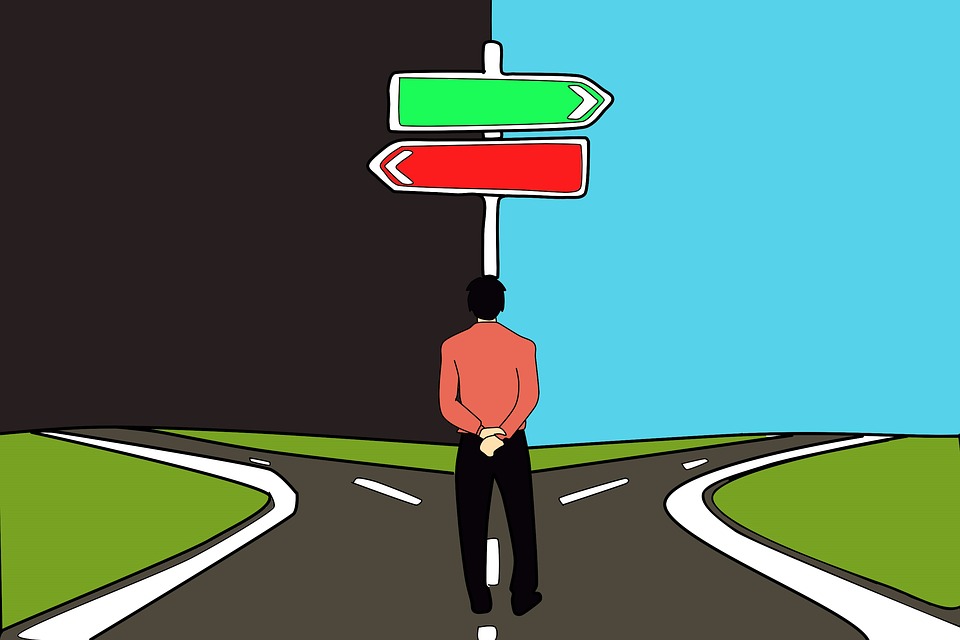
**Применение условного оператора If при решении задач на языке Pascal**

## Понятие разветвляющегося алгоритма

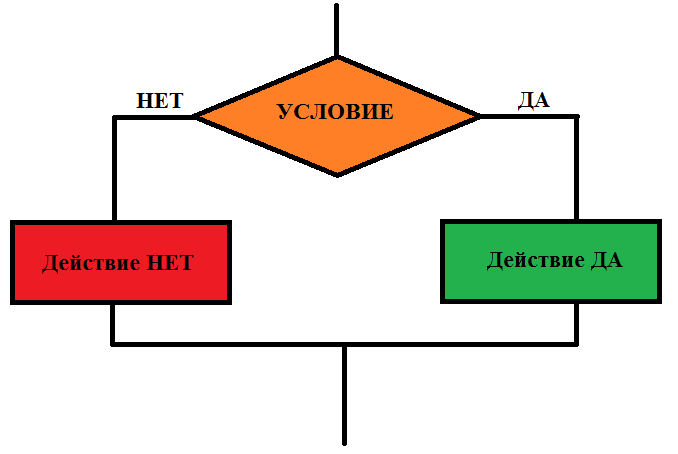
Если при решении задачи, надо выбрать один из вариантов, то необходимо использовать условный оператор (оператор условия).



Он позволяет выбрать только один вариантов, поэтому варианты должны быть самоисключающие. **Например**: число чётное или нечётное, число положительное или отрицательное, число кратное 3 или не кратное 3, буква гласная или согласная и т.д.

