Курс по основам программирования на Python

Действительные числа

- Действительные числа имеют тип **float()**

```
x = float(input())
print(x)
```

Функции

- round() – округление до целого по правилам математики
- abs() – взятие модуля числа
- floor() – округление вниз
- ceil() – округление вверх

```
import math
```

```
import math

x = math.ceil(4.2)
y = math.ceil(4.8)
print(x)
print(y)
```

Некоторые функции библиотеки math

math.fabs(X) - модуль X.

math.factorial(X) - факториал числа X.

math.exp(X) - e^X.

math.hypot(X, Y) - вычисляет гипотенузу треугольника с катетами X и Y (math.sqrt(x * x + y * y)).

math.degrees(X) - конвертирует радианы в градусы.

math.radians(X) - конвертирует градусы в радианы.

math.pi - pi = 3,1415926...

math.e - e = 2,718281...

Задача 1

- Дано положительное действительное число X. Выведите его первую цифру после десятичной точки.

Решение

```
x = float(input())
print(int(x * 10) % 10)
```

Настройка функции print()

- По умолчанию функция print() принимает несколько аргументов, выводит их через пробел, после чего ставит перевод строки. Это можно изменить, используя параметры sep(разделитель) и end(окончание).

```
print(1, 2, 3)
print(4, 5, 6)
print(1, 2, 3, sep=', ', end='. ')
print(4, 5, 6, sep=', ', end='. ')
print()
print(1, 2, 3, sep='', end=' -- ')
print(4, 5, 6, sep=' * ', end='.')
```

```
1 2 3
4 5 6
1, 2, 3. 4, 5, 6.
123 -- 4 * 5 * 6.
```

Задача 2

- Вводится 3 числа и строка.
- Используя sep и end, вывести в следующем виде:

```
12
13
14
Help
12 - 13 - 14 ! Строка: Help.
Help #
```

Задача

- Написать программу для решения линейного уравнения вида $kx+b=0$
- Написать программу для решения квадратного уравнения вида $ax^2+bx+c=0$

Решение

```
print("Решение линейных уравнений")
print("Введите коэффициенты k,b")
k = float(input())
b = float(input())
if k == 0 and b==0:
    print("Бесконечность")
elif k==0:
    print("Корней нет")
else:
    x = -b/k
    print("x = ", x)
```

Решение

```
print("Решение квадратных уравнений")
print("Введите коэффициенты a,b,c")
a = float(input())
b = float(input())
c = float(input())
d = b**2-4*a*c
if d < 0:
    print("Решений в действительной плоскости нет!")
elif d==0:
    print("x = ",(-b/(2*a)))
else:
    print("x1 = ",(-b+d**(1/2))/(2*a))
    print("x2 = ",(-b-d**(1/2))/(2*a))
```

Цикл for

- Так называемый цикл с параметром. В цикле **for** указывается переменная и множество значений, по которому будет пробегать переменная.

```
for i in 1, 2, 3, 'one', 'two', 'three':
    print(i)
```

При первых трех итерациях цикла переменная `i` будет принимать значение типа `int`, при последующих трех — типа `str`.

Цикл for

```
word = "child" # строка word
bag = ["knife","wallet", "pen","notebook"] # список bag
```

```
for letter in word:
    print letter # печатаем по букве из word

for item in bag:
    print item # печатаем по элементу из bag
```

Счётчик

- Инструкция **i += 1** эквивалентна конструкции **i = i + 1**. Такую сокращённую запись можно использовать при всех арифметических операциях: ***=, -=, /=, %=, //=** ...

Использование счётчика

```
sum = 0
n = 5
for i in range(1, n + 1):
    sum += i
print(sum)
```

Функция range

- Для повторения цикла некоторое заданное число раз n можно использовать цикл **for** вместе с функцией **range**.

```
for i in range(4):  # равносильно инструкции for i in 0, 1, 2, 3:
    # здесь можно выполнять циклические действия
    print(i)
    print(i ** 2)
# цикл закончился, поскольку закончился блок с отступом
print('Конец цикла')
```

Range

- Функция Range может принимать не один, а два параметра. Вызов **range(a,b)** означает, что индексная переменная будет принимать значения от **a** до **b-1**. Если a>b то цикл не будет выполнен ни разу.

```
sum = 0
n = 5
for i in range(1, n + 1):
    sum += i
print(sum)
```

