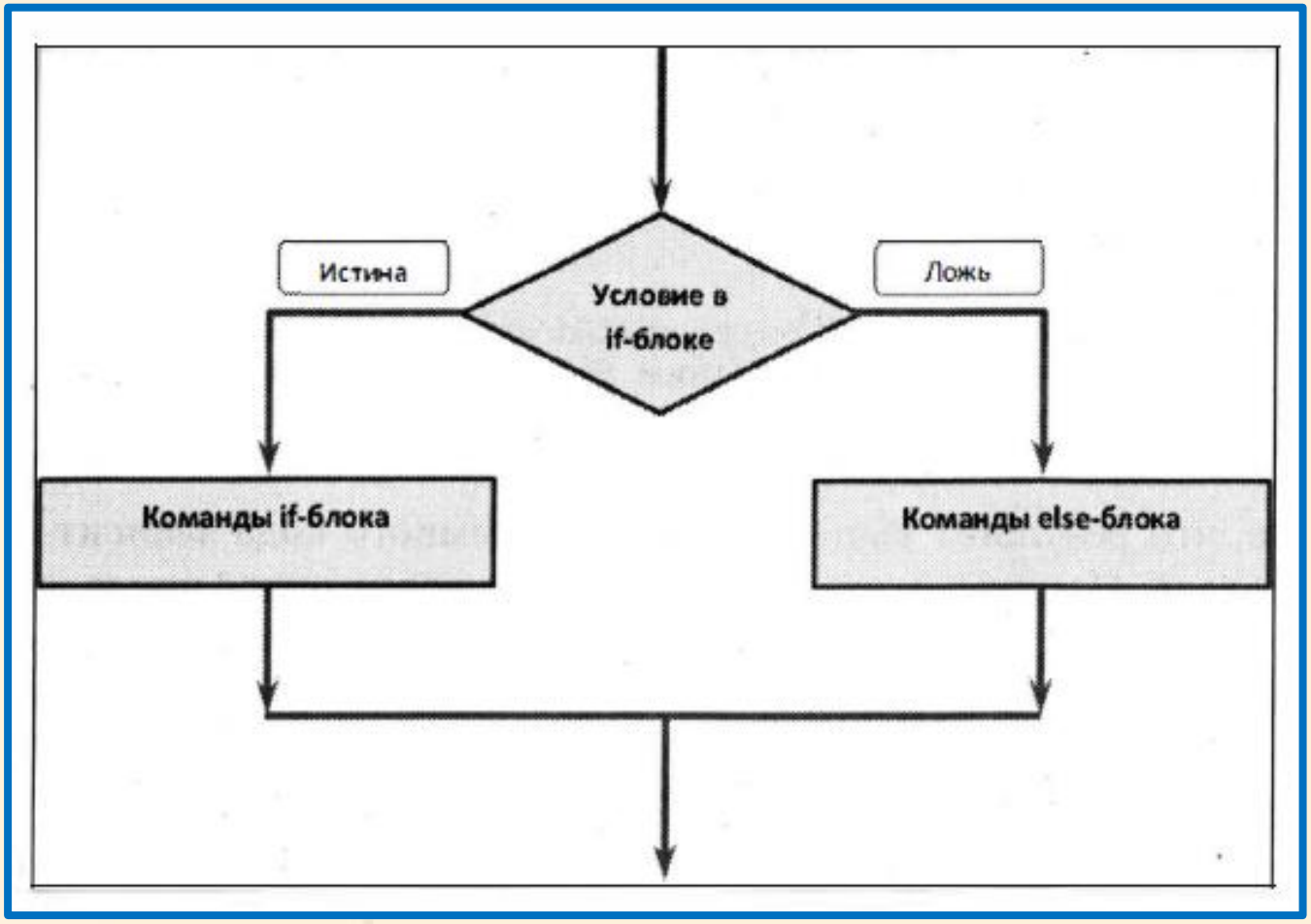
Условный оператор на PYTHON

Урок 5

Актау, 2018

Схема выполнения условного оператора



Оператор ветвления *if* позволяет выполнить определенный набор инструкций в зависимости от некоторого условия. Возможны следующие варианты использования.

Итак, условная инструкция в Питоне имеет следующий синтаксис:

if Условие:

    Блок инструкций 1

else:

    Блок инструкций 2

Блок инструкций 1 будет выполнен, если Условие истинно. Если Условие ложно, будет выполнен Блок инструкций 2.

