Конспект по теме «Первая программа»

Логический тип данных

Простой алгоритм в коде выполняется линейно сверху вниз. Но иногда могут появиться дополнительные условия, из-за которых требуется игнорировать часть кода.
Например, у нас есть алгоритм расчета цены. Но при условии наличия у покупателя пенсионного удостоверения, мы рассчитываем цену со скидкой. Т.е. если условие “покупатель имеет пенсионное удостоверение” — верно, то мы включаем расчет со скидкой, в противном случае — игнорируем эту часть расчетов.

Значения “верно/неверно” содержат тип данных `bool`. Переменные этого типа могут принимать только 2 специальных значения: `True` (истина) и `False` (ложь).
Результаты сравнений являются логическим типом данных:

```
print(3 > 4) #False
print(5 < 9) #True
print(7 >= 7) #True
print(-9 > 0) #False
print(10 == 10) #True
print('hello' != 'world') #True
```

`==` - равно, `!=` - не равно.

Предположим, что мы вводим с клавиатуры число и записываем его в переменную `x`. Мы хотим понять, принадлежит ли значение `x` промежутку от 5 до 10. Можно записать два сравнения и объединить их оператором `and`.

```
x = int(input())
print(x >= 5 and x <= 10)
```

Основные операции с логическим типом данных:

and: Логическое И

x	y	x and y
False	False	False
False	True	False
True	False	False
True	True	True

or: Логическое ИЛИ

x	y	x or y
False	False	False
False	True	True
True	False	True
True	True	True

not: Отрицание

x	not x
False	True
True	False

Условные конструкции

Базовыми конструкциями в алгоритмах являются условные конструкции. Условные конструкции позволяют регулировать направление хода выполнения алгоритма в зависимости от выполнения условий. Как на этой схеме:

Рассмотрим условные конструкции на примере.

С клавиатуры вводится число. Нужно определить его знак.

```
1 x = int(input())
2 if x > 0:
    print('Положительное число')
```

Если `x > 0`, то распечатается строка “Положительное число”. Если `x <= 0`, то не распечатается ничего.

Улучшим программу:

```
1 x = int(input())
2 if x > 0:
3    print('Положительное число')
4 else:
    print('Отрицательное число')
```

Если `x > 0`, то распечатается строка “Положительное число”. Иначе - строка “Отрицательное число”.

Давайте обработаем отдельно случай, когда `x == 0`:

```
1 x = int(input())
2 if x > 0:
3    print('Положительное число')
4 elif x < 0:
5    print('Отрицательное число')
6 else:
    print('Ноль')
```

Если `x > 0`, то печатаем “Положительное число”, иначе, если `x < 0`, то печатаем “Отрицательное число”, иначе печатаем “Ноль”.

Обратите внимание на отступы внутри блока. Все, что выделено отступом, относится к этому блоку. Обычно отступ занимает 4 пробела.

Цикл while

Цикл - это способ организовать многократное выполнение кода.

Например, мы хотим распечатать все числа от 1 до 10. С помощью цикла `while` это реализуется следующим образом:

```
1 i = 1
2 while i <= 10:
3     print(i)
      i += 1
```

В данном случае переменная `i` - это счетчик цикла. В начале `i = 0`. На каждой итерации цикла мы распечатываем текущее значение `i` и увеличиваем `i` на единицу. Повторяем эти действия пока выполняется условие: `i <= 10.`
`i += 1` то же самое, что и `i = i + 1.`