

Разветвляющиеся вычислительные процессы

Программирование разветвляющейся структуры.

Цель работы: закрепить знания условного оператора *if*, оператора переходов *go to*, научиться составлять программы разветвляющейся структуры.

Порядок выполнения работы

- 1. Составить программу на языке Паскаль, дать проверить её преподавателю.*
- 2. Выполнить отладку программы, счет по программе. Записать результаты в отчёт.*
- 3. Оформить отчёт и сделать выводы по работе.*
- 4. Составить блок-схему решения задачи.*

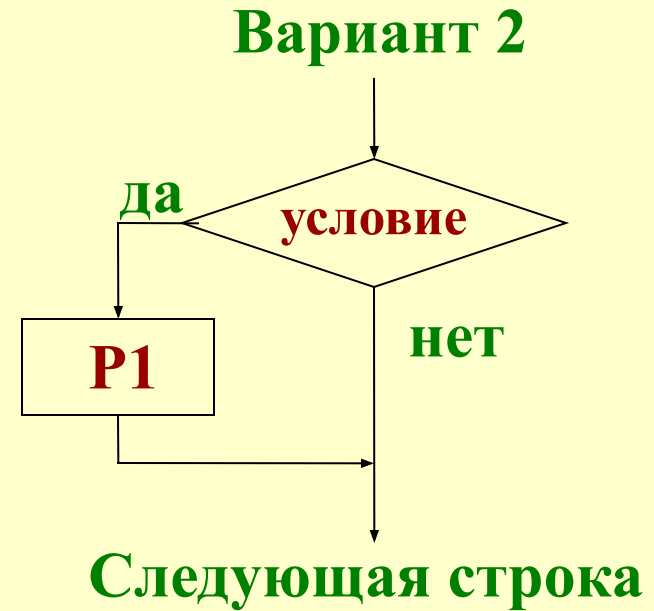
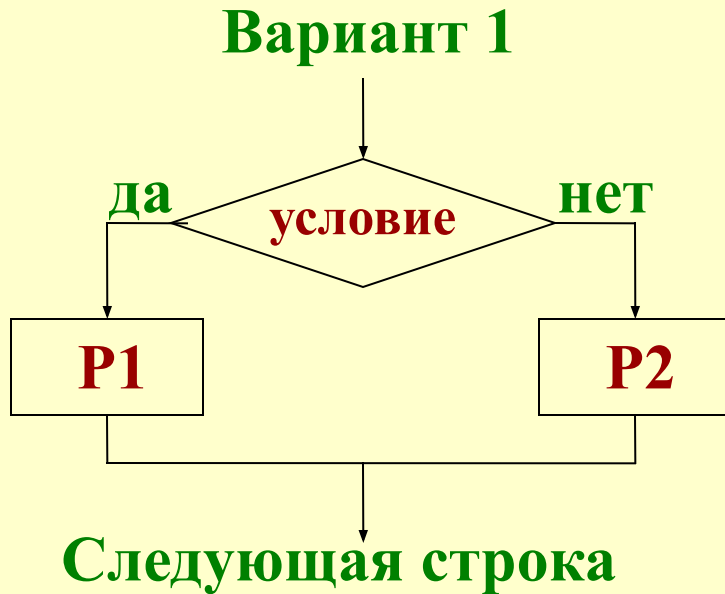
Условный оператор *if*

Общий вид оператора

If условие then оператор p_1 Else оператор p_2;

И по ветке *then* и по ветке *Else* выполняется только один оператор

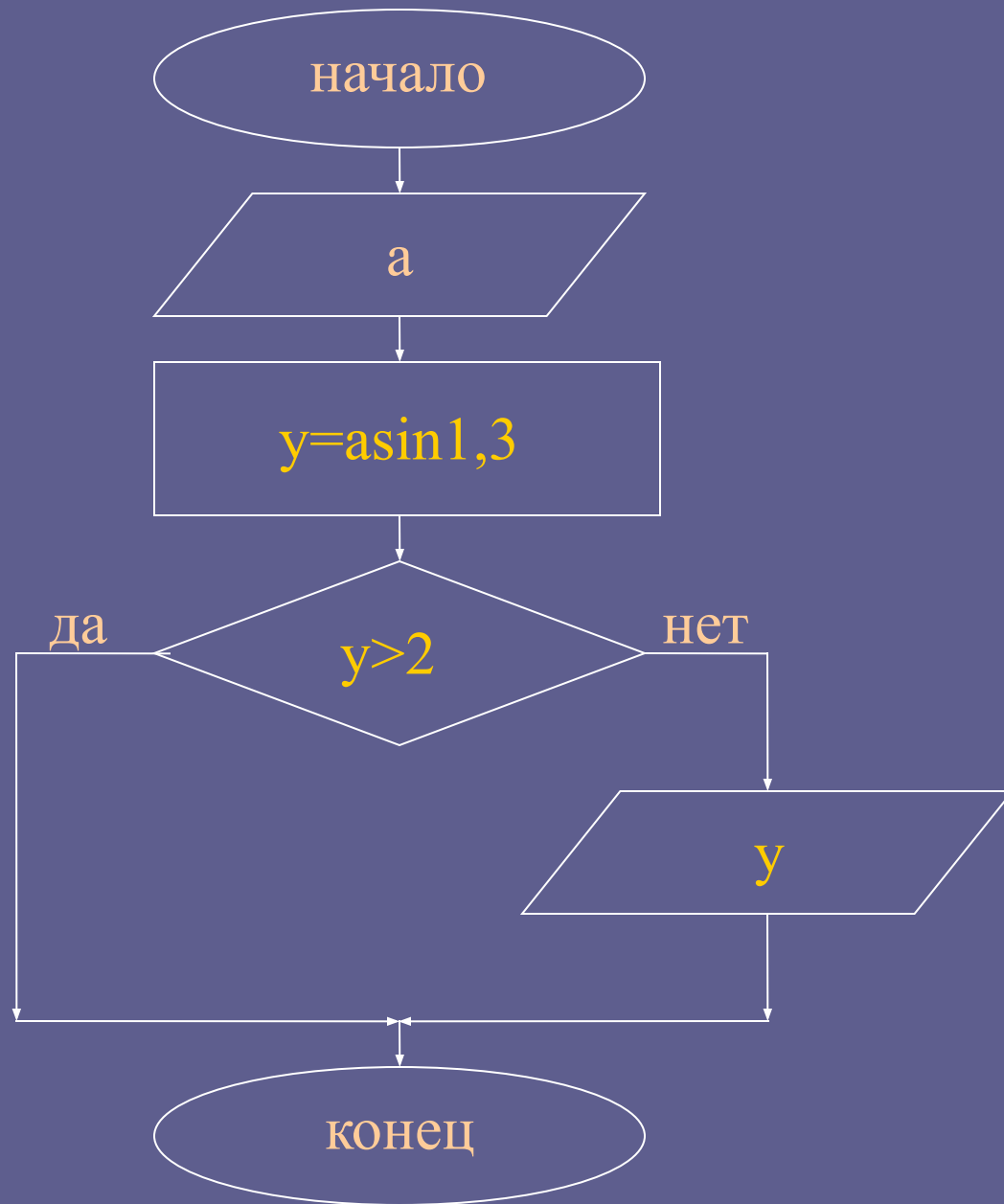
Работа условного оператора: *работа каждого варианта оператора определяется соответствующей схемой.*



Подчеркнём: *после выполнения оператора(ов) P1 или P2 во всех случаях обеспечивается переход к следующей строке.*

Пример:

Вычислить $y = a \sin 1,3$ при $a=3$. Если $y > 2$, закончить задачу. В противном случае перейти к печати результата