Необходимо правильно сформулировать и записать условие, результат которого может быть либо **ДА (ИСТИНА)**, либо **НЕТ (ЛОЖЬ).**

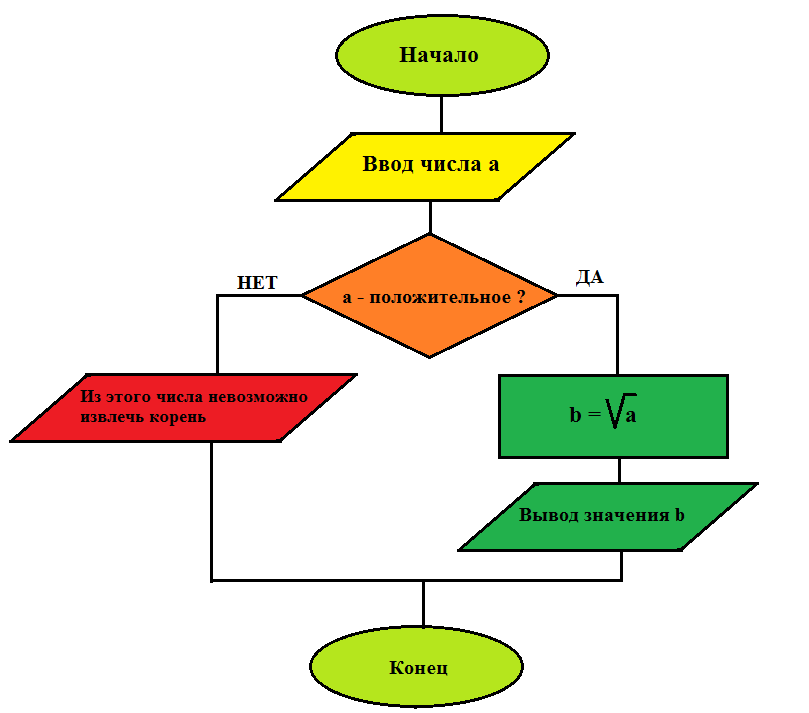
Графически (в виде блок-схемы) действие проверки условия можно изобразить так:



Такая форма называется **полной, т.к. есть Действие-ДА и Действие-НЕТ.**

|  |  |
| --- | --- |
| Возможна ситуация, что компонента **Действие-НЕТ нет**, тогда такая форма называется **сокращенная** (или редуцированная). | Но нельзя реализовать на языке программирования Pascal такой вариант условного алгоритма, к**огда компонент Действие-НЕТ присутствует, а компонент Действие-ДА отсутствует.** |

**Например:** Составить алгоритм для извлечения квадратного корня из введенного с клавиатуры числа



Алгоритм, в котором используется условный оператор, называют **разветвляющийся** или **условный**.

**Разветвляющийся** потому, что в таком алгоритме образуются ветви: ***ДА – «положительная», НЕТ – «отрицательная»***.

## Форма записи условного оператора If

В языке Pascal для реализации разветвляющего алгоритма используется оператор **If**

**Форма записи оператора If:**



**Перевод:**

**IF - Если**

**then – то**

**else - иначе**

**Словесное описание:**

Если условие выполняется, то выполнится **Действие-ДА**, иначе – выполнится **Действие-НЕТ**.

Такая форма называется **полной**. Обратите внимание, что «;» ставится только в самом конце.

**Форма сокращенной записи оператора If:**

****

**Правило**: компонент **then** обязательно должен быть, а **else** – может отсутствовать.

Пример 1: Составить программу, которая определяет, является ли число положительным.

|  |  |
| --- | --- |
| **Листинг программы** | **Комментарий** |
| Program chislo; | Название программы |
| Var x:real; | Описание переменной. Для данной задачи можно использовать переменную вещественного типа (если в условии явно не указано) |
| Begin |  |
| Writeln(‘Введите любое число’); | Запрос числа у пользователя |
| Readln(x); | Ввод числа |
| If (x>0) then | Проверка, является ли число положительным путем сравнения с нулем |
| Writeln (‘Введенное число положительное) | Действие-ДА |
| Else | Иначе |
| Writeln (‘Введенное число отрицательное или равное нулю’); | Действие-НЕТ |
| End. |  |

## Использование сложных (составных) условий

**Иногда возникает необходимость проверить несколько условий одновременно. Для этого можно использовать логические операторы (and, or).**

* Если используется **and,** то составное условие будет истинно, когда все простые условия истинны.
* Если используется **or,** то составное условие будет истинно, когда хотя бы одно простое условие будет истинно.

