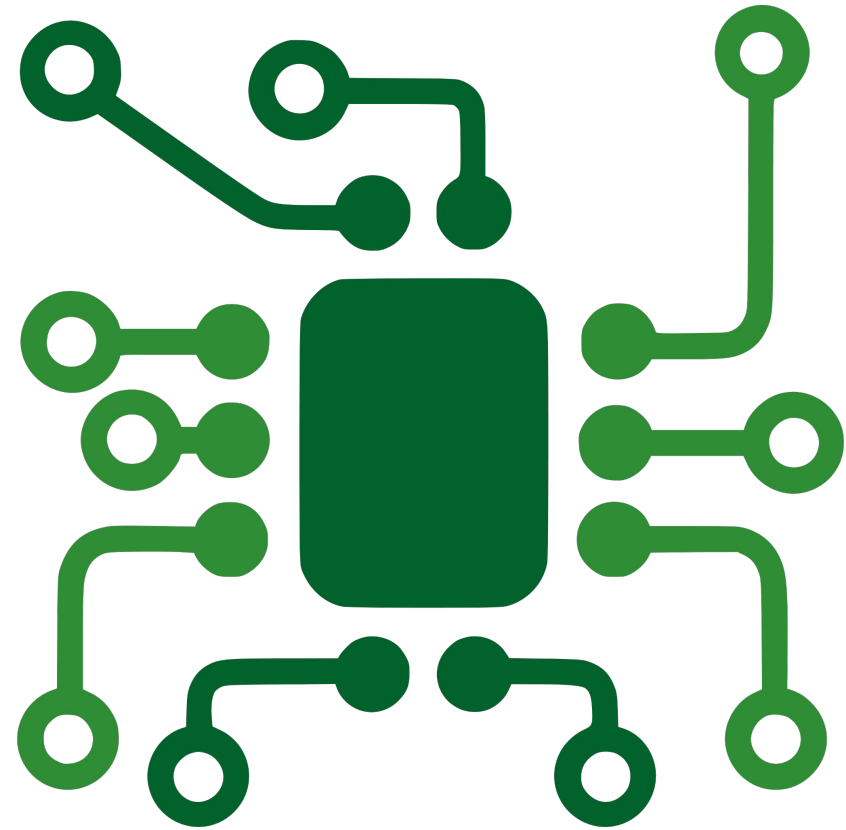


Моя Школа

сертифицированные IT курсы

Операторы ветвления



Задача №1:

У Анны 2 яблока, у Пола 5 яблок. Выведите на экран сколько яблок у Пола и сколько яблок у Анны одной командой `print`. Использовать переменные

```
37  anna_apples = 2
38  paul_apples = 5
39  print('Anna has', anna_apples, 'Paul has', paul_apples)
40  |
```

Задача №2: Дана длина ребра куба. Найти объем куба и площадь его боковой поверхности

```
2  rib = int(input())  
3  v = rib ** 3  
4  s = 4 * rib ** 2  
5
```

Задача №3: Улитка в день проползает 2м, но за ночь сползает на 1 м. За сколько дней улитка доползает, если дерево в высоту 20м

План занятия



1

Операторы
сравнения

2

Операторы
ветвления
(условия)

3

Тип данных
bool



4

Операторы
or, and и not

5

Предикаты
в Python

Оператор	Описание	Примеры
==	Проверяет равны ли оба операнда. Если да, то условие становится истинным.	5 == 5 в результате будет True True == False в результате будет False "hello" == "hello" в результате будет True
!=	Проверяет равны ли оба операнда. Если нет, то условие становится истинным.	12 != 5 в результате будет True False != False в результате будет False "hi" != "Hi" в результате будет True
<>	Проверяет равны ли оба операнда. Если нет, то условие становится истинным.	12 <> 5 в результате будет True. Похоже на оператор !=
>	Проверяет больше ли значение левого операнда, чем значение правого. Если да, то условие становится истинным.	5 > 2 в результате будет True. True > False в результате будет True. "A" > "B" в результате будет False.
<	Проверяет меньше ли значение левого операнда, чем значение правого. Если да, то условие становится истинным.	3 < 5 в результате будет True. True < False в результате будет False. "A" < "B" в результате будет True.
>=	Проверяет больше или равно значение левого операнда, чем значение правого. Если да, то условие становится истинным.	1 >= 1 в результате будет True. 23 >= 3.2 в результате будет True. "C" >= "D" в результате будет False.
<=	Проверяет меньше или равно значение левого операнда, чем значение правого. Если да, то условие становится истинным.	4 <= 5 в результате будет True. 0 <= 0.0 в результате будет True. -0.001 <= -36 в результате будет False.

Условный оператор ветвления if

Оператор ветвления if позволяет выполнить определенный набор команд в зависимости от некоторого условия

Синтаксис оператора if выглядит так.

```
if выражение:  
    команда_1  
    команда_2  
    ...  
    команда_n
```

После оператора if записывается выражение. Если это выражение истинно, то выполняются команды, определяемые данным оператором. Выражение является истинным, если его результатом является число не равное нулю, непустой объект, либо логическое True. После выражения нужно поставить двоеточие :

ВАЖНО: блок кода, который необходимо выполнить, в случае истинности выражения, отделяется четырьмя пробелами слева!

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

if True:
    print("Hello!")
```

Конструкция if – else

Бывают случаи, когда необходимо предусмотреть альтернативный вариант выполнения программы. Т.е. при истинном условии нужно выполнить один набор команд, при ложном – другой. Для этого используется конструкция if – else.

if выражение:

команда_1

команда_2

...

команда_n

else:

команда_a


команда_b





...

команда_x

```
1 a = 2
2 if a < 1:
3     print("a < 1 ")
4 else:
5     print("a > 1")
```

else

Run:  2 x

  C:\Users\User\Desktop
  a > 1

Задание №1

Необходимо написать программу, которая требует у пользователя ввести целое число и проверяет чётное оно или нет

```
# Необходимо написать программу, которая требует у пользователя
# ввести целое число и проверяет чётное оно или нет

# пользователь вводит число
a = input()

# преобразовываем его к целочисленному типу
a = int(a)

# оператор % возвращает остаток от деления,
# чётные числа нацело делятся 2,
# т.е остаток от деления равен 0.
# Нечётные числа имеют остаток, равный 1.

if a % 2 == 0:
    print("Чётное")
else:
    print("Нечётное")
```

Конструкция if – elif – else

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

Проще говоря, она выбирает, какое действие следует выполнить, в зависимости от значения переменных в момент проверки условия.

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

```
a = int(input('Введите число:'))  
if a < 0:  
    print('Число меньше 0')  
elif a == 0:  
    print('Число равно 0')  
else:  
    print('Число больше 0')
```


Задание №2

Пользователь вводит порядковый номер пальца руки, надо вывести его название

```
1 a = int(input("Введите номер пальца: "))
2
3 if a == 1:
4     print("Большой палец")
5 elif a == 2:
6     print('Указательный палец')
7 elif a == 3:
8     print('Средний палец')
9 elif a == 4:
10    print('Безымянный палец')
11 elif a == 5:
12    print('Мизинец')
13 else:
14    print('Нет соответствий')
15
```

Задание №3

Пользователь вводит номер месяца (от 1 до 12). Вывести название сезона года на экран (зима, весна, лето, осень)

```
64 month = int(input())
65
66 if 3 >= month >= 5:
67     print('весна')
68 elif 6 >= month >= 8:
69     print('лето')
70 elif 9 >= month >= 11:
71     print('осень')
72 else:
73     print('зима')
```

Оператор	Описание	Примеры
and	Логический оператор "И". Условие будет истинным если оба операнда истина.	True and True равно True. True and False равно False. False and True равно False. False and False равно False.
or	Логический оператор "ИЛИ". Если хотя бы один из операндов истинный, то и все выражение будет истинным.	True or True равно True. True or False равно True. False or True равно True. False or False равно False.
not	Логический оператор "НЕ". Изменяет логическое значение операнда на противоположное.	not True равно False. not False равно True.

```
1 a = int(input())
2 b = int(input())
3 c = bool(input())
4
5 if a > 0 and b < 0:
6     print("and")
7 elif a > 0 or b < 0:
8     print("or")
9
10 if not c == False:
11     print("not")
12
13
```

Run: 6 x

C:\Users\User\Desktop\overone\

3
-2
True
and
not

Задание №4

Вводятся три целых числа. Определить какое из них наибольшее.

```
# Вводятся три целых числа. Определить какое из них наибольшее.

a = int(input())
b = int(input())
c = int(input())

if a > b and a > c:
    print('Наибольшее: ', a)
else:
    if b > c:
        print('Наибольшее:', b)
    else:
        print('Наибольшее:', c)
```


Задание №5

Напишите игру камень, ножницы, бумага
В качестве хода компьютера используйте
числа от 1 до 3 соответствующие
Решить через модуль random

