Задачи С1

Что нужно знать:

* *Условный оператор* **if–else** служит для организации ветвления в программе на языке Паскаль. Он может иметь полную или неполную форму:

```
if a = b then begin
{ блок-1 }
 end
 else begin
{ блок-2 }
 end;
```

```
if a = b then begin
{ блок-1 }
 end;
```

Допустимы такие операторы:

```
if a = b then
    c:=1
else c:=0;
```

```
if a = b then c:=1;
```

```
if a = b then begin
  c:=1;
end
else c:=0;
```

А вот такие операторы недопустимы:

```
if a = b then begin
    c:=1
 else c:=0;
```

```
if a = b then
  c:=1;
 end
 else c:=0;
```

```
if a = b then
   c:=1;
   d:=1;
else x:=1;
```

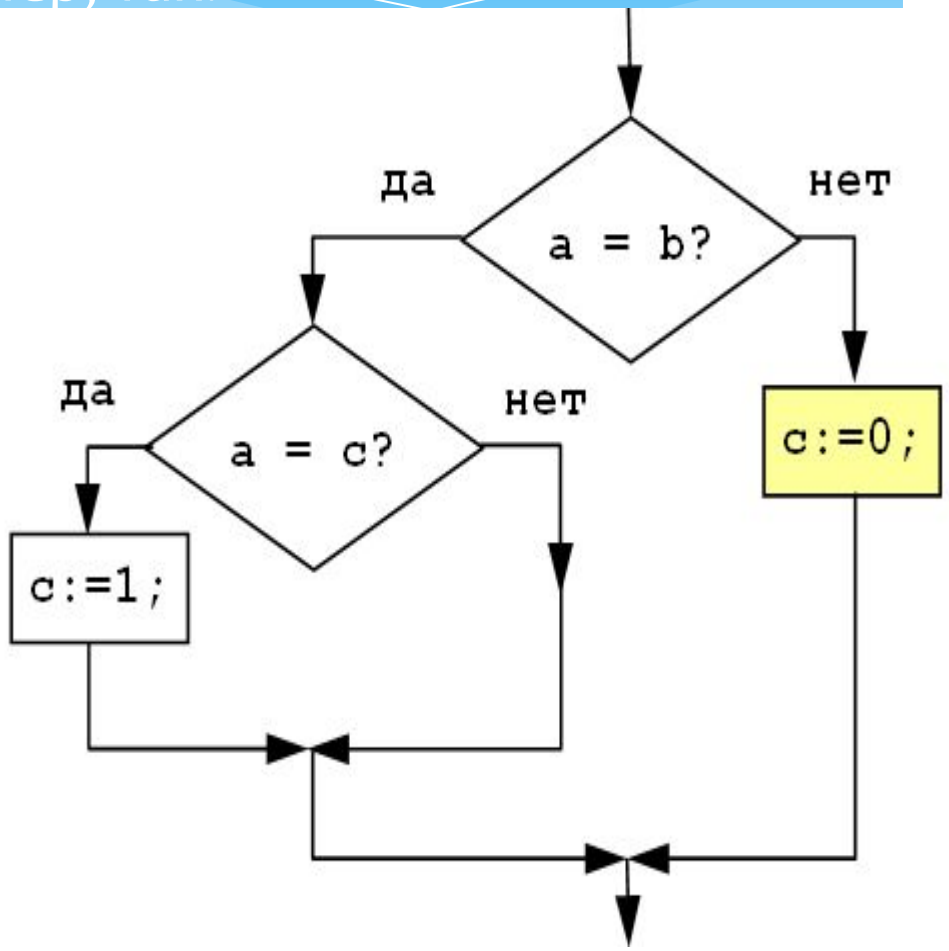
Условный оператор может находиться внутри другого условного оператора, как в блоке-1, так и в блоке-2; например:

```
if a = b then begin
    { блок-10 }
    if a = c then begin
        { блок-11 }
    end
    else begin
        { блок-12 }
    end;
end
else begin
    { блок-2 }
end;
```

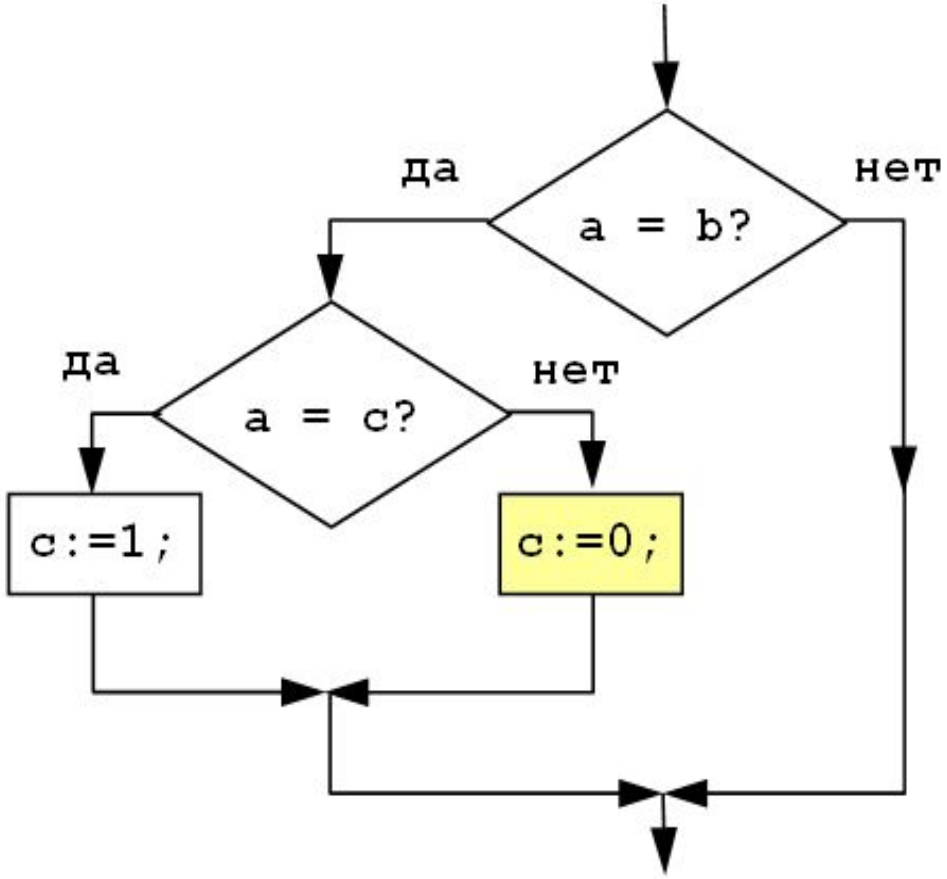
Ключевая тема этого задания ЕГЭ – использование вложенных условных операторов, причем в тексте задания фрагмент программы обычно записан без отступов «лесенкой» или с неправильными отступами, например, так:

```
if a = b then begin
  if a = c then
    c:=1;
  end
  else c:=0;
```

Перед **else** стоит **end**, поэтому нужно найти соответствующий ему **begin**; таким образом определяем, что **else** относится к первому (внешнему) условному оператору.



```
if a = b then
   if a = c then
      c:=1
else c:=0;
```

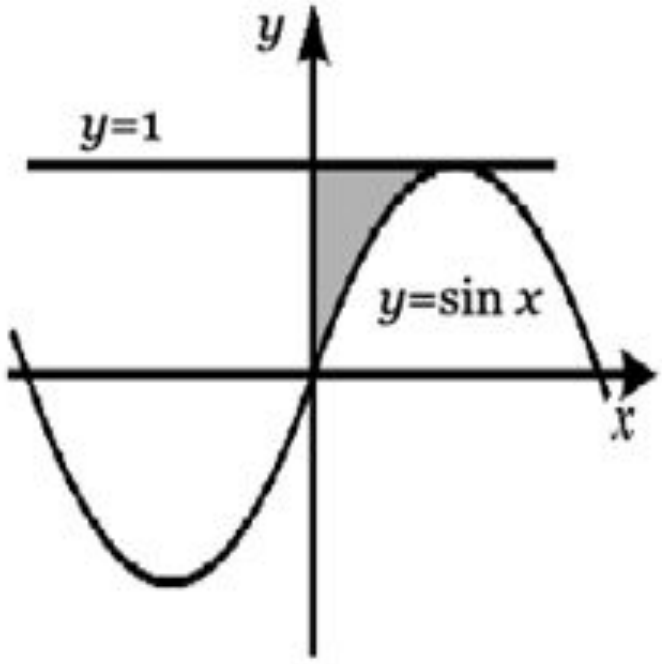


Перед **else** нет **end,** поэтому он относится к ближайшему по тексту внутреннему условному оператору

Пример 1.

*Требовалось написать программу, которая вводит с клавиатуры координаты точки на плоскости (**x, y** – действительные числа) и определяет принадлежность точки заштрихованной области, включая ее границы. Программист торопился и написал программу неправильно. Вот она:*

```
var x,y: real;
begin
readln(x,y);
if y <= 1 then
  if x >= 0 then
    if y >= sin(x) then
      write('принадлежит')
else  write('не принадлежит')
end.
```



Последовательно выполните следующее:

1) Приведите пример таких чисел **x**, **y**, *при которых программа неверно решает поставленную задачу.*

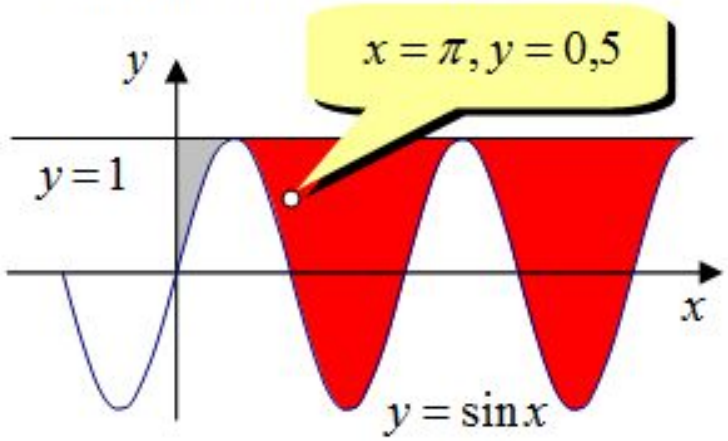
2) Укажите, как нужно доработать программу, чтобы не было случаев ее неправильной работы. (Это можно сделать несколькими способами, поэтому можно указать любой способ доработки исходной программы) .

Решение:

1) сначала лучше отложить в сторону программу и попытаться написать условие, которым должны отвечать точки, попавшие в выделенную область
2) заштрихованная область ограничена по координате x, она находится
   - справа от оси y, что равносильно условию $x \geq 0$ (с учетом границы здесь и далее получаем нестрогие неравенства)
   - слева от первого максимума функции $y = \sin x$; из математики мы знаем, что эта функция достигает максимума при $x = \frac{\pi}{2}$, поэтому получаем второе условие $x \leq \frac{\pi}{2}$
3) заштрихованная область ограничена с двух сторон по координате y: она находится
   - ниже линии $y = 1$, откуда следует третье условие $y \leq 1$
   - выше линии $y = \sin x$, что дает четвертое условие $y \geq \sin x$

Продолжение решения:

4) итак, точка находится в заданной области, если все эти четыре условия выполняются одновременно; можно предположить, что в программе нужно использовать четыре вложенных условных оператора или один условный оператор, в котором четыре простых условия (отношения $x \ge 0$, $x \le \frac{\pi}{2}$, $y \le 1$ и $y \ge \sin x$) связаны с помощью логической операции **and** («И», одновременное выполнение всех условий)
5) теперь смотрим на программу: здесь три (а не четыре!) вложенных условных оператора с простыми отношениями, поэтому явно какое-то условие не учтено; легко найти, что «забыли» условие $x \le \frac{\pi}{2}$
6) оператор **write('принадлежит')** помещен внутрь всех трех условных операторов, то есть, он выполнится тогда, когда три (а не четыре!) условия истинны;
7) отметим на рисунке область, где выполняются все нужные условия, кроме $x \le \frac{\pi}{2}$ (красная зона);
8) для всех точек, которые находятся в «красной» зоне программа выдаст сообщение «принадлежит», хотя в самом деле эти точки не принадлежит заданной области; одна из таких точек имеет координаты $x = \pi, y = 0{,}5$



Продолжение решения:

9) теперь выясним, когда программа выдает сообщение «не принадлежит»

```
if y<=1 then
  if x>=0 then
    if y>=sin(x) then
      write('принадлежит')
else  write('не принадлежит')
```

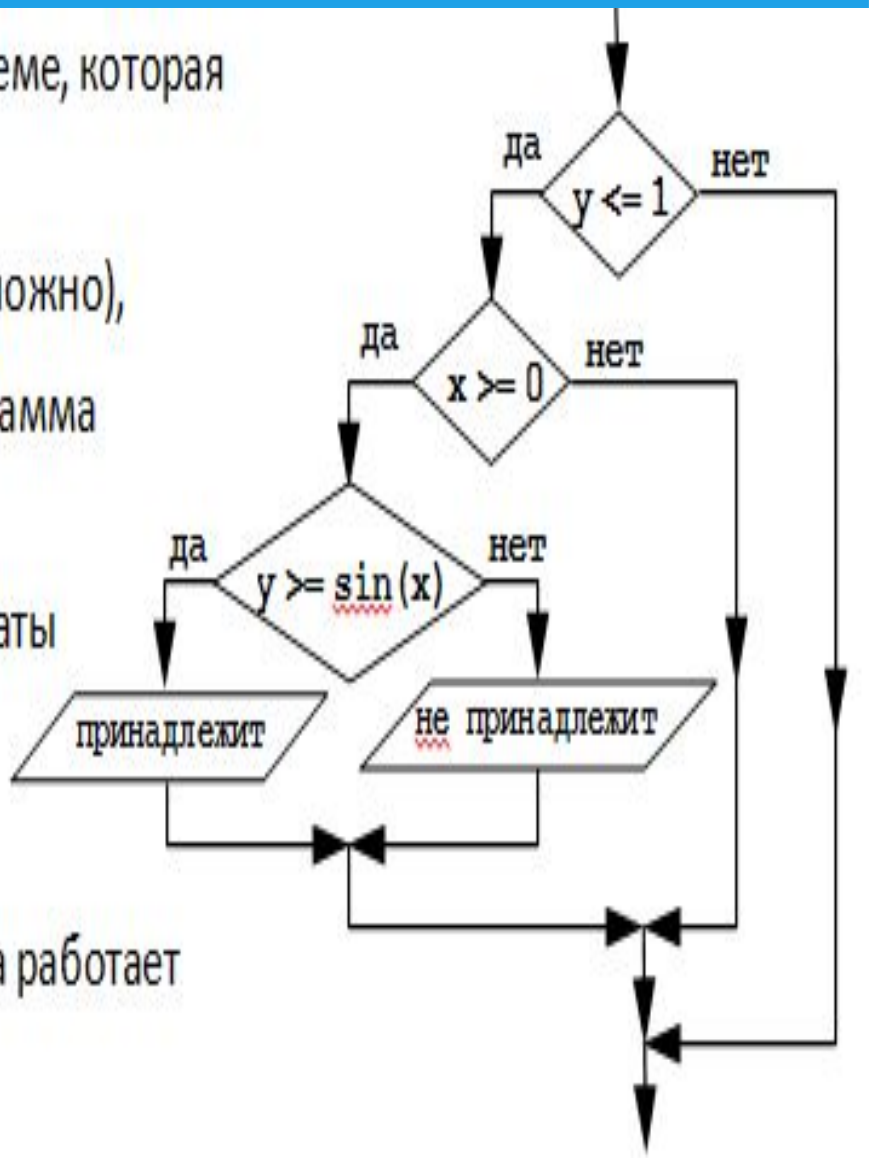
10) судя по записи «лесенкой», **else** относится к самому первому оператору **if**, однако в самом деле это не так; перед словом **else** нет **end**, поэтому ищем ближайший **if**: это самый внутренний оператор, правильная запись «лесенкой» выглядит так:

```
if y<=1 then
  if x>=0 then
    if y>=sin(x) then
        write('принадлежит')
    else write('не принадлежит')
```