**Рассмотрите следующий пример.**

Пример 2: Составить программу, которая определяет, является ли число четным.

|  |  |
| --- | --- |
| **Листинг программы** | **Комментарий** |
| Program chislo; | Название программы |
| Var x:integer; | Описание переменной. Для данной задачи необходимо использовать переменную целочисленного типа |
| Begin |  |
| Writeln(‘Введите любое целое число’); | Запрос числа у пользователя |
| Readln(x); | Ввод числа |
| If (x mod 2=0) and (x<>0) then | Проверка числа на четность путем определения остатка от деления на 2. Проверку, что x не равно нулю делать обязательно, иначе в случае ввода нуля Pascal посчитает его четным числом |
| Writeln (‘Введенное число чётное’) | Действие-ДА |
| Else | Иначе |
| Writeln (‘Введенное число нечётное или равное нулю’); | Действие-НЕТ |
| End. |  |

Пример 3**:** Составить программу, которая генерирует случайное двухзначное целое число, оканчивающееся на ноль.

**Решение**:

Двухзначные числа – это числа от 10 до 99.

Для генерирования случайного числа используется функция **random**, в которой требуется указать: количество возможных значений и число, с которого начнется генерация (Всего чисел 99-10+1 = 90, начальное число = 10).

Запись выбора случайного числа будет выглядеть так:

**a:=random(90)+10;**

По условию задачи необходимо, чтобы число оканчивалось на ноль, поэтому после генерации числа надо проверить с помощью условного оператора, является ли последняя цифра числа нулем.

Если последняя цифра не равна нулю, то можно из генерированного числа отбросить количество единиц (вычесть).

**Например**, сгенерировалось число 50, оно устраивает условие задачи, поэтому дополнительную обработку делать не надо

**Например**, сгенерировалось число 67, оно не оканчивается на ноль, поэтому его надо преобразовать путем отбрасывания единиц, то есть 67-7=60

Для определения последней цифры двухзначного числа (как и любого другого целого числа) понадобится функция mod.

То есть запись **b:=a mod 10** позволяет в переменную b записать значение единиц числа a.

|  |  |
| --- | --- |
| **Листинг программы** | **Комментарий** |
| Program chislo; | Название программы |
| Var a:integer; | Описание переменной. Для данной задачи необходимо использовать переменную целочисленного типа |
| Begin |  |
| a:=random(90)+10; | Формула генерирования случайного числа |
| If (a mod 10=0) then | Проверка, является ли последняя цифра сгенерированного числа нулем |
| Writeln (‘Случайное число ’, a) | Если последняя цифра числа нуль, то выводим это число на экран. Перед else «;» не ставить. |
| Else | Иначе |
| Writeln (‘Случайное число ’, a-a mod 10); | Если последняя цифра числа не нуль, то выводим это число на экран, но вычитая последнюю цифру |
| End. |  |

*Попробуйте составить блок-схему для данной задачи.*

Задачи для самостоятельной работы**:**

1. Попробуйте решить программу (Пример 3) другим способом. (**Подсказка**. Сгенерировать цифру от 1 до 9 и потом умножить ее на 10. Тогда проверять, оканчивается ли число на ноль, не нужно).
2. Составить программу, которая генерирует случайное трехзначное целое число, оканчивающееся на ноль.
3. Составить программу, которая генерирует случайное двухзначное целое число. Определить, является ли число четным, а также какое количество в записи числа больше, единиц или десятков.
4. Составить программу, которая генерирует случайное число из диапазона от a до b. Причем значения a и b вводит пользователь с клавиатуры. Прежде чем генерировать число, необходимо проверить, что b>a.
5. Написать программу, которая определяет максимальное из трех введенных с клавиатуры чисел.
6. Написать программу, которая выводит пример на умножение двух однозначных чисел, запрашивает ответ пользователя, проверяет его и выводит соответствующее сообщение «Правильно!» или «Вы ошиблись!». Примеры должны составлять из случайно сгенерированных чисел.

