

1 курс

ПЛАН – КОНСПЕКТ
проведения занятий (лекционного, практического) по дисциплине
«Информатика»

Раздел . «Основы алгоритмизации и программирования.»

**Тема : «Общие принципы построения базовых
алгоритмических структур в среде программирования.»**

часть 6

Подготовил: преподаватель
В.Н. Борисов

Вопросы занятия:

1. Алгоритмическая структура «Ветвление».
2. Оператор условного перехода.
3. Неполная и полная формы оператора условного перехода.
4. Программирование условного алгоритма (практическое занятие , теоретическая часть).

Время проведения занятий (лекционного, практического) – 4 часа.

Первый вопрос: Алгоритмическая структура «Ветвление».

Ветвление – в зависимости от справедливости проверяемого условия (да или нет) алгоритм может пойти по одной из двух возможных ветвей. Происходит выбор одного из альтернативных путей работы алгоритма. Каждый из путей ведет к общему выходу так, что работа алгоритма будет продолжаться независимо от того, какой путь будет выбран.

Структура «ветвление» существует в трех основных вариантах:

- 1) если – то;
- 2) если – то – иначе;
- 3) выбор.

The screenshot shows a PDF document page titled "1.4. Базовая структура «ветвление»". The text on the page describes the branching structure and lists three main variants. The first variant, "1) если – то;", is illustrated by a flowchart. The flowchart starts with a diamond-shaped decision node labeled "Условие". An arrow labeled "да" points from the top of the diamond to a rectangular action node labeled "Действие". An arrow labeled "нет" points from the right side of the diamond to a line that joins the bottom of the "Действие" node, indicating that both paths lead to the same final step.

Below the flowchart, the text "2) если – то – иначе;" is visible. The right sidebar of the Adobe Acrobat Reader contains various PDF tools, including "Экспорт PDF", "Редактировать PDF", "Создать PDF", "Добавить комментарий", and "Объединить файлы".

2) если – то – иначе;

```

graph TD
    A{Условие} -- да --> B[Действие 1]
    A -- нет --> C[Действие 2]
    B --> D[ ]
    C --> D
  
```

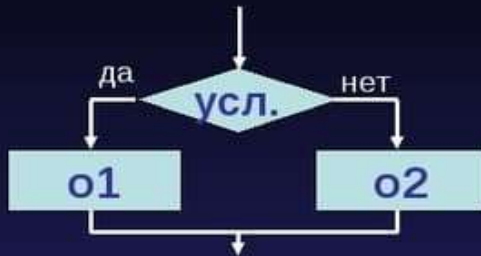
3) выбор.

```

graph TD
    A{Условие 1} -- да --> B[Действие 1]
    A -- нет --> C{Условие 2}
    C -- да --> D[Действие 2]
    C -- нет --> E{Условие 3}
    E -- да --> F[Действие N]
    E -- нет --> A
    B --> G[ ]
    D --> G
    F --> G
  
```

Второй вопрос: Оператор условного перехода.

Условный оператор на языке Pascal



полная

If <условие>

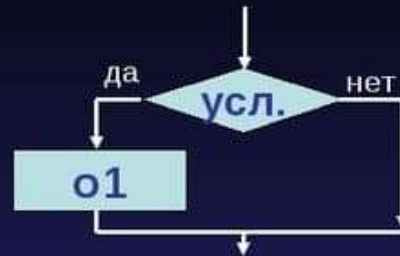
(если)

then <оператор1>

(то)

else <оператор2>;

(иначе)



неполная

If <условие>

then <оператор1>;

Условный оператор



Условные операторы в QBasic и Turbo Pascal 7.0 помогают нам осуществить "ветвление" программы, т.е. передать управление по условию.

Условный оператор имеет вид:

IF условие THEN <операторы1> [ELSE <операторы2>]

Выполнение условного оператора начинается с вычисления значения логического выражения, записанного в условии. Простые условия записываются в виде равенств или неравенств. Сложные условия составляют из простых с помощью логических операций.

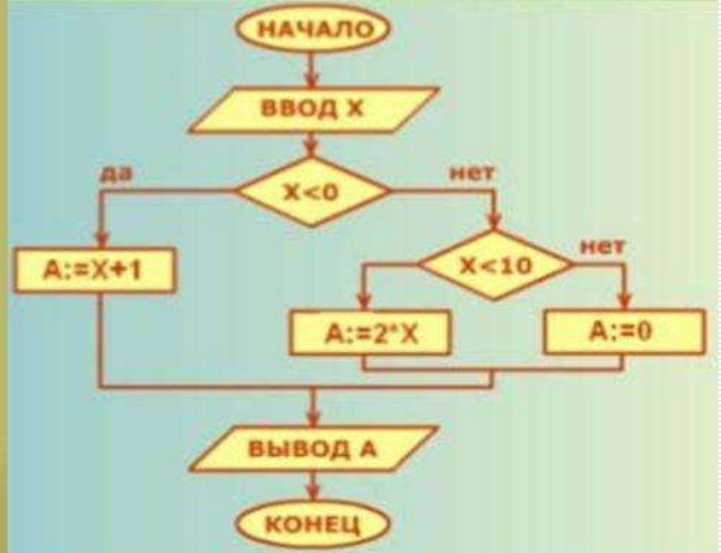
Если условие истинно, то выполняется <операторы1>, в противном случае -<операторы2>.

Программа, соответствующая этому алгоритму, выглядит так

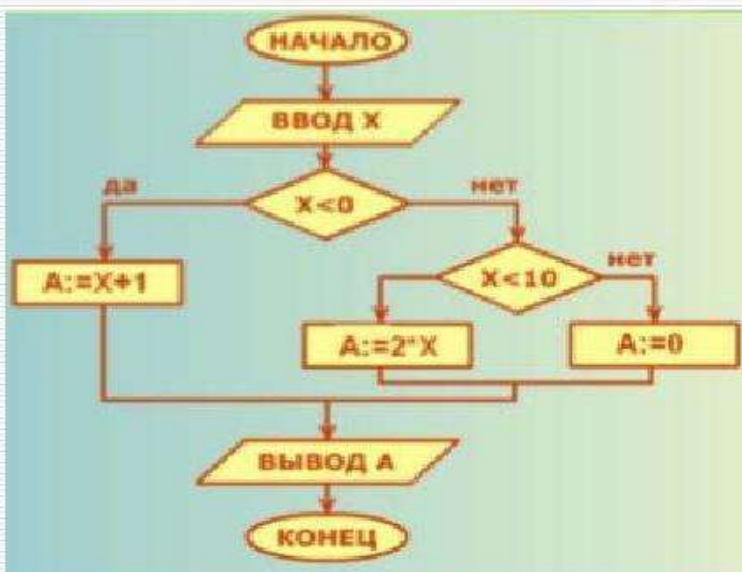
```

Program Calculate_A;
Var a, x: integer;
BEGIN
  Writeln('Введите x');
  Readln(x);
  If (x<0)
    then a:=x+1
    else if (x<10)
      then a:=2*x
      else a:=0;
  Writeln('A = ',a);
  Readln;
END.

```



Блок-схема, соответствующая этому алгоритму, выглядит так



$$A = \begin{cases} x + 1, & x < 0 \\ 2 \cdot x, & 0 \leq x < 10 \\ 0, & x \geq 10 \end{cases}$$

Составной условный оператор

Если в качестве оператора должна выполняться серия операторов, то они объединяются в операторные скобки **begin-end**.

IF условие THEN

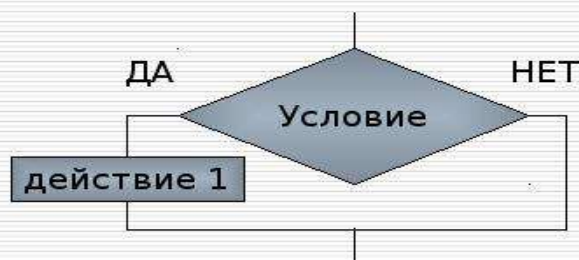
BEGIN действие1; действие 2; END

ELSE

BEGIN действие3; действие 4; END;

Третий вопрос: Неполная и полная формы оператора условного перехода.

Условный оператор (неполное ветвление)



IF условие THEN действие1;

Условный оператор (полное ветвление)



**IF условие THEN действие1
ELSE действие2;**

В качестве условий используются логические выражения,
например, такие:

(C=D)

(a>b) and (a>c) or (a=d)

Условный оператор **if...then...else** полная форма

Синтаксис:

if <логическое выражение>
then <оператор1>
else <оператор2>;

Семантика:

1. Вычисляется значение <логического выражения>
2. Если <логическое выражение> истинно (TRUE), то выполняется оператор1, иначе выполняется оператор2.



Пример:

If2. Дано целое число **N**.

Если оно положительное, то прибавить к нему 1; если отрицательно вычесть из него 2. Вывести полученное число.

```

program if_2;
var n: integer;
begin
write ('введите целое число n=');
readln (n);
if n>0
then n:=n+1
else n:=n-2;
writeln ('n=',n);
end.
  
```

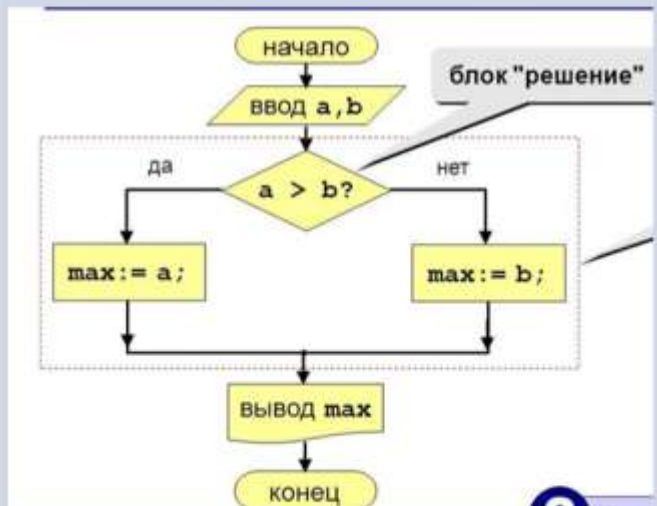

Задача на Паскаль. Условные операторы.

Задание №1

Вывести на экран наибольшее из двух чисел

```
Program MaxOfTwo;  
Var a,b:integer;  
Begin  
  readln(a,b)  
  if a>b then begin  
    writeln(a)  
  end  
  else begin  
    writeln(b)  
  end  
End.
```

Даны два числа. Вывести на экран то из них, которое больше



Вложенные условные операторы

Возможно использование в качестве оператор1 или оператор 2 других условных операторов:



if <условие1> **then**

if <условие2> **then** <оператор1>

else <оператор2>;

Задача 1. Составить программу на языке программирования Turbo Pascal: найти значение функции $y = \sqrt{x-2}$



```

program znach_funk;
uses crt;
var x,y:real;
begin clrscr;
  writeln('vvedite x');
  readln(x);
  if x >= 2 then begin
    y:=sqrt(x-2);
    writeln('y=',y:3:2)
  end
  else writeln('значение y не
  существует');
  readln;
end.
  
```

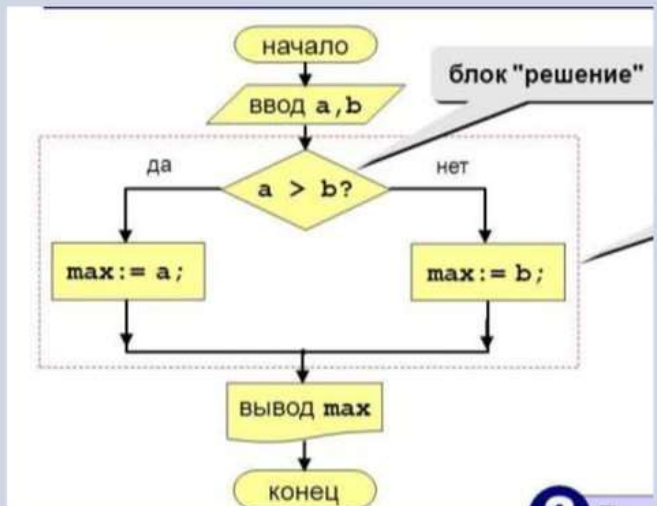
Четвертый вопрос: Программирование условного алгоритма (практическое занятие, теоретическая часть).

Задача на Паскаль. Условные операторы. Задание №1

Вывести на экран наибольшее из двух чисел

```
Program MaxOfTwo;  
Var a,b:integer;  
Begin  
  readln(a,b)  
  if a>b then begin  
    writeln(a)  
  end  
  else begin  
    writeln(b)  
  end  
End.
```

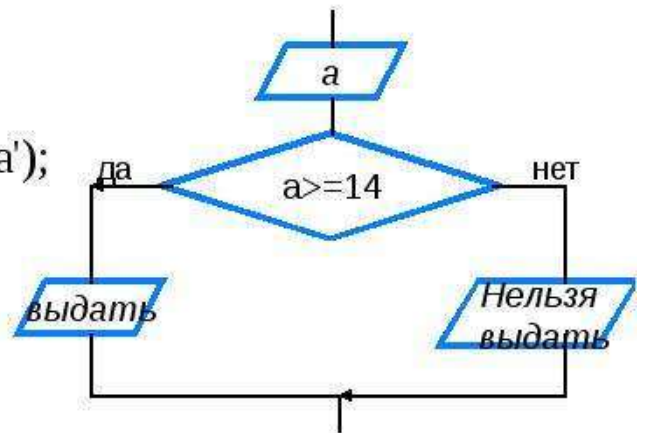
Даны два числа. Вывести на экран то из них, которое больше



Условный оператор

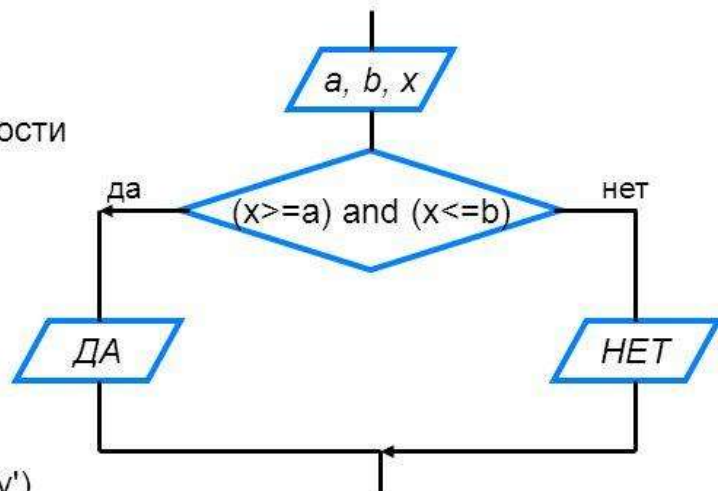
Пример 1: Составить программу для выдачи паспорта ребенку впервые (паспорт выдается при достижении 14 лет).

```
program n_1;  
  var a: real;  
begin  
  writeln ('Введите возраст ребенка');  
  readln (a);  
  if (a>=14) then  
    writeln ('Выдать паспорт')  
  else  
    writeln ('Паспорт выдавать нельзя')  
end.
```



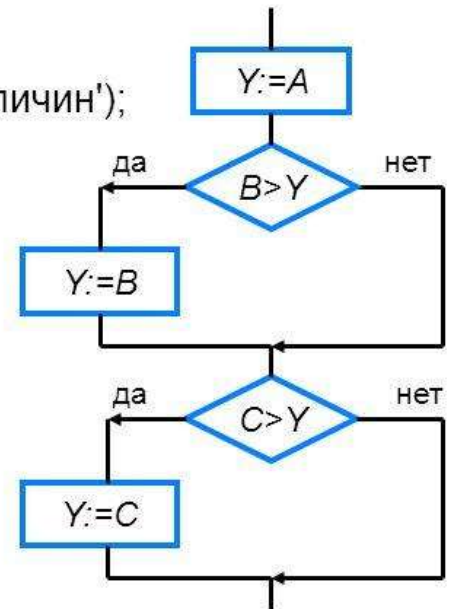
Условный оператор

```
program n_9;  
  var x, a, b: real;  
begin  
  writeln ('Определение принадлежности  
           точки отрезку');  
  write ('Введите a, b>>');  
  readln (a, b);  
  write ('Введите x>>');  
  readln (x);  
  if (x>=a) and (x<=b) then  
    writeln ('Точка принадлежит отрезку')  
  else writeln ('Точка не принадлежит отрезку')  
end.
```



Неполная форма условного оператора

```
program n_10;  
  var y, a, b, c: integer;  
begin  
  writeln ('Нахождение наибольшей из трёх величин');  
  write ('Введите a, b, c>>');  
  readln (a, b, c);  
  y:=a;  
  if (b>y) then y:=b;  
  if (c>y) then y:=c;  
  writeln ('y=', y)  
end.
```



Примеры решения задач (записать в тетрадь)

Задача 2. Даны три величины A, B, C . Переменной Y присвоить значение большей из них.

Разберемся в условии задачи:

Понятно, что значения нужно сравнить. Предположим, что наибольшим является значение A – сохраним его в переменной Y . Затем сравним с Y переменную B . Если B больше по значению – сохрани его в Y . Аналогично поступим с переменной C . Выводим значение Y .

```

program Z2;
var
  y, a, b, c: integer;
begin
  writeln ('Нахождение max(A, B, C)');
  write ('Введите a, b, c>>');
  readln (a, b, c);
  y:=a;
  if (b>y)
    then y:=b;
  if (c>y)
    then y:=c;
  writeln ('y=', y);
  readln;
end.
  
```

Задача решена с использованием двух кратких форм условного оператора