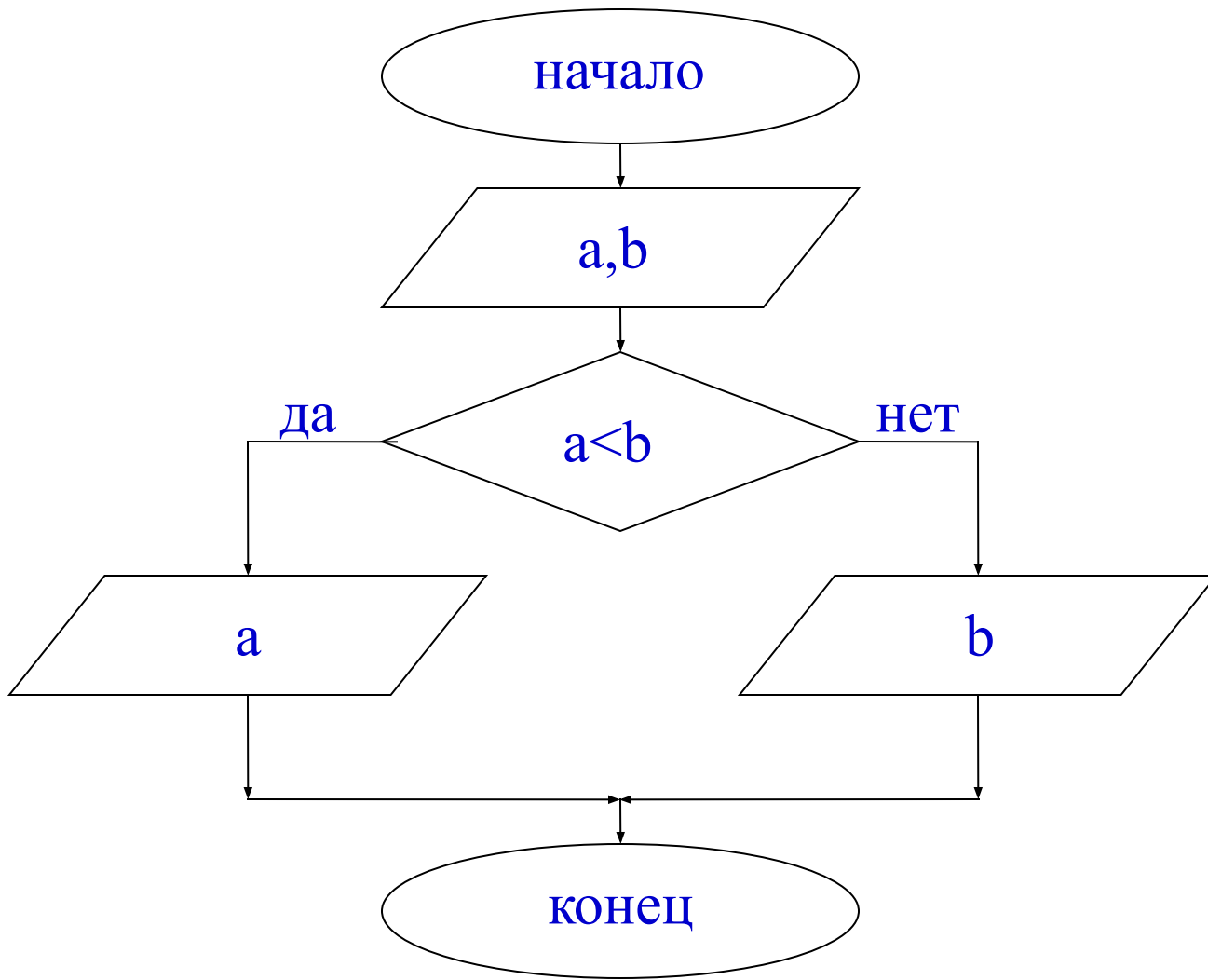


Программа (случай, когда действия по ветке Else отсутствуют)

```
Program p_3;  
Var  
    a, y: real;  
Begin  
Writeln ( 'a=' ); readln (a);  
    Y:=a*sin(1.3);  
    If y<2 then write ( 'y=', y:4:2)  
    Else  
end.
```


Пример:

Найти минимальную из
двух величин



Программа

(действия по обеим веткам присутствуют)

```
Program p_4;  
Var  
    A, b: real;  
Begin  
    Readln (a);  
    Readln (b);  
    If a<b then  
        Writeln ('a меньшая', a)  
    Else  
        Writeln ('b меньшая', b)  
End.
```

$$y = \begin{cases} \ln x, & x > 0 \\ e^x, & x < 0 \end{cases}$$

Конструкция *If – then - else*

- Конструкция *If – then - else* является одним оператором (внутри которого могут содержаться другие операторы), поэтому перед *else* нельзя ставить символ (;), так как это будет означать окончание работы оператора.
- Работает он следующим образом. Если значение логического выражения истинно (*True*), то выполняется *оператор 1*, если ложно (*False*), то выполняется *оператор 2*. Далее в любом случае выполняется оператор, стоящий первым за этой конструкцией.

Структура оператора if

- Случай, когда по ветке **then** и по ветке **Else** выполняются несколько операторов

If условие **then**

Begin

 Оператор_1;

 Оператор_2;

 ...

 Оператор_n

End

Else

Begin

 Оператор_1;

 Оператор_2;

 ...