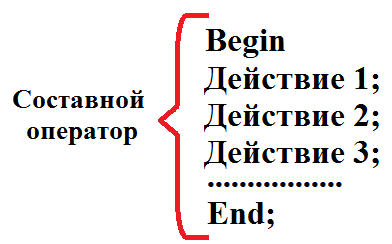
## Составной оператор

В решении задачи (Пример 3) мы немного схитрили и объединили обработку числа с выводом в одну команду. То что можно было написать в несколько строк, мы написали в одну. Разберемся, в чем отличие.

|  |  |
| --- | --- |
| **Как написано в программе?** | **Как можно было бы написать** |
| Writeln (‘Случайное число ’, a-a mod 10); | Begin  a:=a-mod 10;  Writeln (‘Случайное число ’, a);  End; |

Если в качестве Действие-Да или Действие-Нет надо записать несколько действий подряд, то их необходимо объединить (сгруппировать) с помощью дополнительных операторов **begin …. end;** Это называется **составной оператор**.

**Составной оператор** – это группа операторов**.** А пару операторов **begin …. end** называют **операторные скобки.**



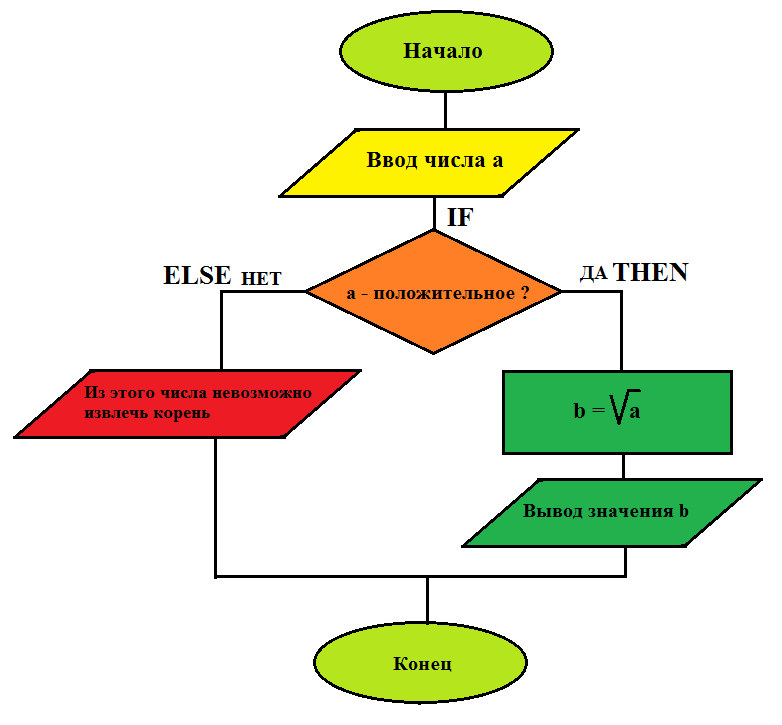
**Важно**: Составной оператор может быть и после then, и после else, т.е. в любой части оператора If.