Продолжение решения:

11) этот фрагмент программы соответствует блок-схеме, которая показана на рисунке справа:

12) по схеме видим, что при $y > 1$ (первое условие ложно), а также при $x < 0$ (второе условие ложно) программа вообще не выдает никакого сообщения, то есть, работает неправильно; таким образом, координаты любой точки, для которой $y > 1$ или $x < 0$, могут быть указаны в ответе как пример набора входных данных, при которых программа работает неправильно



Продолжение решения:

13) итак, первая часть ответа такова

примеры входных данных, на которых программа работает неверно:

(x=3.14, y=0.5) (неправильно определяет принадлежность точки области)

(x=0, y=1) или **(x=-1, y=0)** (не выдает вообще никакого сообщения)

14) остается исправить эту программу;

начнем с самого «лобового способа»: добавим в программу четвертый (вложенный) условный оператор, проверяющий условие $x \le \frac{\pi}{2}$, и еще три блока **else**, чтобы выводить строку «не принадлежит» в том случае, когда хотя бы один из них не сработал:

```
if x<=pi/2 then
  if y<=1 then
    if x>=0 then
      if y>=sin(x) then
          write('принадлежит')
      else write('не принадлежит')
    else write('не принадлежит')
  else write('не принадлежит')
else write('не принадлежит');
```

обратите внимание, что точка с запятой есть только после самого последнего оператора **write**, так как остальные стоят перед ключевым словом **else**, перед которым точка с запятой не ставится

Продолжение решения:

15) хотя приведенный выше метод дает работоспособную программу, она получается слишком длинная и некрасивая для такой простой задачи; достаточно сказать, что оператор **write('не принадлежит')** повторяется в тексте 4 раза

16) более элегантное решение формулируется на словах так: «точка принадлежит области, если выполняются одновременно 4 приведенных выше условия, а иначе – не принадлежит»; а вот реализация на Паскале (приведем программу-ответ целиком):

```
var x,y: real;
begin
readln(x,y);
if (x>=0) and (x<=pi/2) and
   (y<=1) and (y>=sin(x)) then
     write('принадлежит')
else write('не принадлежит');
end.
```

здесь использовано сложное условие, в котором 4 отношения связаны операциями **and** («И», требуется одновременное выполнение всех условий)

Возможные проблемы:

- как показывает анализ опубликованных задач этого типа, нужно уметь, прежде всего, разбираться в серии вложенных условных операторов в полной и неполной форме
- неправильная «лесенка» в записи сбивает с толку и подталкивает к неверному решению; чтобы разобраться в программе, лучше на черновике построить блок-схему алгоритма и правильную «лесенку»
- чтобы не запутаться, к какому оператору относится **else**, используйте следующее правило:

* если перед **else** нет слова **end**, нужно искать ближайший сверху условный оператор **if**
* если перед **else** стоит **end** (конец блока), нужно искать парный ему **begin** (начало блока) и соответствующий условный оператор **if ... then begin**
  - проверяйте, все ли необходимые условия учтены в программе, это особенно актуально для *немонотонных* функций типа синуса или косинуса (немонотонные функции на некоторых участках возрастают при увеличении аргумента, а на некоторых – убывают);

Возможные проблемы:

- не перепутайте, где нужно использовать операцию **and** («И», одновременное выполнение условий), а где – **or** («ИЛИ», хотя бы одно условие)
- нужно внимательно проверять, всегда ли программа выдает сообщение, если заданное условие не выполняется
- часто бывает полезно нарисовать блок-схему алгоритма, которая позволяет увидеть ход выполнения программы при всех возможных вариантах
- проверяйте, включает ли заданная область свои границы; если включает – в отношениях будут нестрогие неравенства (**<=, >=**), если не включает – строгие (**<, >**)
- при оценке работы можно (при абсолютно правильном решении) потерять баллы из-за синтаксических ошибок в программе (скобки, точки с запятой, неправильное написание оператора и т.п.); не забывайте, что
- в сложном условии все простые условия (отношения) нужно брать в скобки, так как в Паскале отношения при вычислении логического выражения имеют самый низкий приоритет
- перед **else** точка с запятой никогда не ставится
- в конце программы после последнего **end** ставится точка

За что снимают баллы:

- неправильно определены входные данные, при которых исходная программа работает неверно
- исправлены не все ошибки в программе, например, легко «просмотреть», что необходимо еще условие
- программа работает правильно в большем количестве случаев, чем исходная, но не для всех возможных исходных данных
- перепутаны знаки < и >, логические операции **or** и **and**
- неверно расставлены операторные скобки **begin-end**

* синтаксические ошибки (знаки пунктуации – запятые, точки, точки с запятой; неверное написание ключевых слов); чтобы получить 3 балла, нужно при абсолютно правильном решении сделать не более одной синтаксической ошибки; на 2 балла – до двух ошибок, на 1 балл – до трех ошибок

Задача 1 (для тренировки)

Требовалось написать программу, которая решает уравнение «ax+b=0 » относительно **x** для любых чисел **a** и **b**, введенных с клавиатуры. Все числа считаются действительными. Программист торопился и написал программу неправильно:

```
var a, b, x: real;
begin
readln(a,b,x);
if b = 0 then
write('x = 0')
else
if a = 0 then
write('нет решений')
else
write('x =',-b/a);
end.
```

Последовательно выполните три задания:
1) Приведите пример таких чисел **a**, **b**, **x**, при которых программа неверно решает поставленную задачу.
2) Укажите, какая часть программы является лишней.
3) Укажите, как нужно доработать программу, чтобы не было случаев ее неправильной работы. (Это можно сделать несколькими способами, поэтому можно указать любой способ доработки исходной программы).

Задача 2 (для тренировки)

Требовалось написать программу, которая решает уравнение $«a|x| = b»$ относительно x для любых чисел **a** и **b**, введенных с клавиатуры. Все числа считаются действительными. Программист торопился и написал программу неправильно:

```
var a,b,x: real;
begin
readln(a,b,x);
if a = 0 then
if b = 0 then
write ('любое число')
else write ('нет решений')
else
if b = 0 then
write('x = 0')
else write('x =',b/a,' или x =',-b/a);
end.
```

Последовательно выполните три задания:
1) Приведите пример таких чисел a, b, x, при которых программа неверно решает поставленную задачу.
2) Укажите, какая часть программы является лишней.
3) Укажите, как нужно доработать программу, чтобы не было случаев ее неправильной работы. (Это можно сделать несколькими способами, поэтому можно указать любой способ доработки исходной программы).

Задача 3 (для тренировки)

Требовалось написать программу, которая определяет, лежит ли точка A (x_0,y_0) внутри треугольной области, ограниченной осями координат и прямой («внутри» понимается в строгом смысле, т.е. случай, когда точка A лежит на границе области, недопустим). В результате программа должна выводить соответствующее текстовое сообщение. Программист сделал в программе ошибки.

```
var x0, y0, y: real;
begin
readln (x0, y0);
if (x0 < 2)then begin
if (x0 > 0)then begin
y = 2 – x0;
if (y0 < y) then
writeln ('точка лежит внутри области')
else writein ('точка не лежит внутри области');
end
else writeln ('точка не лежит внутри области');
end
else writeln ('точка не лежит внутри области');
end.
```

* Последовательно выполните задания:
* Приведите пример таких чисел x_0 и y_0, при которых программа неверно решает поставленную задачу.
* Укажите, как нужно доработать программу, чтобы не было случаев ее неправильной работы (можно указать любой способ доработки исходной программы).
* Укажите, как можно доработать программу, чтобы вместо вложенных операторов **IF** она содержала логическую операцию **AND**.