 Оператор_n

end

-

Квадратное уравнение

Program kv;

Var

A,b,c,d,x,x1,x2: real;

begin

Readln (a, b, c);

*D:=b*b-4*a*c;*

If d=0 then

begin

*Write ('корень один', -b/(2*a));*

End

Else

If d>0 then

Begin

Write ('корни действительные');

*x1:=(-b-sqrt(d))/(2*a); x2:=(-b+sqrt(d))/(2*a);*

Writeln ('x1=', x1:4:2);

Writeln ('x2=', x2:4:2);

End

Else

Begin

Write ('корни мнимые');

End

End.

Случай вложения условных операторов друг в друга

- ***If условие then***

If подусловие then

Begin

.....

end

else

begin

.....

end

else

begin

.....

end;

При вложениях условных операторов всегда действует правило: альтернатива else считается принадлежащей ближайшему условному оператору if, имеющему ветвь else.

Логические операции в конструкции *If – then - else*

- ▶ Помимо логических отношений, в конструкции *If – then - else* используются *логические операции*.
- ▶ *Пример*: можно ли построить треугольник из отрезков заданной длины x, y, z ($x > 0, y > 0, z > 0$).
- ▶ Запись в программе:
- ▶ *If (x + y > z) And (x + z > y) And (y + z > x) then*
- ▶ *Writeln('треугольник построить можно')*
- ▶ *else*
- ▶ *Writeln ('треугольник построить нельзя');*
- ▶ В данном случае оператор `Writeln` позволяет вывести на экран дисплея нужное сообщение. Следует обратить внимание, что *логические выражения*, связываемые через логические операции, всегда заключаются в *круглые скобки*.

Пример: Найти наибольшее из трёх чисел.

```
Program max_3;  
  var  
    A, b, c: real;  
begin  
  Writeln ('b='); readln (b);  
  Writeln ('a='); readln (a);  
  Writeln ('c='); readln (c);  
  If (a>b) and (a>c) then  
  begin  
    Writeln ('a большая', a)  
  end  
  Else  
  begin  
    If (b>a) and (b>c) then  
      Writeln ('b большая', b)  
    else  
      Writeln ('c большая', c)  
  end  
end.
```

Оператор варианта Case

- ▶ Оператор варианта *Case* является более общим условным оператором, чем *If-then-else*. Он дает возможность выполнить один из нескольких операторов (или групп операторов) в зависимости от значения некоторого выражения. В общем случае оператор имеет вид:
- ▶ *Case* <выражение> *of*
- ▶ <список значений 1>: <оператор_1>;
- ▶ <список значений 2>: <оператор_2>;
- ▶ <список значений 3>: <оператор_3>;
- ▶
- ▶ <список значений n>: <оператор_n>
- ▶ *else*
- ▶ альтернативный оператор
- ▶ *end*;

Оператор варианта Case

- Оператор варианта работает следующим образом. Если *выражение* принимает значение из *списка значений 1*, то выполняется *оператор_1*. Если *выражение* принимает значение из *списка значений 2*, то выполняется *оператор_2*.. Если *выражение* принимает значение из *списка значений n*, то выполняется *оператор_n*. Если *выражение* не принимает ни одно значение из имеющихся списков значений, то выполняется альтернативный оператор. Каждый оператор, идущий за двоеточием, отделяется от следующего списка значений точкой с запятой. Ветвь *else*, отвечающая всем неперечисленным значениям выражения, необязательна. В списках значение оператора *case* допустимыми являются целые и некоторые другие (но не вещественные) типы.

