Алгоритмизация и программирование разветвляющихся алгоритмов.

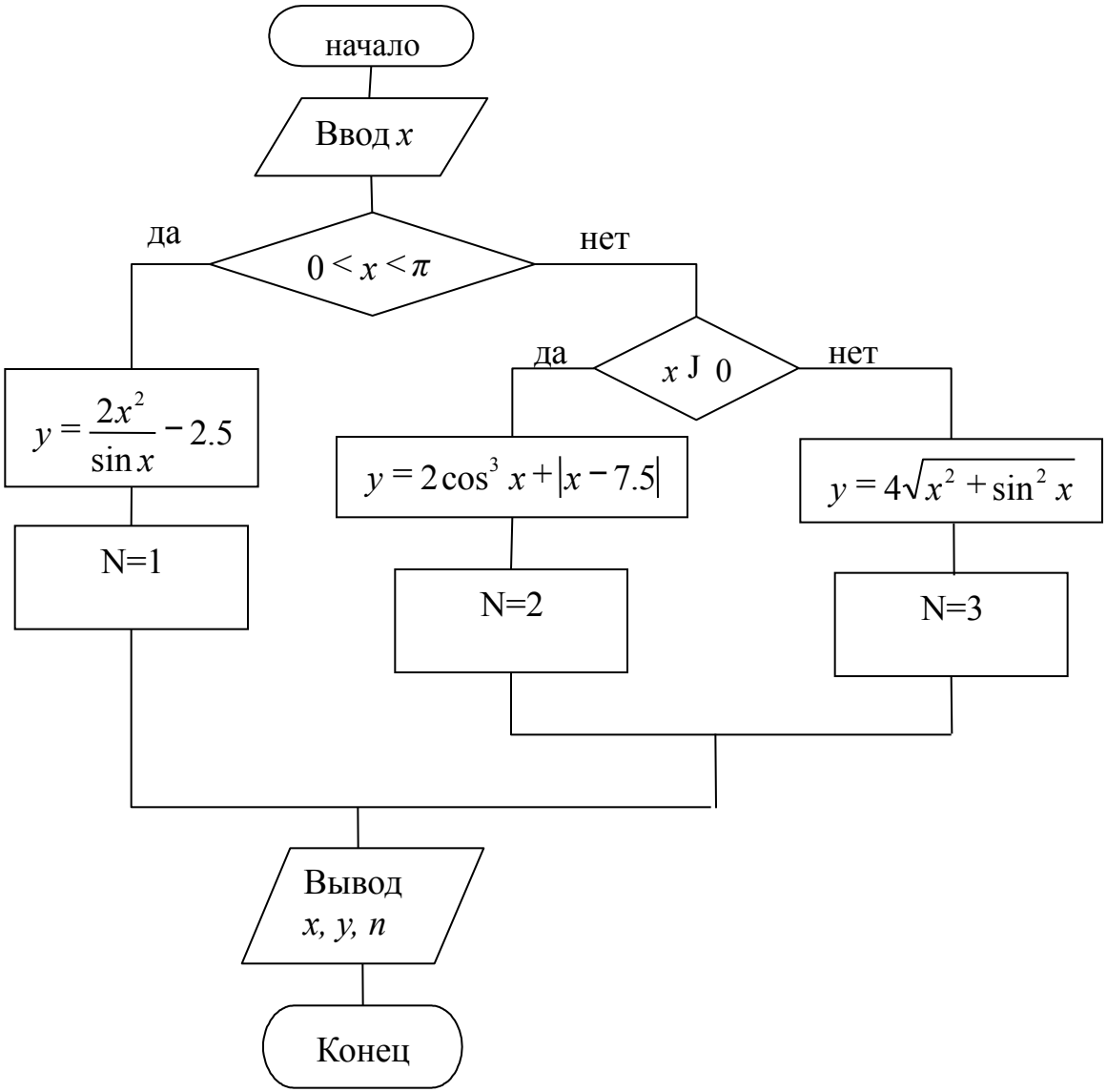
1. Примеры разветвляющихся алгоритмов

Разветвляющийся - алгоритм, в котором некоторые действия выполняются один раз или не выполняются совсем в зависимости от заданного условия.

Пример 1. Составить алгоритм вычисления функции.