В условной инструкции может отсутствовать слово else и последующий блок. Такая инструкция называется неполным ветвлением. Например, если дано число x и мы хотим заменить его на абсолютную величину x, то это можно сделать следующим образом:

```
x = int(input())
if x < 0:
    x = -x
print(x)
```

Операторы сравнения

Как правило, в качестве проверяемого условия используется результат вычисления одного из следующих операторов сравнения:

`<`

Меньше — условие верно, если первый операнд меньше второго.

`>`

Больше — условие верно, если первый операнд больше второго.

`<=`

Меньше или равно.

`>=`

Больше или равно.

`==`

Равенство. Условие верно, если два операнда равны.

`!=`

Неравенство. Условие верно, если два операнда неравны.

Например, условие `(x * x < 1000)` означает "значение `x * x` меньше 1000", а условие `(2 * x != y)` означает "удвоенное значение переменной `x` не равно значению переменной `y`".

Операторы сравнения в Питоне можно объединять в цепочки (в отличии от большинства других языков программирования, где для этого нужно использовать логические связки), например, `a == b == c` или `1 <= x <= 10`.

Условный оператор - примеры

python

```
if 1:
   print("hello 1")
```

Напечатает: *hello 1*

```
a = 3
if a == 3:
    print("hello 2")
```

Напечатает: *hello 2*

```
a = 3
if a > 1:
   print("hello 3")
```

Напечатает: *hello 3*

```
lst = [1, 2, 3]
if lst :
   print("hello 4")
```

Напечатает: *hello 4*

Условный оператор - примеры

```
a = 3
if a > 2:
    print("H")
else:
    print("L")
```

```
a = 1
if a > 2:
    print("H")
else:
    print("L")
```

Условие такого вида можно записать в строчку, в таком случае оно будет представлять собой **тернарное выражение**.

```
a = 17
b = True if a > 10 else False
print(b)
```

В результате выполнения такого кода будет напечатано: *True*

Для реализации выбора из нескольких альтернатив можно использовать конструкцию ***if – elif – else*.**

```
if выражение_1:
    инструкции_(блок_1)
elif выражение_2:
    инструкции_(блок_2)
elif выражение_3:
    инструкции_(блок_3)
else:
    инструкции_(блок_4)
```

```
a = int(input("введите число:"))
if a < 0:
   print("Neg")
elif a == 0:
   print("Zero")
else:
   print("Pos")
```

Проверим, что хотя бы одно из чисел a или b оканчивается на 0:

```
a = int(input())
b = int(input())
if a % 10 == 0 or b % 10 == 0:
    print('YES')
else:
    print('NO')
```

Пример программы, определяющий четверть координатной плоскости

```
x = int(input())
y = int(input())
if x > 0 and y > 0:
    print("Первая четверть")
elif x > 0 and y < 0:
    print("Четвертая четверть")
elif y > 0:
    print("Вторая четверть")
else:
    print("Третья четверть")
```


```
# Пользователь вводит значение
res=eval(input("Введите что-нибудь: "))
# Используем условный оператор для проверки
# типа введенного пользователем значения
if type(res)==int:
    # Если целое число
    print("Вы ввели целое число!")
else:
    # Если что-то другое
    print("Это точно не целое число!")
# После выполнения условного оператора
print("Работа завершена!")
```

Результат выполнения программы

```
Введите что-нибудь: 12
Вы ввели целое число!
Работа завершена!
```

Если пользователь вводит не целое
дет несколько иным (жирный шри

```
Введите что-нибудь: 1+2**3-4
Вы ввели целое число!
Работа завершена!
```

Результат выполнения программы

```
Введите что-нибудь: 12.0
Это точно не целое число!
Работа завершена!
```

python

```
# Пользователь вводит значение
res=eval(input("Введите что-нибудь: "))
# Тип значения запоминаем в переменной
resType=type(res)
# Используем условные операторы (упрощенная форма)
# для проверки типа введенного пользователем значения
if resType==int:
    # Если целое число
    print("Это целое число!")
if resType==float:
    # Если действительное число
    print("Это действительное число!")
if resType!=int and resType!=float:
    # Если не число
    print("Наверное, это текст!")
# После выполнения условных операторов
print("Работа завершена!")
```

```
Введите что-нибудь: 12
Это целое число!
Работа завершена!
```

```
Введите что-нибудь: "Изучаем Python"
Наверное, это текст!
Работа завершена!
```

