**Условная инструкция**

Есть такое высказывание, что самая главная в программе – операция присвоение, т.к. она меняет данные в программе. А все остальные управляющие конструкции нужны, чтобы в правильном порядке выполнять операции присвоения.

**Ветвление** – это управляющая конструкция, которая позволяет выполнить то или другое действие в зависимости от выполнения условия.

В общем виде условный оператор

**if** условие:

действия 1

**else**:

действия 2

Не забывайте про : в конце условия и после else.

В команде if при выполнении условия можно выполнять более одной команды. Для этого их все необходимо выделить отступом. Такая запись называется блоком кода. По отступам интерпретатор Питона определяет, какие команды исполнять при выполнении каких условий. Аналогично можно делать и для команды else.

x = 2

y = 0

z = 0

**if** x > 0:

y += 1

z += 2

**else**:

y -= 1

z -= 2

print(y, z)

Перед последней строчкой нет отступа — это означает, что она будет выполнена в конце работы программы в любом случае.

Действия, которые должны выполняться при выполнении или невыполнении условия должны быть написаны с отступом (2 или 4 пробела). Если отступа нет, то это означает, что инструкция закончилась. Если в блоке несколько инструкций, то они должны быть написаны с одинаковым отступом. Обычно для обозначения блоков используются специальные слова или символы (begin end, {}). В Python их нет. Здесь все определяется отступами. Для отступов лучше использовать пробелы, а не табуляции. Среды программирования часто преобразуют табуляции в пробелы. Нельзя в одном блоке смешивать символы табуляции и пробела.

Чтобы сделать отступ нескольким строчкам, выделите их и нажмите Tab.

Чтобы уменьшить отступ у нескольких строчек, выделите их и нажмите Shift+Tab.

**Неполная условная инструкция**

Части else может не быть

Найти модуль числа

1 вариант

x=int(input())

**if** x<0:

x=-x

**else**:

x=x

print(x)

2 вариант

x=int(input())

**if** x<0:

x=-x

print(x)

Условия возвращают логический тип и могут быть True или False

**Условия могут содержать**

1. Операции сравнения

> < >= <= == !=

2. Логические операции

И **and** оба условия истинны x>0 **and** y>0

ИЛИ **or** хотя бы одно условие истинно x!=0 **or** y!=0

НЕ **not** **not**(x>0)

Если можно найти значение логического выражения, вычислив только первое условие, то вторая часть вычисляться не будет.

**Приоритет операций** not, and, or

**Задача 1.** Проверить является ли число нечетным. Вывести YES или NO.

n = int(input())

**if** n % 2 !=0:

print("YES")

**else:**

print("NO")

**Задача 2.** Найти максимум из трех чисел.

a = int(input())

b = int(input())

c = int(input())

mmax = a

**if** b > mmax:

mmax = a

**if** c > mmax:

mmax = c

print(mmax)

**Задача 3**. Дано 3 числа. Найти сколько среди них совпадает.

Ответом будет 3, 2 или 0

a=int(input())

b=int(input())

c=int(input())

**if** a==b **and** b==c: #можно **if** a==b==c:

print(3)

**elif** a==b **or** b==c **or** a==c:

print(2)

**else**:

print(0)

А вот такой вариант будет неправильным

**if** a==b==c:

print(3)

**elif** (a != b != c:

print(0)

**else**:

print(2)

При a=2 b=4 c=2 выдаст 0

Второе условие должно выглядеть так

**elif** (a != b != c !=a: