

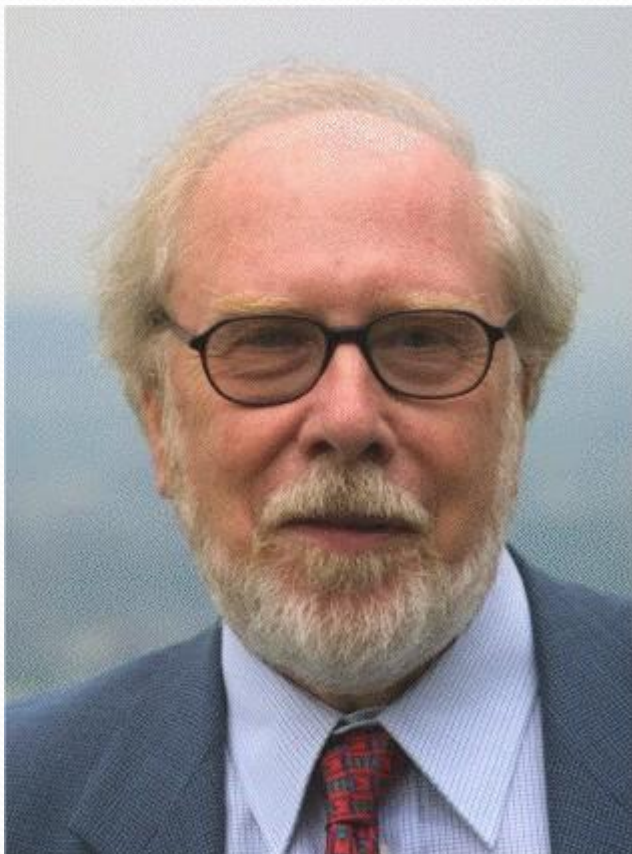
ОСНОВЫ ПРОГРАММИРОВАНИЯ НА ЯЗЫКЕ PASCAL

Немного истории...



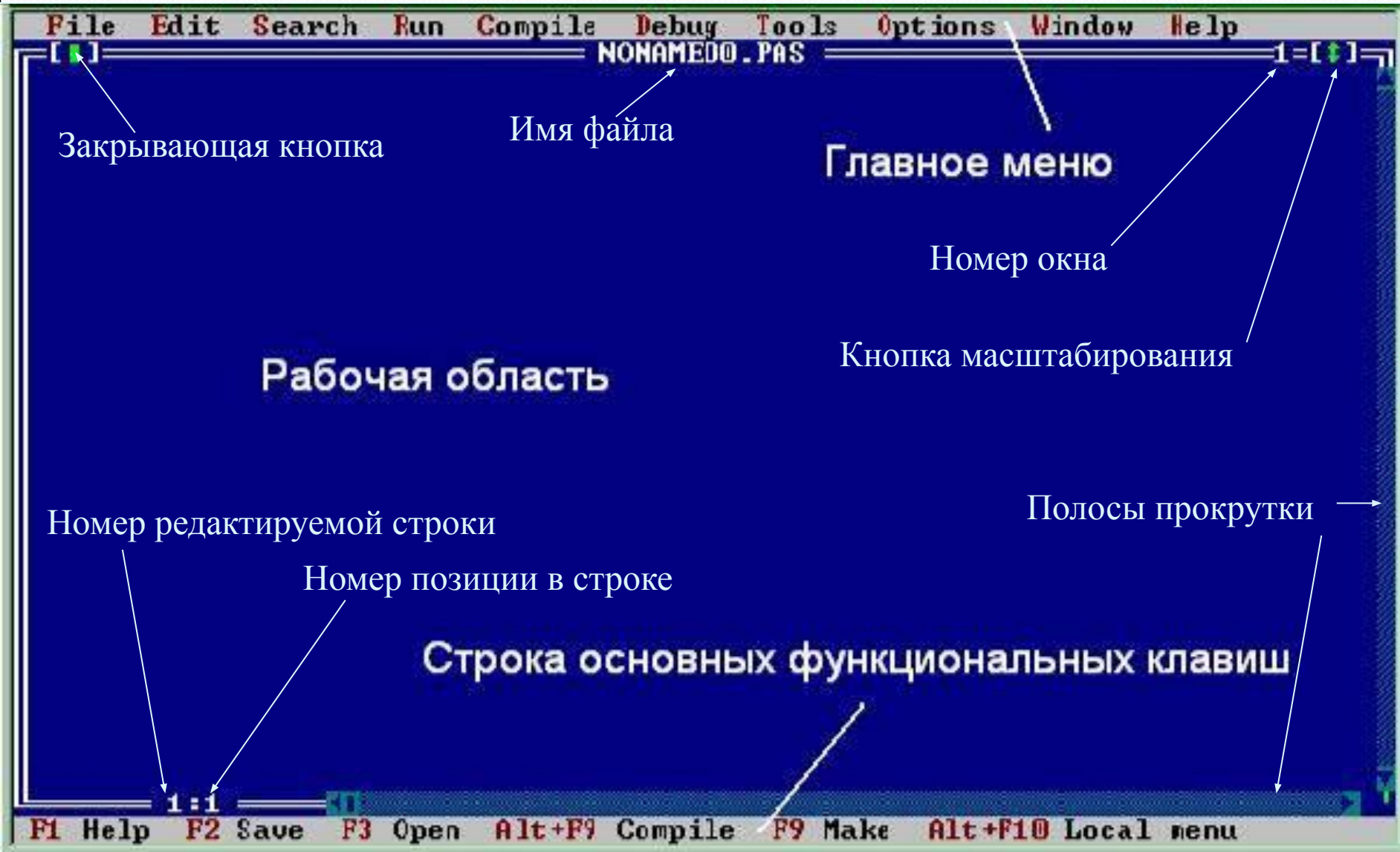
Язык назван в честь выдающегося французского математика, физика, литератора и философа Блеза Паскаля, который создал первую в мире механическую машину, складывающую два числа.

Система PascalABC



Язык Паскаль был разработан Никлаусом Виртом в 1970 г. как язык со строгой типизацией и интуитивно понятным синтаксисом. В 80-е годы наиболее известной реализацией стал компилятор Turbo Pascal фирмы Borland, в 90-е ему на смену пришла среда программирования Delphi, которая стала одной из лучших сред для быстрого создания приложений под Windows. Delphi ввела в язык Паскаль ряд удачных объектно-ориентированных расширений, обновленный язык получил название Object Pascal. Из альтернативных реализаций Object Pascal следует отметить многоплатформенный open source компилятор Free Pascal.

Элементы экрана



Структура программы

Программа на языке PascalABC.NET имеет следующий вид:

program *имя программы;*

раздел uses

раздел описаний **Var** *Описание переменных;*

Определение процедур;

Определение функций;

begin

операторы

end.

Константы

Константы – это данные, значения которых в процессе выполнения программы не могут изменяться.

Константы вводятся в блоке `const`:

```
const
```

```
  a=5;
```

```
  b=1E-3/a;
```

```
  c='значение неизвестно';
```

В общем виде:

ИМЯ КОНСТАНТЫ = выражение;

Переменные

Переменные – это данные, которые могут изменяться в процессе выполнения программы.

Переменные имеют имя, тип и значение.

Описание переменных происходит в блоке var:

```
var
```

```
  a: integer;
```

```
  b: real;
```

```
  c: char;
```

В общем виде:

имя переменной: тип переменной;

Пример описания переменных

- Например:

- **var**

 - a,b,c: integer;

 - d: real := 3.7;

 - s := 'Pascal forever';

 - al := new ArrayList;

 - p1 := 1;

Типы переменных

Некоторые простые типы:

- 1. Целые типы** (ShortInt, Integer, LongInt, Byte, Word).
- 2. Вещественные типы** (Real, Single, Double, Extended, Comp).
- 3. Логический** (Boolean).
- 4. Символьный** (Char).
- 5. Строковые типы** (String, String [n]).

Оператор присваивания:

Общий вид:

переменная := выражение;

Может быть: переменной,
элементом массива,
полем записи и др.

Не может быть:
константа, выражение.