Задача 4 (для тренировки)

* Требовалось написать программу, которая решает уравнение относительно x для действительных чисел **a**, **b**, **c**, введенных с клавиатуры, о которых заведомо известно, что **a≠0**, **b≠0** и **c≠0**. Была написана следующая программа:

```
var a, b, c, D, xl, x2: real;
begin
readln(a, b, c, xl, x2);
D := b*b - 4*a*c;
if D > 0
then begin
xl := (-b + sqrt(D))/(2*a);
x2 := (-b - sqrt(D))/(2*a);
write('xl =', xl);
write('x2 =', x2); end
else writeln ('действительных корней нет');
end.
```

Известно, что программа написана с ошибками. Последовательно выполните три задания:

* Приведите пример таких чисел **a**, **b**, **c**, при которых программа неверно решает поставленную задачу.
* Укажите, какая часть программы является лишней.
* Укажите, как, по-вашему мнению, нужно доработать программу, чтобы не было случаев ее неправильной работы.

Задача 5 (для тренировки)

* Требовалось написать программу, которая определяет, имеется ли среди введенных с клавиатуры положительных целых чисел **a** и **b** хотя бы одно четное. Была написана следующая программа:

```
var a, b: integer;
begin
readln(a, b);
a := a mod 2;
if a > 0 then b := b mod 2;
if b > 0 then
    writeln ('четных чисел нет')
else writeln ('четное число есть');
end.
```

Известно, что программа написана с ошибками. Последовательно выполните три задания:

* приведите пример таких чисел **a**, **b**, при которых программа неверно решает поставленную задачу;
* укажите, как, по вашему мнению, нужно доработать программу, чтобы не было случаев ее неправильной работы;
* укажите, как можно доработать программу, чтобы она вместо вложенных операторов **IF** содержала логическую операцию **OR**.

Задача 6 (для тренировки)

* Требовалось написать программу, которая определяет, можно ли построить треугольник из отрезков с длинами **x, y, z**. Программа должна выводить соответствующее текстовое сообщение. Программист сделал в программе ошибки.

```
var x, y, z: real;
begin
readln (x, y, z);
if (x + y > z) then
begin
if (x + z > y) then
if (y + z > x) then
writeln('треугольник построить можно');
end
else writeln('треугольник построить нельзя');
end
```

Последовательно выполните задания:

* Приведите пример таких чисел **x, y, z**, при которых программа неверно решает поставленную задачу.
* Укажите, как нужно доработать программу, чтобы не было случаев ее неправильной работы (можно указать один из способов доработки исходной программы).
* Укажите, как можно доработать программу, чтобы она вместо вложенных операторов **IF** содержала логическую операцию **AND**.

Задача 7 (для тренировки)

Требовалось написать программу, которая решает неравенство $\frac{x-a}{bx} > 0$ относительно x для всех ненулевых действительных чисел $a \neq 0$ и $b \neq 0$, введенных с клавиатуры. Программист торопился и сделал в программе ошибки.

```
var a, b, x: real;
begin
readln(a, b, x);
if b > 0 then
write('x > ', a, ' или x < 0')
else
if a > 0 then
write('0 < x < ', a)
else
write(a, ' < x < 0');
end.
```

Последовательно выполните три задания:

* Приведите примеры таких чисел a, b, x, при которых программа неверно решает поставленную задачу.
* Укажите, какая часть программы является лишней?
* Укажите, как нужно доработать программу, чтобы не было случаев ее неправильной работы (можно указать любой способ доработки исходной программы).

Спасибо за внимание