C');
(a, b, c);
sr:=(a+b+c)/N;
rite ('Sr=',sr)
```

Вопросы

- Кто является автором языка Паскаль? Для каких целей был создан Паскаль?
- Какие буквы, цифры и специальные символы составляют алфавит языка Паскаль?
- Что такое зарезервированное слово? Приведите примеры.
- Игнорируются ли Паскалем различие в высоте символов из которых состоит программа?
- Для чего используются разделители?
- Можно ли располагать несколько операторов в одной строке?
- Как обозначаются ограничители комментария?
- Можно ли комментарии с однотипными ограничителями вкладывать друг в друга?
- Опишите структуру Паскаль-программы.
- Что обозначают в разделе описаний const, var? Приведите примеры.
- Каково назначение идентификаторов? Где они используются?
- Каковы правила записи идентификаторов?
- Приведите примеры правильных и неправильных идентификаторов.
- Сколько первых символов являются значимыми при записи идентификатора?

Вопросы

Что такое выражение? Приведите примеры.
- Где могут использоваться выражения?
- Можно ли считать константу, переменную или обращение к функции выражением?
- Как определяется тип выражения? Приведите пример.
- Вычислите ((1+1/2)*2)/3
63 mod 16
18 div 3
- Какие операции Вы знаете?
- Что такое отношения? Для чего они используются?
- Что такое приоритет выполнения операций?
- Что вы понимаете под логическими операциями? Приведите таблицы истинности для логических операций.
- Имеют ли приоритет логические операции над операциями сравнивания (отношения)?