Пример 4**:** Составить программу, которая извлекает квадратный корень из введенного с клавиатуры числа

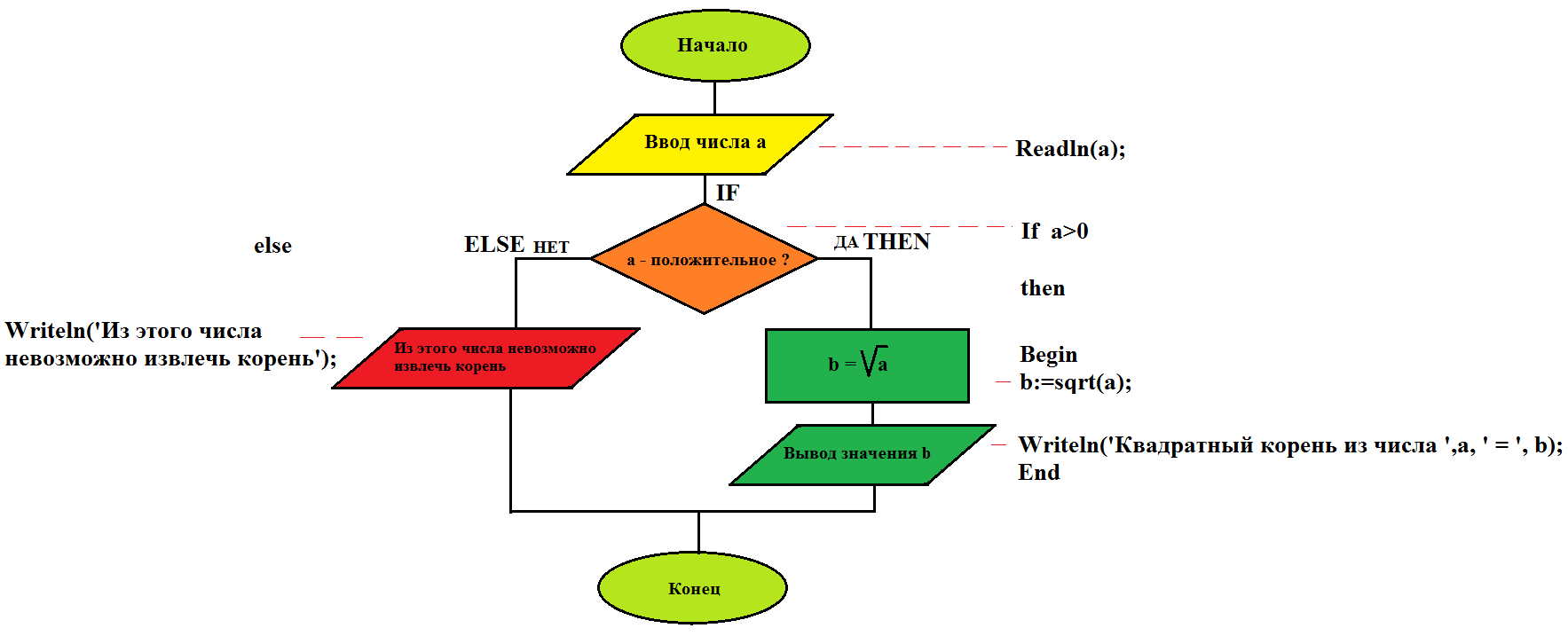
**Решение**: Прежде чем вычислять значение квадратного корня из числа, необходимого проверить, что число положительное (нулевые значения учитывать не будем).

Составим блок-схему. В «положительной» ветви у нас будет 2 действия (вычисление и вывод ответа), поэтому будем использовать составной оператор.

В блок-схеме подписаны компоненты оператора if: до условия пишем if, Да – then, Нет – else.



Далее в блок-схеме подпишем, какие операторы будем использовать при составлении программы.



|  |  |
| --- | --- |
| **Листинг программы** | **Комментарий** |
| Program kvadrat; | Название программы |
| Var a,b:real; | Описание переменных. При использовании функции вычисления квадратного корня результат будет вещественного типа. |
| Begin |  |
| Writeln(‘Введите любое число’); | Вывод текста-приглашения, чтобы пользователю было понятно, что необходимо ввести. |

|  |  |
| --- | --- |
| **Листинг программы** | **Комментарий** |
| Readln(a); | Ввод числа и запись его значения в переменную а |
| If a>0 | Начало условного оператора. Проверяем, что введенное число a положительное |
| Then | Начало «положительной» ветви |
| Begin | Начало составного оператора |
| b:=sqrt(a); | Вычисление квадратного корня |
| Writeln(‘Квадратный корень из числа ’,a, ‘ = ’, b); | Вывод результата на экран |
| End | Конец составного оператора. «;» перед оператором else не ставится |
| Else | Иначе |
| Writeln (‘Из этого числа невозможно извлечь корень’); | Вывод результата, в случае, если введенное число отрицательное или равное нулю |
| End. |  |