Может быть: константой,
переменной, элементом
массива, арифметическим
или логическим
выражением.

Работа оператора: если справа стоит выражение, то сначала вычисляется его значение, а затем это значение пересылается в переменную стоящую слева.

Операторы ввода и вывода

Оператор ввода:

```
read (список переменных);
```

Оператор вывода:

```
write (‘сообщение’, список переменных);
```

Отличие операторов **read** и **write** от операторов **readln** и **writeln** состоит в том, что после выполнения операторов **readln** и **writeln** курсор переводится на новую строку.

Ввод и вывод данных

- **Ввод данных**

- **read**(<список ввода>);
- **readln**(<список ввода>);

Примеры:

- **read(a,b,c);**{где a,b,c - переменные. Ввод данных осуществляется через пробел}
- **readln(a,b,c);**{где a,b,c - переменные. Ввод данных осуществляется через enter}
- Список вывода может содержать константы, переменные, выражения, формат вывода. Выражения в списке вывода разделяются запятыми.

Ввод и вывод данных

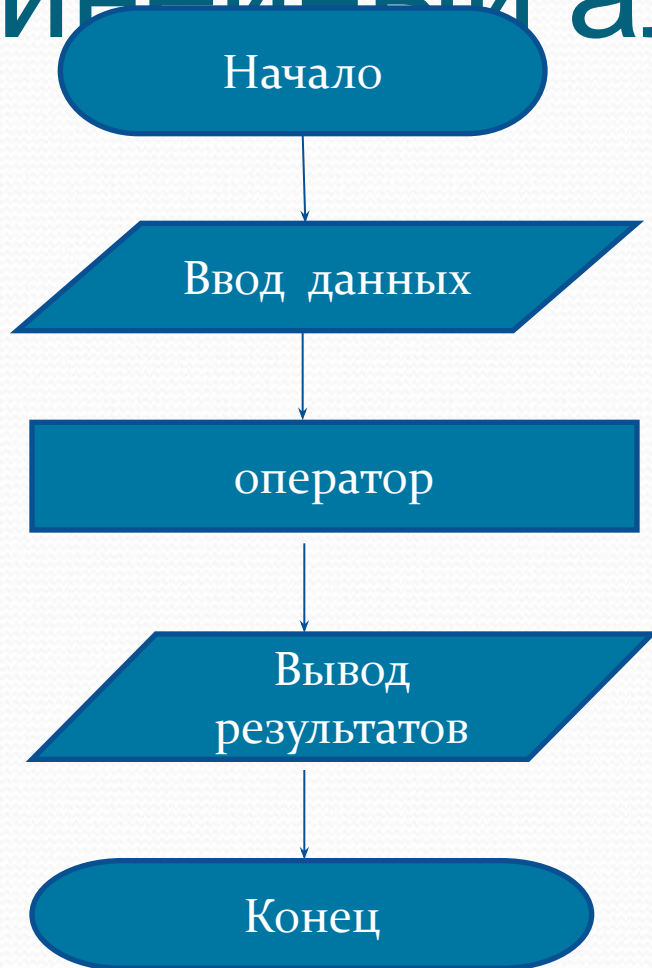
- Вывод данных

- write(<список вывода>);
- writeln(<список вывода>);

Примеры:

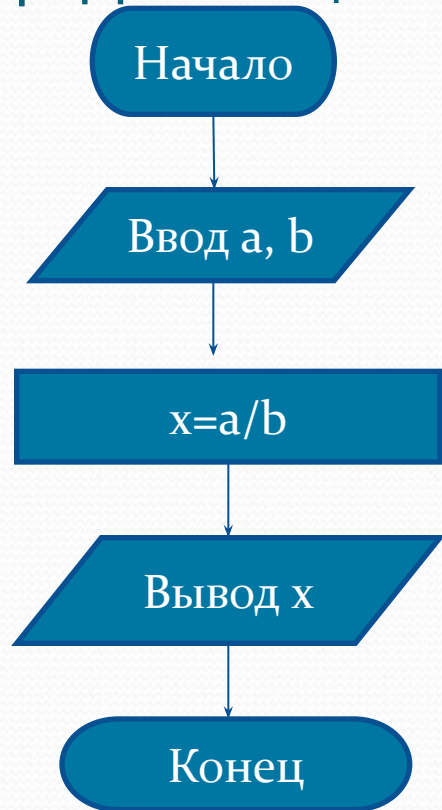
- **write(a,b,c);**{где a,b,c - переменные. После вывода данных на экран, курсор останется на последнем символе}
- **writeln(a,b,c);**{где a,b,c - переменные. После вывода данных на экран, курсор перейдет на новую строку)}
- Окончание **ln** в имени процедуры означает, что курсор автоматически будет переведен в начало следующей строки экрана.

Линейный алгоритм



```
Program имя_программы;  
var  
  {описание данных}  
begin  
  readln(ввод данных);  
  оператор  
  writeln(вывод  
результатов);  
end;
```


Пример: Даны 2 целых числа, найти частное этих чисел



```
program E1;  
var  
  a,b: integer;  
  r: real;  
begin  
  readln(a,b);  
  x := a/b;  
  writeln(x);  
end;
```

Оператор условия

Для реализации ветвления можно использовать условный оператор:

```
if условие then  
    begin  
        оператор; ...  
    end  
else  
    begin  
        оператор; ...  
    end;
```

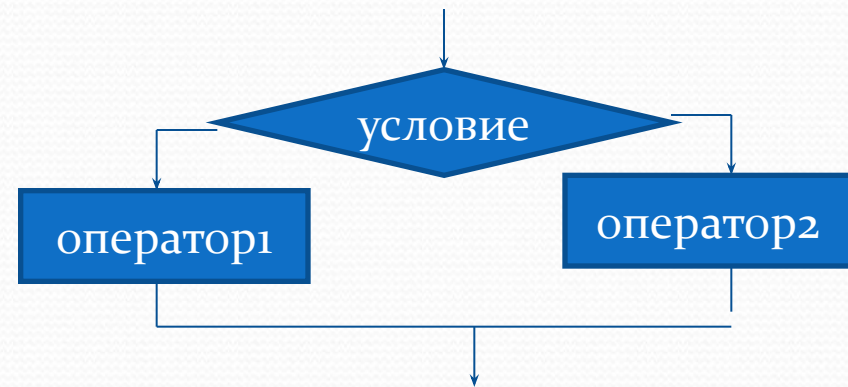
Если условие выполняется, то выполняется ветвь **then**, если условие не выполняется – то ветвь **else**.

Где *условие* – это выражение логического типа.

Условный оператор

Полный условный оператор

- IF *условие* THEN *оператор1*
 ELSE *оператор2*;
- IF *условие* THEN
 BEGIN
 оператор1_1;
 оператор1_2;
 END
 ELSE
 BEGIN
 оператор2_1;
 оператор2_2;
 END;

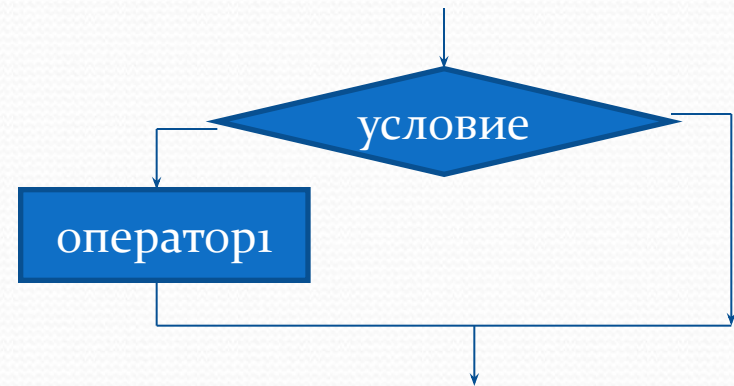


Перед ELSE точка с запятой никогда не ставится!!!