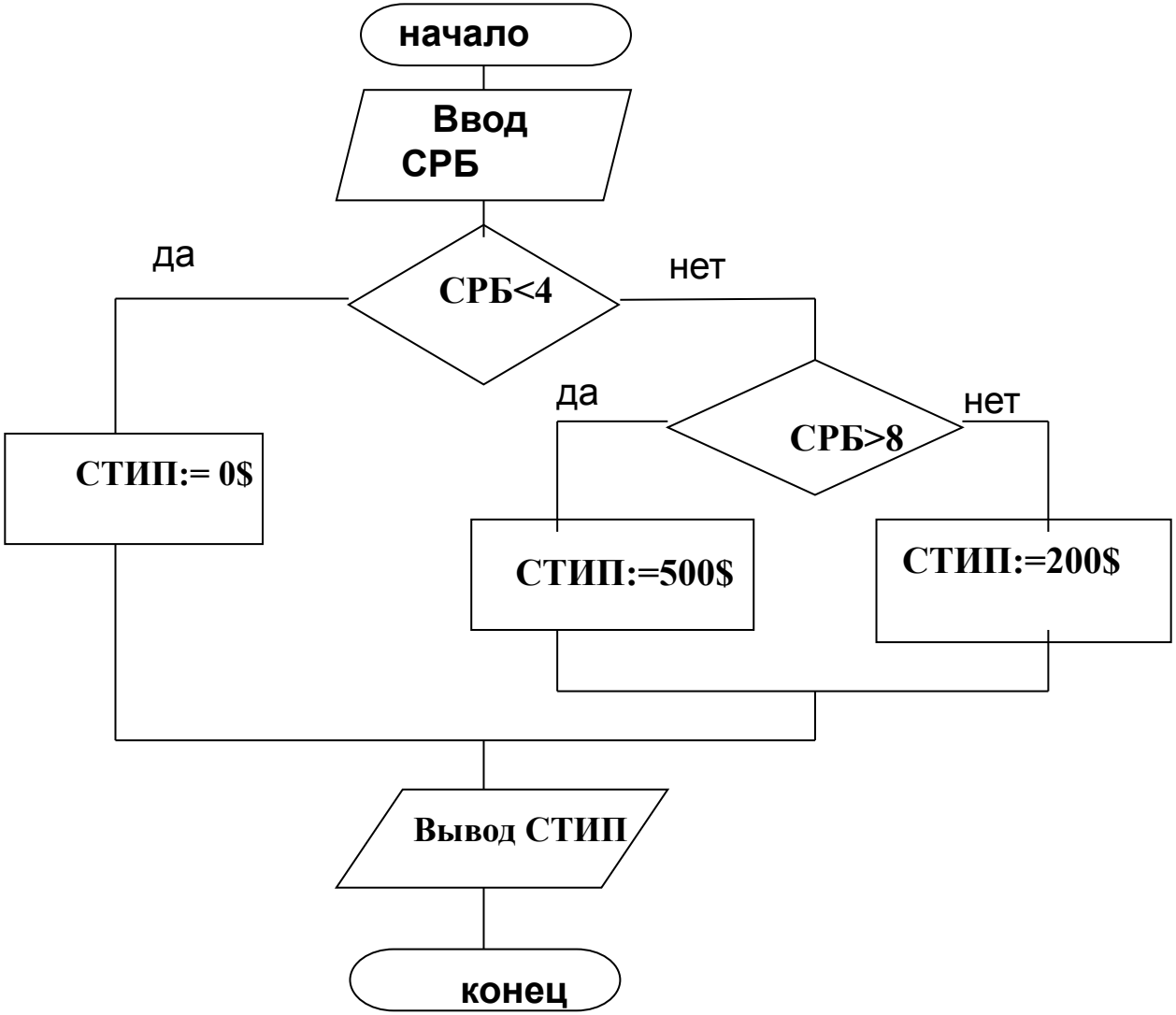
$$y = \begin{cases} \dfrac{2x^2}{\sin x} - 2{,}5 & \textit{если} \quad 0 < x < \pi \\ 2\cos^3 x + \left|x - 7{,}5\right| & \textit{если} \quad x \le 0 \\ 4\sqrt{x^2 + \sin^2 x} & \textit{в остальных случаях} \end{cases}$$

Предусмотреть вывод номера расчетной формулы.


начало
Ввод x
да
$0 < x < \pi$
нет
да
x J 0
нет
$y = \frac{2x^2}{\sin x} - 2.5$
$y = 2\cos^3 x + |x - 7.5|$
$y = 4\sqrt{x^2 + \sin^2 x}$
N=1
N=2
N=3
Вывод
x, y, n
Конец

Пример2. Примером разветвляющегося алгоритма может служить алгоритм начисления стипендии по среднему баллу.

- в качестве исходного данного задается значение среднего балла сдачи сессии студеном;
- если средний балл меньше **4**, то стипендия – **0$**;
- если средний балл больше **8**, то начисляется стипендия в **500$**;
- в остальных случаях начисляется стипендия размером в **200$**;
- выводится значение начисленной стипендии.


начало
Ввод СРБ
СРБ<4
да
нет
СТИП:= 0$
СРБ>8
да
нет
СТИП:=500$
СТИП:=200$
Вывод СТИП
конец

2. Элементы языка программирования, необходимые для реализации разветвляющегося алгоритма.

Логические выражения.

Логическое выражение – любое выражение, возвращающее логическое значение (true или false).

При составлении логического выражения могут быть использованы все виды операций , в том числе операции отношения и логические операции.

Операции отношения
предназначены для сравнения двух величин.
Результат сравнения имеет логический тип.

< -меньше

<= - меньше или равно

> - больше

>= - больше или равно

<> - не равно

= -равно

Логические операции.

Применяются к величинам логического типа.

Результат тоже логический.

and (и), or (или), not (не).

Таблица истинности not : пусть A и B – некоторые логические выражения.

A	Not A
true	false
false	true

Таблица истинности and и or

A	B	A and B	A or B
true	true	true	true
true	false	false	true
false	true	false	True
false	false	false	false

Например, пусть a:=3 b:=7

not(a>b) =true not(b>a)=false

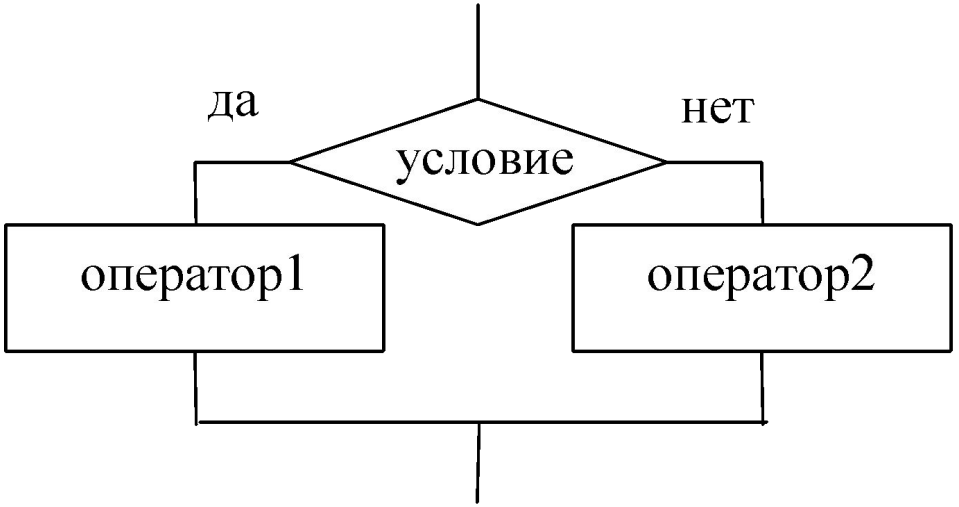
Примеры логических выражений:

- *(x+1)<y*
- *A>=B*
- *Name1=Name*
- *Sin(x+1)>(x+2)/3*
- *((a>0) or (b<0)) and (c<>0)*

Условный оператор.

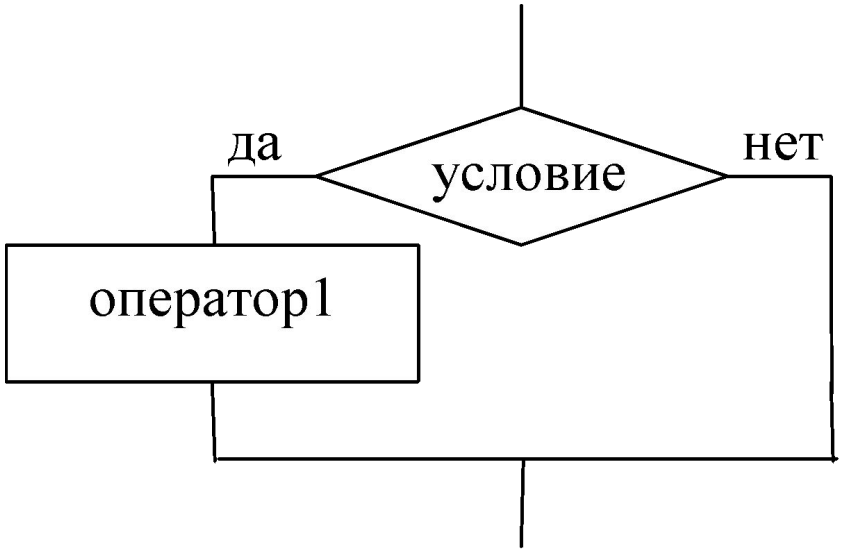
Реализует алгоритмическую конструкцию Ветвление и изменяет порядок выполнения операторов в зависимости от истинности или ложности некоторого условия.

а) Полная форма



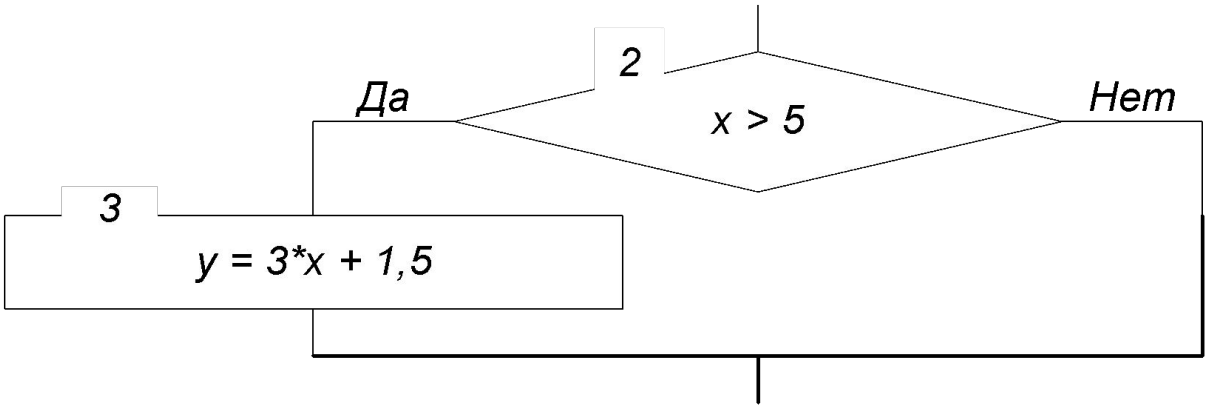

```
if <условие> then
   <оператор 1>
else
    <оператор 2>;
```

а) Сокращенная форма

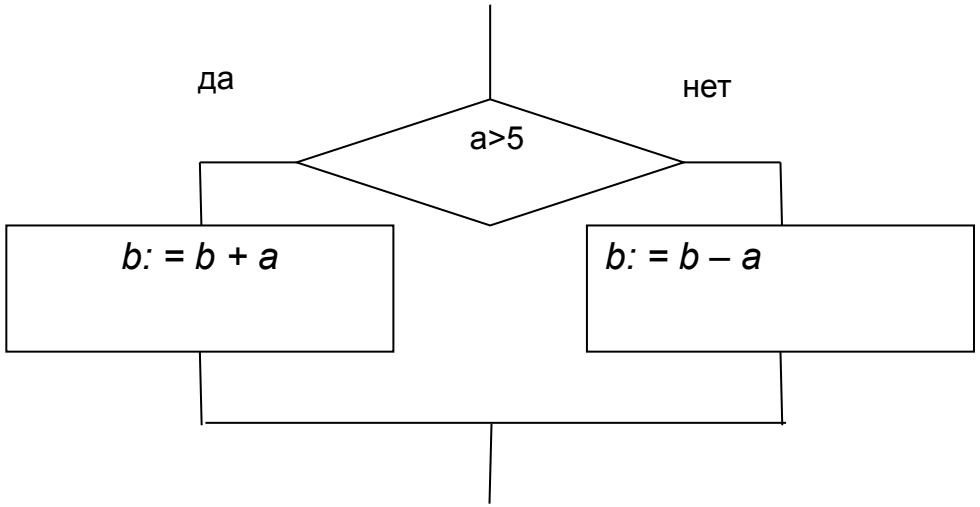



```
if <условие> then
    <оператор >;
```

Например,




```
If  x>5 Then y:=3*x+1.5;
```




```
If a > 5 Then
        b: = b + a
    Else
        b: = b – a;
```

Условный оператор выполняется следующим образом.

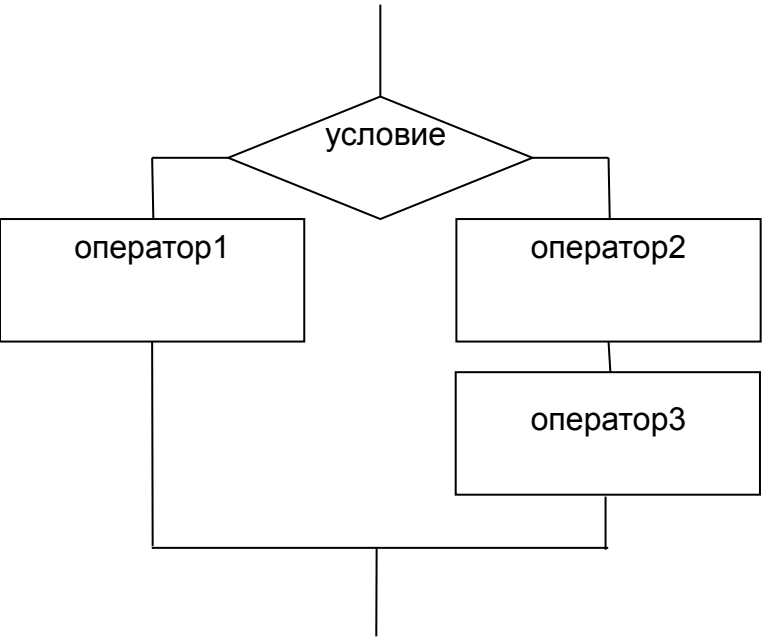
- **Сначала вычисляется выражение, стоящее в условии.**
- **Если значение выражения равно true, выполняется оператор, стоящий после слова *Then*, а оператор, стоящий после слова *Else* игнорируется.**
- **Если значение выражения равно false, выполняется оператор, стоящий после слова *Else*, а оператор, стоящий после слова *Then* игнорируется.**

Составной оператор.

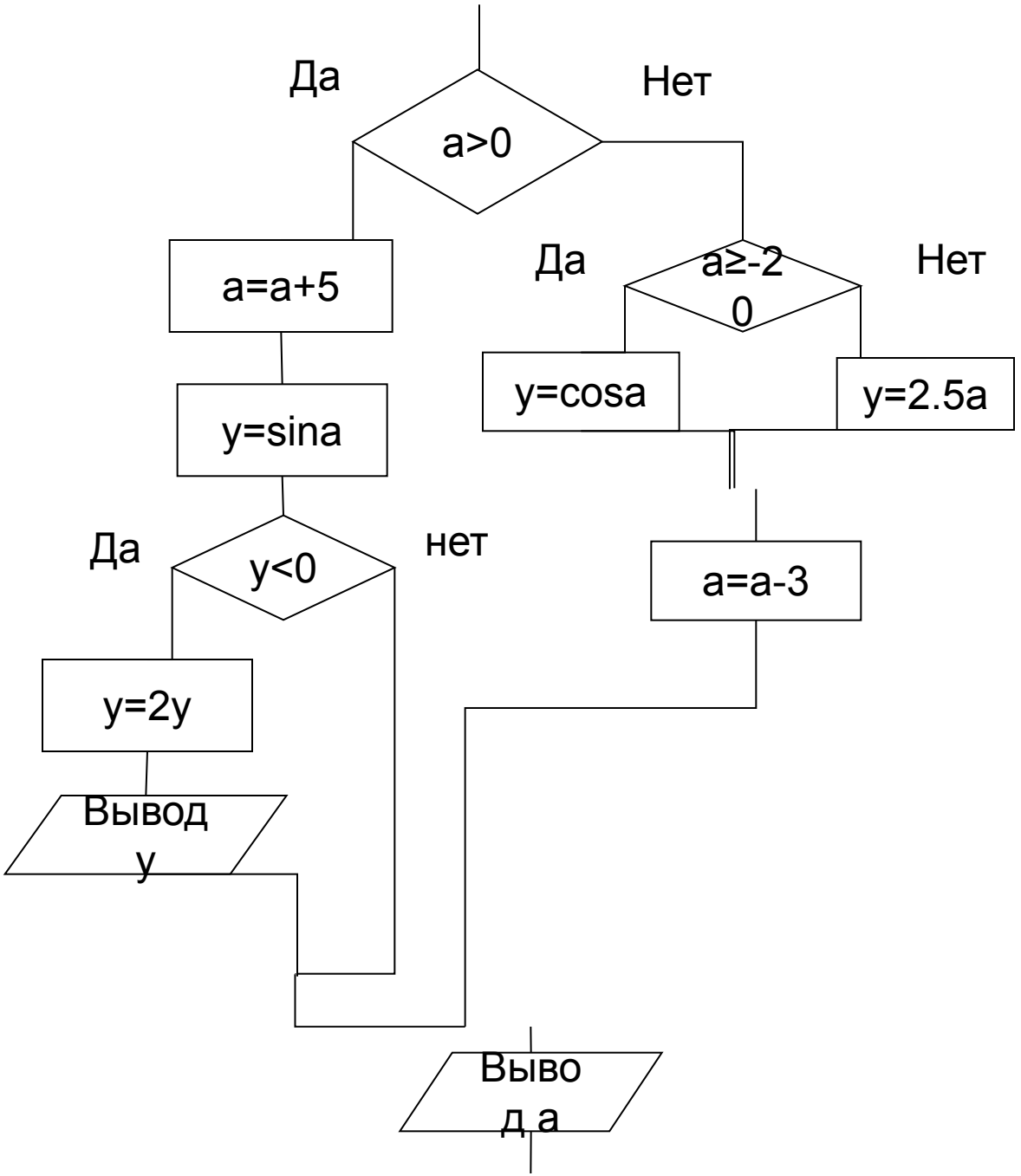
Составной оператор представляет собой совокупность последовательно выполняемых операторов, заключенных в операторные скобки **begin** и **end**.

```
begin
  <оператор 1>;
 <оператор 2>;
  …
  <оператор n>
end;
```

Он нужен в тех случаях, когда в соответствии с правилами построения конструкций языка можно использовать один оператор, а выполнить нужно несколько действий.




```
if <условие> then
        <оператор 1>
  else
      begin
          <оператор 2>;
           <оператор 3>
      end;
```


Да
Нет
a>0
a=a+5
Да
a≥-2
0
Нет
y=cosa
y=2.5a
y=sina
Да
y<0
нет
a=a-3
y=2y
Вывод
y
Выво
д a

```
If a>0 then
  begin
    a:=a+5; y:= sin(a);
    If y <0 then
      begin
        y:= 2*y;
        Writeln('y=',y)
      end
  end
 else
    begin
      If a>= -20 then y:=cos(a) else y:=2.5*a;
      a:=a-3
    end;
 Writeln('a=',a);
```

3. Составление программы

Составим программу для примера 1. Сначала подберем имена для всех переменных, которые будут использоваться в программе. Эти имена должны соответствовать правилам формирования идентификаторов.