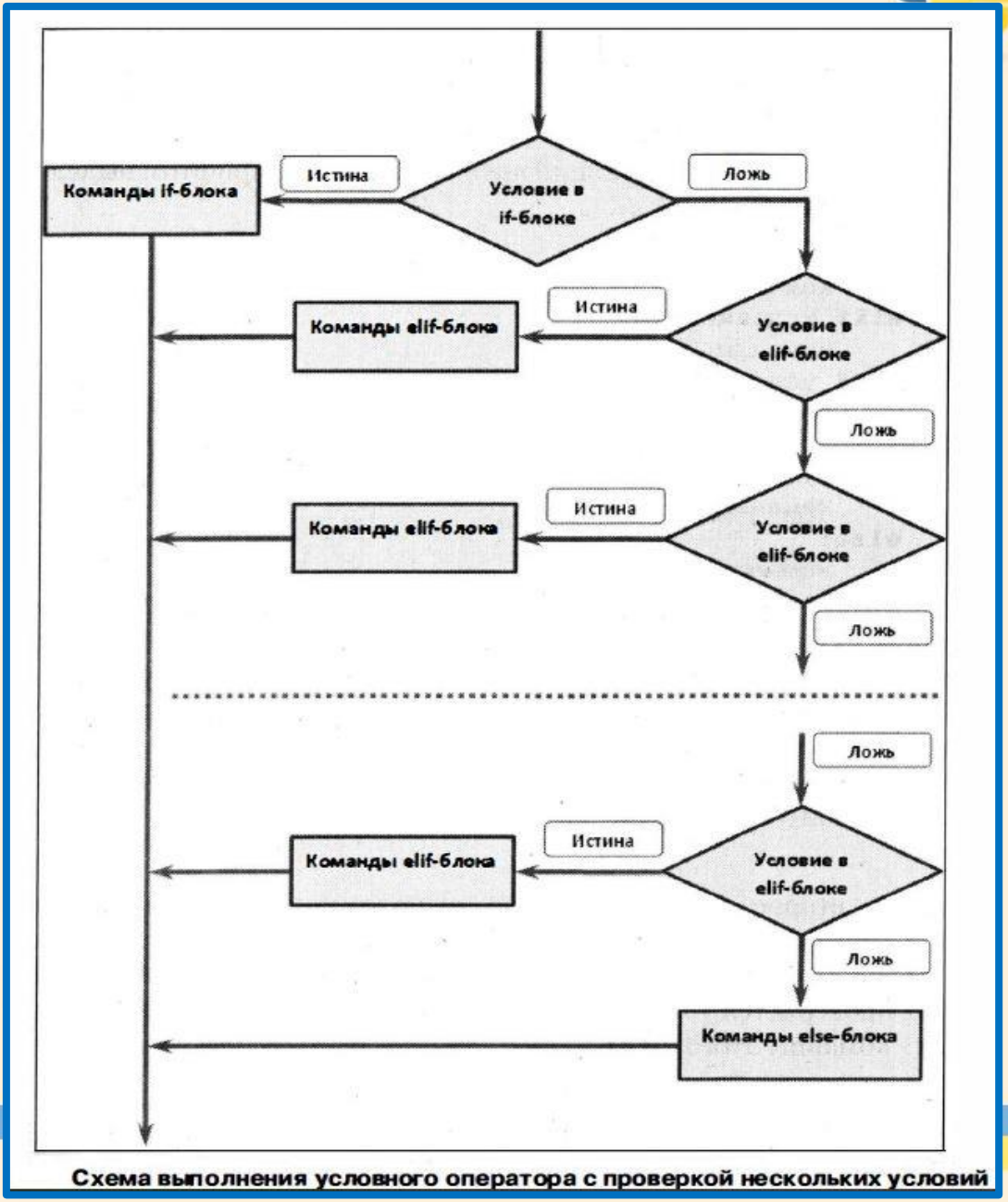


Схема выполнения условного оператора с проверкой нескольких условий


```
# Пользователь вводит значение
res=eval(input("Введите что-нибудь: "))
# Тип значения запоминаем в переменной
resType=type(res)
# Используем условные операторы (упрощенная форма)
# для проверки типа введенного пользователем значения
if resType==int:
    # Если целое число
    print("Это целое число!")
elif resType==float:
    # Если действительное число
    print("Это действительное число!")
else:
    # Если не число
    print("Наверное, это текст!")
# После выполнения условных операторов
print("Работа завершена!")
```