Условный оператор

Неполный условный оператор

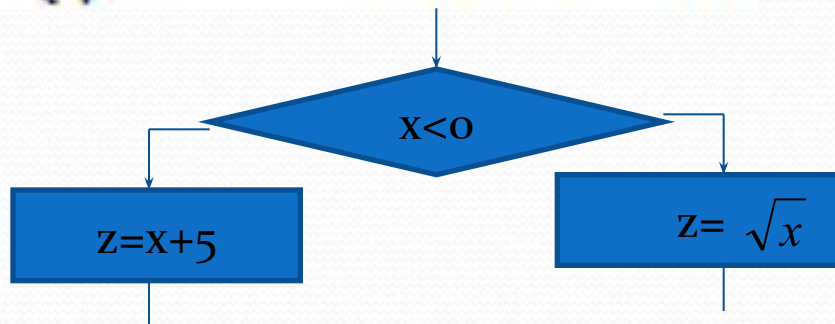
- IF *условие* THEN *оператор1* ;
- IF *условие* THEN
 BEGIN
 оператор1_1;
 оператор1_2;
 END;



условие - это логическое выражение, в зависимости от которого выбирается одна из двух альтернативных ветвей алгоритма. Если значение условия истинно (TRUE), то будет выполняться *оператор 1*, записанный после ключевого слова **then**. В противном случае будет выполнен *оператор 2*, следующий за словом **else**, при этом *оператор 1* пропускается. После выполнения указанных операторов программа переходит к выполнению команды, стоящей непосредственно после оператора **if**.

Пример: Вычислите значение функции

$$z = \begin{cases} x + 5, & \text{при } x < 0 \\ \sqrt{x}, & \text{при } x \geq 0 \end{cases}$$



IF $x < 0$ **THEN** $z := x + 5$ **ELSE** $z := \text{sqrt}(x)$;

ЦИКЛЫ

Если заранее известно количество необходимых повторений, то цикл называется *арифметическим*. Если же количество повторений заранее неизвестно, то говорят об *итерационном* цикле.

В итерационных циклах производится проверка некоторого условия, и в зависимости от результата этой проверки происходит либо выход из цикла, либо повторение выполнения тела цикла. Если проверка условия производится перед выполнением блока операторов, то такой итерационный цикл называется циклом с *предусловием* (цикл "пока"), а если проверка производится после выполнения тела цикла, то это цикл с *постусловием* (цикл "до").

Особенность этих циклов заключается в том, что тело цикла с постусловием всегда выполняется хотя бы один раз, а тело цикла с предусловием может ни разу не выполниться.

Оператор цикла со счётчиком

```
for параметр := нач_значение to кон_значение do  
    begin  
        оператор; ...  
    end;
```

Работа оператора:

1. Вычисляются начальное и конечное значения параметра и фиксируются;
2. Если *нач_значение* \leq *кон_значения*, то выполняется оператор;
3. Значение параметра цикла возрастает (для целого – на единицу);
4. Если значение параметра \neq *кон_значения*, то выполняется оператор, и переходит к п.3
5. Цикл выполняется последний раз, когда параметр = *кон_значению*, затем управление передается оператору после цикла.

Оператор цикла с предусловием

```
while условие do  
    begin  
        оператор; ...  
    end;
```

Работа оператора:

Сначала проверяется условие, если оно верно, то выполняется оператор, затем опять проверяется условие и т.д., пока условие не перестанет выполняться.

Если условие не верно, то оператор игнорируется и управление передается следующему за циклом оператору.

Оператор цикла с постусловием

repeat

оператор; ...

until *условие;*

Работа оператора:

Выполнение операторов
повторяется, пока условие
не станет верным.

Виды циклических алгоритмов

Цикл с
предусловием



Цикл *Пока*

Цикл с
постусловием



Цикл *ДО*

Цикл с
параметром



Цикл *ДЛЯ*

Программа нахождения суммы чисел

```
var i, s: integer;  
begin  
  s:=0;  
  for i:= 1 to 10 do  
    s:=s+i;  
    writeln('s=',s);  
end.
```

ЦИКЛ С ПОСТУСЛОВИЕМ

REPEAT

оператор;

оператор;

...

оператор

UNTIL *выражение;*

Операторы между словами REPEAT и UNTIL повторяются, пока логическое выражение является ложным. Как только логическое выражение становится истинным, происходит выход из цикла.