## Задачи для самостоятельной работы

1. Дракон каждый год отращивает по три головы, но после того, как ему исполнится 100 лет – только по две. Написать программу, которая определяет, сколько голов и глаз у дракона, которому N лет?
2. Написать программу для проверки трехзначного числа на четность. Найти сумму цифр числа, если число четное, или произведение его цифр, если число нечетное.
3. Сумма покупки составляет а рублей. Если а больше 1000 рублей, то предоставляется скидка 15%. Вывести на экран сумму покупки с учетом скидки либо сообщение о том, что скидка не предоставляется.
4. Дано двузначное число. Определить:

* входит ли в него цифра 3
* входит ли в него цифра а

1. Определить, является ли число а делителем числа b.

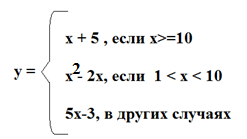
## Вложение условных операторов

Можно использовать несколько операторов If вложенных друг в друга. Количество вложений не ограничено, причем формы могут быть как полные, так и сокращенные.

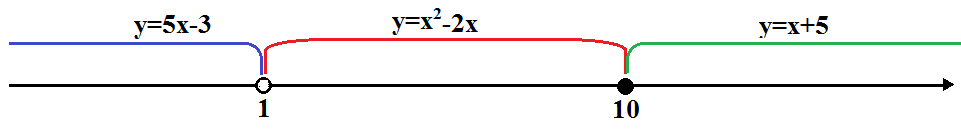
Варианты вложений могут быть абсолютно разные, в зависимости от задачи.

|  |  |
| --- | --- |
| **Фрагмент блок-схемы** | **Фрагмент программы** |
|  | **If условие\_1 then**  If условие\_2 then Действие-ДА  Else Действие-НЕТ\_1  **Else Действие-НЕТ\_2;** |
|  | **If условие\_1 then Действие-ДА\_1**  **Else**  If условие\_2 then Действие-ДА\_2  Else Действие-НЕТ\_2; |

Пример 5**:** С клавиатуры вводится число x. Вычислить значение y:

****

Рассмотрим все случаи на числовой оси, проверим, все ли значения учитываются условием задачи.

****

Составим фрагменты с использованием условного оператора:

**If x>=10 then y:=x+5**

**If (x>1) and (x<10) then y:=sqr(x)-2\*x**

Остальные значения попадают под вариант «в других случаях»

|  |  |
| --- | --- |
| **Листинг программы** | **Комментарий** |
| Program znach\_y; | Название программы |
| Var x,y:real; | Описание переменных. |
| Begin |  |
| Writeln(‘Введите значение x’); | Вывод текста-приглашения, чтобы пользователю было понятно, что необходимо ввести. |
| Readln(x); | Ввод числа и запись его значения в переменную x |
| **If x>=10 then y:=x+5** | Проверяем, подходит ли число под первое условие |
| **Else** | Иначе |
| **If (x>1) and (x<10)**  **then y:=sqr(x)-2\*x** | Проверяем, подходит ли число под второе условие |
| **Else** | Иначе |
| **y:=5\*x-3;** | Эта строка выполнится, если ни первое, ни второе условие не выполняется |
| Writeln (‘y=’,y); | Вывод результата на экран |
| End. |  |

Пример 6:

**Задача.** С клавиатуры вводится трехзначное целое положительное число. Определить, сколько цифр в записи числа четных, сколько нечетных. Организовать проверку, что введено именно трехзначное число.