Таблица соответствия переменных

Имя переменной в условии задачи	Имя переменной в программе	Тип	Комментарии
x	x	вещественный	Аргумент функции
y	y	вещественный	Значение функции
	n	вещественный	Номер формулы

```
program Project2;
  {$APPTYPE CONSOLE}
uses
   SysUtils;
{Раздел описания переменных}
var
    x,y:real;
    n:integer;

begin
  {Ввод исходных данных}
    write(' vvedite x');
    readln(x);
```

```
{Вычисление значения функции}
   if (x>0) and (x<pi) then
    begin
      y:=2*sqr(x)/sin(x)-2.5; n:=1
    end
    else
     if x<=0 then
        begin
          y:=2*sqr(cos(x))*cos(x)+abs(x-7.5);
          n:=2
        end
       else
         begin
          y:=4*sqrt(x*x+sqr(cos(x))); n:=3
         end;
```

```
{Вывод исходных данных и результатов}
   writeln('  x=',x:6:2,'  y=',y:7:3);
   writeln('Raschet proveden po formule ',n);


 {Остановка выполнения программы до нажатия
клавиши
         ENTER }
   readln
end.
```

Тесты для проверки:

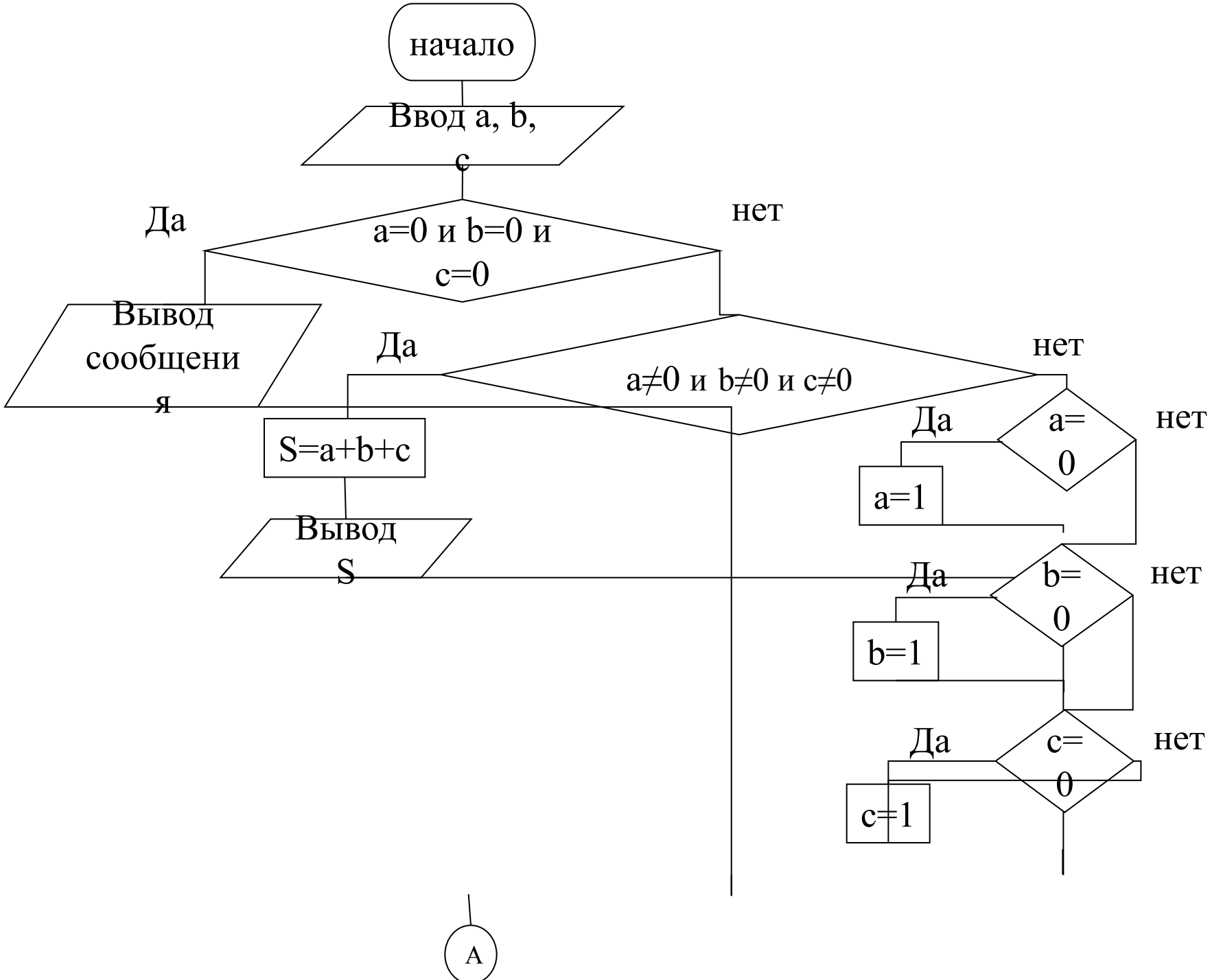
x = -1	*y = 8.815*	*n = 2*
x = 0	*y = 9.5*	*n = 2*
x =1	*y = -0.123*	*n = 1*
x = 3.14	*y = 13.188*	*n = 3*
x = 5	*y = 20.032*	*n = 3*