```
1 import random
2
3 k = random.randint(1, 3)
4 k = str(k)
5 r = input("Введите: камень(1), ножницы(2) или бумага(3): ")
6
7 if k == r:
8     print("Ничья")
9 elif k == "1" and r == '2':
10     print('Вы проиграли(')
11 elif k == "1" and r == '3':
12     print('Вы выиграли')
13 elif k == "2" and r == '3':
14     print('Вы проиграли(')
15 elif k == "2" and r == '1':
16     print('Вы выиграли')
17 elif k == "3" and r == '2':
18     print('Вы выиграли')
19 elif k == "3" and r == '1':
20     print('Вы проиграли(')
21 else:
22     print("Некорректный ввод")
23
```

Логический тип данных (bool) (или булевый тип) это примитивный тип данных, который принимает 2 значения – истина или ложь.

В Python имеется самостоятельный логический тип bool, с двумя предопределенными значениями:

True - истина

False - ложь

Важно! True и False пишутся с большой буквы. Если написать с маленькой true, интерпретатор выдаст ошибку: `NameError: name 'true' is not defined`

Задание №6

Напишите программу, которая выполняет сравнение двух переменных , если условие верно то выполняется вложенный блок части if. Обязательное условие: if $a > b$ писать нельзя. Только использование булевого значения выражения.

```
1  # Напишите программу, которая выполняет сравнение
2  # двух переменных , если условие верно то выполняется
3  # вложенный блок части if.
4  |
5  x = int(input())
6  y = int(input())
7
8  two_values = (x < y)
9
10 if two_values:
11     print(x, "меньше", y)
```

Задание №7

Существует ли треугольник с заданными сторонами?

Примечание: Треугольник существует при условии, что сумма 2-ух его сторон больше третьей

```
print("Длины сторон:")
a = float(input("a = "))
b = float(input("b = "))
c = float(input("c = "))

# Треугольник существует только тогда, когда сумма любых двух его сторон
# больше третьей стороны. В заголовке if сразу проверяются три стороны.
# Каждая из них должна быть меньше суммы других. Поэтому используется
# логический оператор AND.

if a+b>c and a+c>b and b+c>a:
    print("Треугольник есть")

# Если хотя бы одна сторона оказывается больше суммы
# других, то треугольник из заданных отрезков построить нельзя.
|
else:
    print("Треугольника нет")

# Примечание. Существует понятие вырожденного треугольника.
# В этом случае третья сторона может равняться сумме двух других.
# Тогда в условии вместо знака ">" # следует использовать ">=".
```

Задание №8

Написать примитивный калькулятор. Пользователь должен ввести число, потом операцию (+-/*) и потом ещё одно число, после этого пользователь получает ответ. Числа могут быть дробными


```
4 a = int(input("Введите первое число: "))
5 b = int(input("Введите второе число: "))
6 s = input("Введите операцию которую вы хотите сделать: ")
7
8 if s == "+":
9     print(a+b)
10 elif s == "-":
11     print(a-b)
12 elif s == "*":
13     print(a*b)
14 elif s == "/":
15     print(a/b)
16 elif s == "%":
17     print(a%b)
18 elif s == "^":
19     print(a**b)
20
21 elif s == "=":
```

Run: 5 x

C:\Users\User\Desktop\overone\venv\Scripts\python.exe "C:/Users/User/Desktop/overone/main.py"

Введите первое число: 6

Введите второе число: 2

Введите операцию которую вы хотите сделать: ^

36

Функция `is_infant()` — это функция-предикат (или функция-вопрос). Предикат отвечает на утвердительный вопрос «да» или «нет», возвращая значение типа `bool`. Предикаты во всех языках принято именовать особым образом для простоты анализа. В Python предикаты, как правило, начинаются с префикса `is` или `has`:

`is_infant()` — «младенец ли?»

`has_children()` — «есть ли дети?»

`is_empty()` — «пустой ли?»

`has_errors()` — «есть ли ошибки?»

Функция может считаться предикатом, только если она возвращает булевы значения `True` или `False`.

```
string = input()
is_castle = (string == 'Castle')
print(is_castle)
```

```
>> True
```

Задание №9

Напишите программу, которая считывает строку и проверяет является ли она словом 'Mister'

```
106     string = input()
107     is_mister = (string.capitalize() == 'Mister')
108     print(is_mister)
109
```

Задание №10

Напишите программу, которая принимает на вход:

1. Цвет доспехов(строка). Возможные варианты: red, yellow, black

2. Цвет щита(строка). Возможные варианты: red, yellow, black

Программа выводит True если цвет доспехов не красный и цвет щита чёрный. В остальных случаях выводит False

```
119 armor = input()
120 shield = input()
121
122 is_knight = (armor != 'red' and shield == 'black')
123 print(is_knight)
124
```

Задача №1: Лотерея

Реализуйте свою игру лотереи с выбором числа или набора чисел

Задача №2:

Человек вводит день и месяц своего рождения,
выведите, кем он является по знаку зодиака

Задача №3: Творческое задание

Придумать свою задачу на тему занятия. Обязательно
использовать несколько вложений if-else(elif)