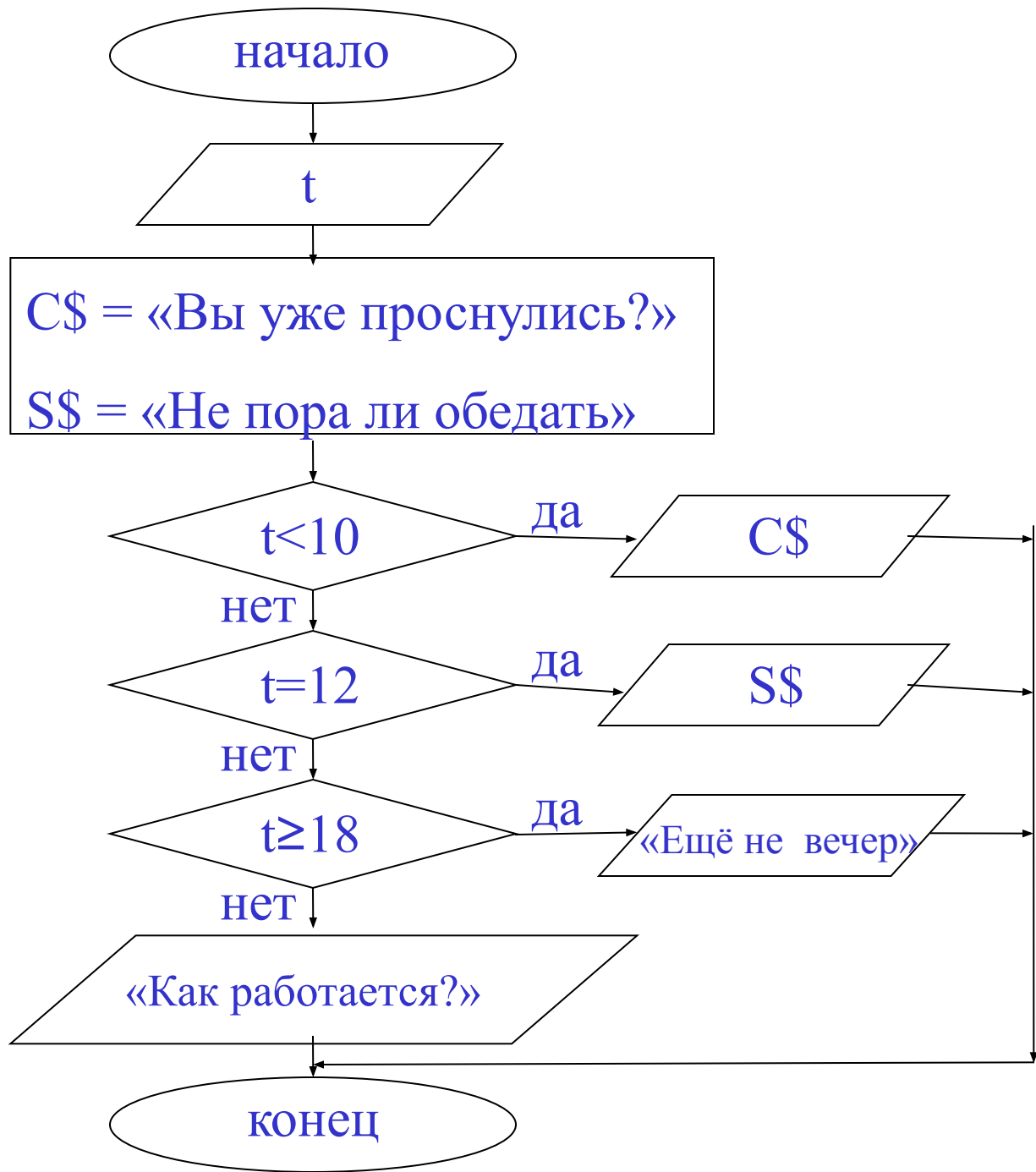
Составить программу ввода величины t - времени в течении суток, и выдачи текста:

«Вы уже проснулись?», если $t < 10$

«Не пора ли обедать?», если $t = 12$

«Ещё не вечер», если $t \geq 18$

«Как работается?», в остальных случаях.



Программа

■ Program vremja_1;

Var

T: integer;

Begin

writeln ('t='); readln (t);

case t of

8, 9: writeln ('вы уже проснулись?');

12: writeln ('не пора ли обедать?');

18, 19: writeln ('ещё не вечер')

Else

writeln ('как работается?');

End;

End.

Программирование разветвляющейся структуры.

Контрольные вопросы.

1. Какой алгоритм называется разветвляющимся?
2. Какие блоки используются при построении схем разветвляющихся алгоритмов?
3. Какой блок называется логическим?
4. Каково назначение условного оператора?
5. Какой формат имеет условный оператор *IF*?
6. Как записывается полная форма условного оператора?
7. Можно ли после оператора `_1` в структуре *If-then-else* ставить «;»?
8. Как записывается краткая форма условного оператора?
9. Какую структуру имеет составной оператор?
10. Какую структуру имеет оператор выбора?

Литература

- Учебное пособие по информатике для студентов 1 курса под ред. Чекановой Н.Н. с. 75 - 87