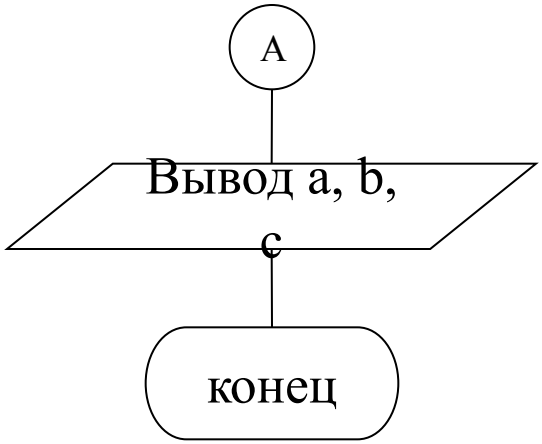
Пример 3.

Даны числа a, b, c. Если все они равны нулю, вывести об этом сообщение, если среди чисел есть нули, заменить их единицами, в противном случае найти и вывести сумму исходных чисел.

Таблица соответствия переменных

Имя переменной в условии задачи	Имя переменной в программе	Тип	Комментарии
a	a	real	Исходное число
b	b	real	Исходное число
c	c	real	Исходное число
	S	real	Сумма чисел


начало
Ввод a, b, c
Да
a=0 и b=0 и c=0
нет
Вывод сообщения
Да
a≠0 и b≠0 и c≠0
нет
S=a+b+c
Вывод S
Да
a= 0
нет
a=1
Да
b= 0
нет
b=1
Да
c= 0
нет
c=1
A




```
program Project2;
  {$APPTYPE CONSOLE}
uses
   SysUtils;

{Раздел описания переменных}
var
   a,b,c,S:real;
```

```
begin
  {Ввод исходных данных}
    write(' vvedite tri chisla');
    readln(a,b,c);
    if (a=0)and(b=0)and(c=0) then
      writeln ('vse chisla ravny nulju')
    else
      if (a<>0) and (b<>0) and (c<>0)  then
        begin
          S:=a+b+c;
          writeln('summa chisel =',s:5:2)
        end
```

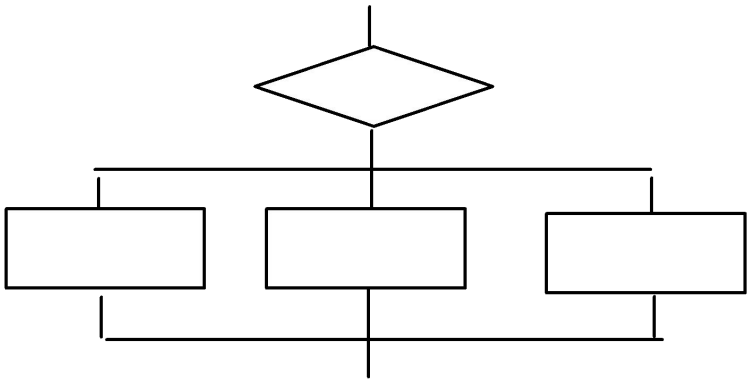
```
    else
      begin
          if a=0 then a:=1;
          if b=0 then b:=1;
          if c=0 then c:=1;
       end;
writeln('a =',a:5:2,' b =',b:5:2,' c =',c:5:2);
   readln

end.
```

Оператор выбора.

Оператор выбора позволяет выбрать один из нескольких возможных вариантов продолжения программы.

Реализует алгоритмическую структуру «**Выбор**»



Общий вид:

Case <ключ выбора> of

<список выбора 1>: <оператор 1>;

<список выбора 2>: <оператор 2>;

…

<список выбора N>: <оператор N>

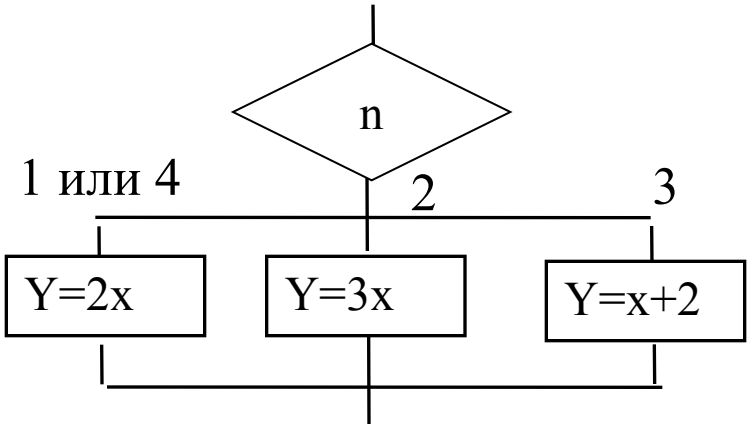
[Else <оператор>]

End;

Ключ выбора - это выражение целого, логического или символьного типа.

Список выбора содержит перечисленные через запятую константы того же типа, что и ключ выбора.