**Решение:**

Для того чтобы определить четная или нечетная каждая цифра в записи числа необходимо сначала число «разбить» на цифры. Для этого используются операторы целочисленного деления div и mod.

Div определяет целую часть от деления, mod определяет остаток от деления.

**Например:**

x:=124;

a:=x mod 10; *в переменную a будет записано число единиц числа x (т.е. 4)*

b:=(x div 10) mod 10; *в переменную b будет записано число десятков числа x (т.е. 2)*

с:=x div 100; *в переменную c будет записано число сотен числа x (т.е. 1)*

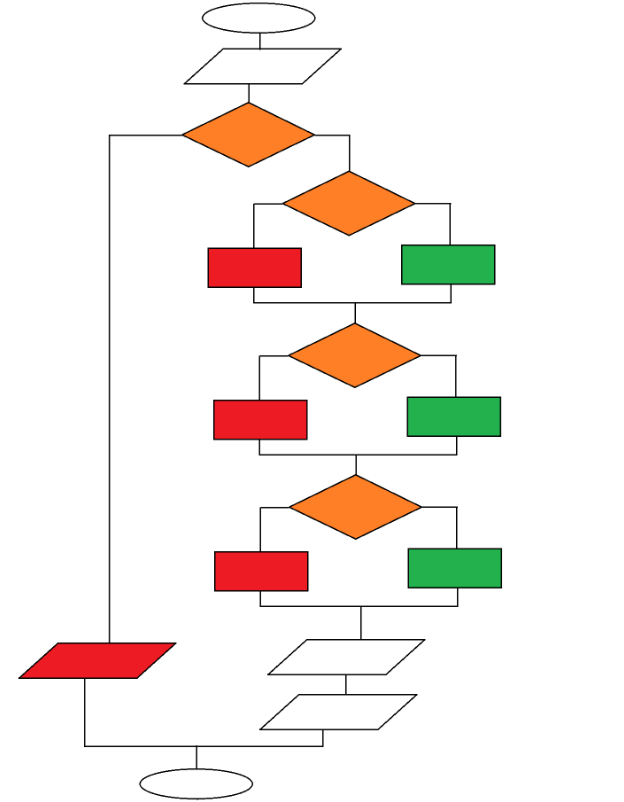
В данном решении будем использовать следующие переменные:

x – число, которое вводится с клавиатуры

k – количество четных цифр

m – количество нечетных цифр

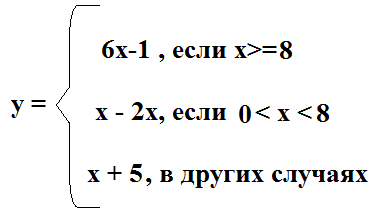
Для хранения цифр отдельных переменных использовать не будем. Ниже представлена обобщенная блок-схема.