Так как выражение оценивается после выполнения операторов, то в любом случае операторы выполняются хотя бы один раз.

оператор присваивания

<имя_переменной>:=<выражение>

Примеры:

$$y = \sqrt{x}$$

$$y := \text{sqr}(x);$$

$$y = |x - 5|$$

$$y := \text{abs}(x - 5);$$

$$y = \sin^2 x$$

$$y := \text{sqr}(\sin(x));$$

$$y = \frac{x + 5}{2}$$

$$y := (x + 5)/2;$$

Тест

Вопрос №1

Вопрос №2

Вопрос №3

Вопрос №4

Вопрос №5

Вопрос №6

Вопрос №7

Завершить тест

Вопрос № 1

Определить значение переменной c после выполнения фрагмента программы.

```
a := -5;  
a := a + 6;  
b := -a;  
c := a - 2 * b;
```

1) $c = -11$

2) $c = 15$

3) $c = 27$

4) $c = 33$

Вопрос № 2

Определить значение целочисленных переменных a и b после выполнения фрагмента программы.

```
a:=3+8*4;  
b:=(a div 10)+14;  
a:=(b mod 10)+2;
```

1) $a = 0, b = 18$

3) $a = 10, b = 18$

2) $a = 11, b = 19$

4) $a = 9, b = 17$

Вопрос № 3

Определить значение переменной c после выполнения следующего фрагмента программы.

```
a:=100;  
b:=30;  
a:=a-b*3;  
if a>b then  
  c:=a-b  
else c:=b-a;
```

1) $c = 20$

3) $c = -20$

2) $c = 70$

4) $c = 180$

Вопрос № 4

Определить значение целочисленных переменных x , y и z после выполнения фрагмента программы.

```
x:=52;  
y:=x mod 10;  
z:=x div 10;  
x:=y*10+z;
```

1) $x = 55, y = 2, z = 5$

3) $x = 25, y = 5, z = 2$

2) $x = 22, y = 2, z = 5$

4) $x = 25, y = 2, z = 5$

Вопрос № 5

Определить значение переменной b после выполнения фрагмента программы.

```
a:=2;  
b:=12;  
for i:=1 to 5 do  
    a:=a+3;  
b:=b+a;
```

1) $b = 65$

2) $b = 67$

3) $b = 29$

4) $b = 17$

Вопрос № 6

Определить значение переменной b после выполнения фрагмента программы.

```
a:=1;  
b:=20;  
while a<8 do  
begin  
  b:=b-3;  
  a:=a*2;  
end;
```

1) $b = 16$

2) $b = 8$

3) $b = 28$

4) $b = 17$

Вопрос № 7

Определить значение переменной x после выполнения фрагмента программы.

```
x:=0;  
y:=3;  
repeat  
  x:=2*x-y;  
  y:=y+2;  
until y>10
```

1) $x = -74$

2) $x = -65$

3) $x = 47$

4) $x = -47$

Каждую задачу опишите по следующем
этапам:

1. **Исходные данные:**

Требуется найти:

2. Соотношения, связывающие исходные данные и резуль-
тат:

3. **Блок-схема:**

4. **Программа на языке Паскаль:**

Задачи

- Вычислите периметр и площадь стены , на которой расположены дверь и окно.
- Даны действительные числа x и y , не равные друг другу. Меньшее из этих двух чисел увеличить на 4, а большее – в 4 раза.
- Найдите сумму и количество целых чисел в диапазоне $[1;50]$ кратных 5.

Задача на условный оператор

- Определите, является ли заданное целое число A нечётным числом.
- Определите, имеется ли среди заданных целых чисел A, B, C хотя бы одно чётное.
- Даны три числа. Выберите те из них, которые принадлежат заданному отрезку $[a, b]$.
- Для заданных вещественных чисел a, b и c определите, имеет ли уравнение $ax^2 + bx + c = 0$ хотя бы одно вещественное решение.
- Вычислите площадь кольца, ширина которого равна H , а отношение радиуса большей окружности к радиусу меньшей окружности равно D .
- Заданы площади круга и квадрата. Определите, поместится ли квадрат в круге.