Например,




```
case   n   of
    1, 4 :  y:=2*x;
    2      :  y:=3*x;
    3      :  y:=x+2;
  end;
```

Оператор выбора работает следующим образом.

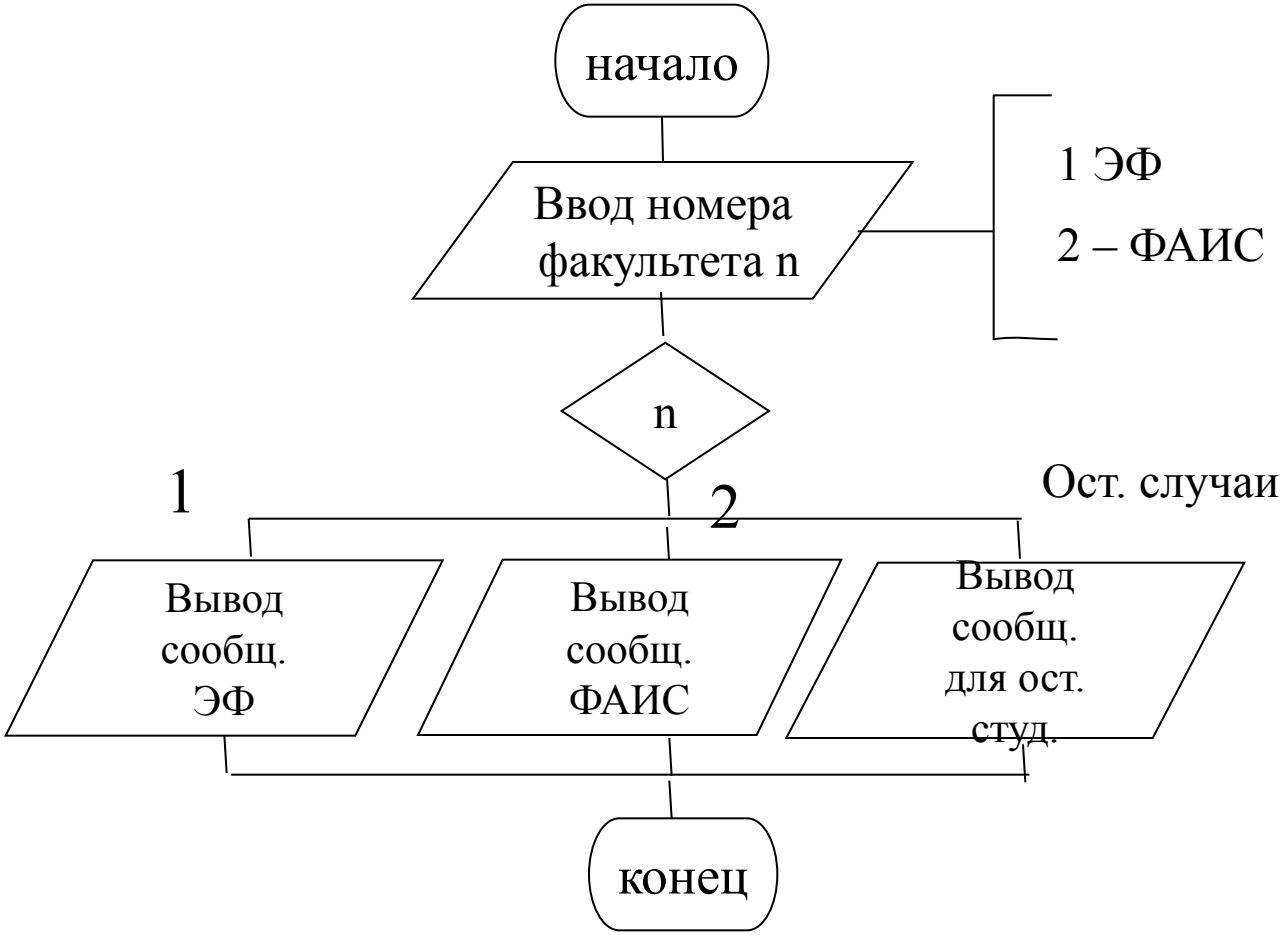
Сначала вычисляется значение выражения **<ключ выбора>**, затем в списках выбора отыскивается константа, равная вычисленному значению, и выполняется оператор, соответствующий списку выбора с найденной константой.

После этого оператор выбора завершает работу.

Если в списках выбора не будет найдена подходящая константа, управление передается операторам, стоящим после слова **else.**

Если часть **else** отсутствует, то при отсутствии в списках выбора нужного значения оператор case завершит свою работу.

Пример. Составить программу, которая доводит до сведения студентов распоряжение деканата.



```
program Project2;
  {$APPTYPE CONSOLE}
uses
   SysUtils;

var
    n:byte;

begin
 writeln('На каком факультете Вы учитесь?');
 writeln(' 1 – ЭФ, 2 - ФАИС');

 Readln(n);
```

```
Case n of
  1:writeln('Вам увеличили стипендию на 100$! ');

  2:writeln('Всем привет от деканата !!!')
 Else writeln('А Вы с какого факультета?');
End;

End.
```