**

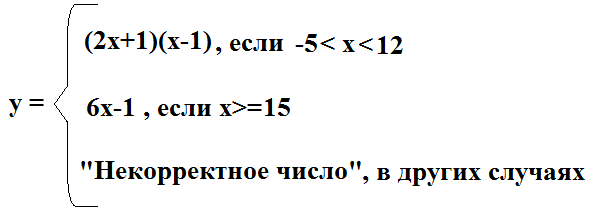
|  |  |
| --- | --- |
| **Листинг программы** | **Комментарий** |
| Program chislo; | Название программы |
| Var x,k,m:integer; | Описание переменных. В данной задаче надо использовать переменные целочисленного типа |
| Begin |  |
| k:=0; m:=0; | Инициализация переменных k,m. Задаем начальные значения равные нулю. |
| Writeln(‘Введите значение x’); | Вывод текста-приглашения, чтобы пользователю было понятно, что необходимо ввести. |
| Readln(x); | Ввод числа и запись его значения в переменную x |
| **If (x>99) and (x<1000) then** | Проверяем, что число трехзначное |
| **Begin** |  |
| **If ((x mod 10) mod 2 = 0) and (x mod 10 <> 0)** | Проверяем цифру, обозначающую единицы. Является ли оно чётным. |
| **then k:=k+1** | Если цифра чётная, то увеличиваем значение переменной k + 1 |
| **else m:=m+1;** | Иначе увеличиваем значение переменной m + 1 |
| **If (((x div 10) mod 10) mod 2 = 0) and (((x div 10) mod 10) <> 0)** | Проверяем цифру, обозначающую десятки. Является ли оно чётным. |
| **then k:=k+1** | Если цифра чётная, то увеличиваем значение переменной k + 1 |
| **else m:=m+1;** | Иначе увеличиваем значение переменной m + 1 |
| **If ((x div 100) mod 2 = 0) and (x div 100 <> 0)** | Проверяем цифру, обозначающую сотни. Является ли оно чётным. |
| **then k:=k+1** | Если цифра чётная, то увеличиваем значение переменной k + 1 |
| **else m:=m+1;** | Иначе увеличиваем значение переменной m + 1 |
| **Writeln(‘Было введено число ’,x);** | Вывод на экран исходного числа |
| **Writeln(‘В записи этого числа ‘, k, ’ чётных цифр’);** | Вывод на экран количества чётных цифр |
| **Writeln(‘В записи этого числа ‘, m, ’ нечётных цифр’);** | Вывод на экран количества нечётных цифр |
| **End** | Перед else «;» не ставится |
| **Else** |  |
| **Writeln(‘Вы ввели не трехзначное число’);** | Если было введено не трехзначное число, то выполнится только вывод сообщения «Вы ввели не трехзначное число» |
| End. |  |

## Задачи для самостоятельной работы

1. С клавиатуры вводится число x. Вычислить значение y:



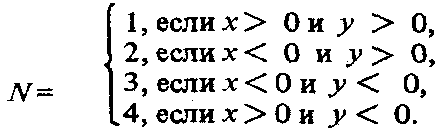
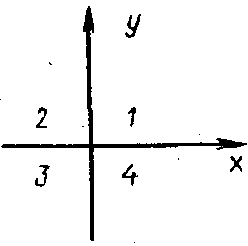
1. С клавиатуры вводится число x. Вычислить значение y:



1. С клавиатуры вводится число x. Вычислить значение y:

1. С клавиатуры вводится четырехзначное целое положительное число. Определить, сколько цифр в записи числа четных, сколько нечетных, сколько равных нулю. Организовать проверку, что введено именно четырехзначное число.
2. Написать программу определения стоимости разговора по телефону с учетом скидки – 20%, предоставляемой по субботам и воскресеньям. *(Исходные данные: стоимость минуты разговора; день недели, когда разговор произошел; время разговора)*
3. Составить программу для определения, к какой четверти (квадранту) принадлежит точка T с координатами x и y (x≠0 и y≠0, организовать проверку на ввод данных). Результатом решения является номер квадранта.



1. Даны две точки, известны их координаты. Написать программу, которая проверяет, лежат ли точки в одном квадранте или в разных *(и в том, и в другом случае сообщить номер квадранта)*
2. Написать программу, решающую следующую задачу: дано целое число. Определить, четное число или нет, и попадает ли в диапазон от 10 до 100. Выдать соответствующее сообщение.
3. Написать программу решения квадратного уравнения вида ax2+bx+c=0*. (Программа должна проверять исходных данных: если коэффициент при неизвестном во второй степени равен 0, выдать соответствующее сообщение – «Это не квадратное уравнение»)*
4. Написать программу, которая определяет оптимальный вес пользователя, сравнивает его с реальным и выдает рекомендации о необходимости поправиться или похудеть. *(Оптимальный вес вычисляется по формуле: Рост (в см) - 100)*
5. Имеется билет с шестизначным номером. Определить, является ли билет счастливым.