Range с тремя параметрами

- Range(1, 100, 2) - цикл по всем нечётным числам.
- Range(100, 0, -1) – цикл по всем числам от 100 до 1

Получение индекса

```
week_days = ["Sun","Mon","Tue","Wed","Thu","Fri","Sat"]

for index in range(len(week_days)): # функция len(sequence) возвращает длину
    print week_day[index]
```

Break

- Команда break прекращает выполнение цикла и переводит выполнение программы на сроку следующую после цикла.

```
# пример команды break в цикле for
metals = ["Cu","Fe","Al","Au","U","Mg"]
for item in metals:
    print item
    if item == "Au":
        print "Ура! Я нашел золото!"
        break
```

Задача 1

Даны два целых числа A и B. Выведите все числа от A до B включительно, в порядке возрастания, если A < B, или в порядке убывания в противном случае.

Решение

```
A=int(input())
B=int(input())
if A<B:
    for i in range(A,B+1):
        print(i)
else:
    for i in range(A,B-1,-1):
        print(i)
```

Задача 2

Дано несколько чисел. Вычислите их сумму. Сначала вводите количество чисел N, затем вводится ровно N целых чисел. Какое наименьшее число переменных нужно для решения этой задачи?

Решение

```
N=int(input())
sum = 0
for i in range(N):
    number = int(input())
    sum += number
print(sum)
```

Задача 3

По данному натуральному n вычислите сумму $1^3+2^3+3^3+...+n^3$.

Решение

```
n=int(input())
sum = 0
for i in range(n+1):
    num=i**3
    sum += num
print(sum)
```

Задача 4

Факториалом числа n называется произведение 1 × 2 × ... × n.

Обозначение: n!.

По данному натуральному n вычислите значение n!. Пользоваться математической библиотекой math в этой задаче запрещено.

Решение

```
res = 1
n = int(input())
for i in range(1, n + 1):
    res *= i
print(res)
```

Задача 5

Дано N чисел: сначала вводится число N, затем вводится ровно N целых чисел. Подсчитайте количество нулей среди введенных чисел и выведите это количество. Вам нужно подсчитать количество чисел, равных нулю, а не количество цифр.

Решение

```
N = int(input())
k=0
for i in range(N):
  n = int(input())
  if n==0:
    k+=1
print(k)
```

Цикл while

- Цикл **while** позволяет выполнить одну и ту же последовательность действий, пока проверяемое условие истинно. Условие записывается до тела цикла и проверяется до выполнения тела цикла. Если это условие ложно, то выполнение цикла прекращается и управление передается на следующую инструкцию после тела цикла. Этот цикл используется, когда невозможно определить точное значение количества проходов исполнения цикла.

```
while условие:
    блок инструкций
```

Цикл while

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

Цикл while

- Определение количества цифр натурального числа

```
n = int(input())
length = 0
while n > 0:
    n //= 10  # это эквивалентно n = n // 10
    length += 1
print(length)
```

В языке Питон есть и другой способ решения этой задачи:

`length = len(str(i))` .

Управление циклом

- После цикла можно написать **else** и после него блок операций, который будет выполнен один раз после окончания цикла.

```
i = 1
while i <= 10:
    print(i)
    i += 1
else:
    print('Цикл окончен, i =', i)
```

Управление циклом

- Инструкция **break** может встречаться внутри цикла, но после того как программа встречает её, выполнение цикла прекращается и при этом, если была ветка **else**, она исполняться не будет.

```
a = int(input())
while a != 0:
    if a < 0:
        print('Встретилось отрицательное число', a)
        break
    a = int(input())
else:
    print('Ни одного отрицательного числа не встретилось')
```

Множественное присваивание

- Инструкция присваивания позволяет изменять сразу значения нескольких переменных. Главное, чтобы слева и справа от знака присваивания было одинаковое число элементов.

```
a = 0
b = 1
```

```
a, b = 0, 1
```

Множественное присваивание

- Удобно использовать, когда нужно обменять значения двух переменных.

Другой язык

```
a = 1
b = 2
tmp = a
a = b
b = tmp
print(a, b)
# 2 1
```

Python

```
a = 1
b = 2
a, b = b, a
print(a, b)
# 2 1
```

Задача 1

По данному целому числу N распечатайте все квадраты натуральных чисел, не превосходящие N, в порядке возрастания.

Тесты

Входные данные	Правильный ответ
50	1 4 9 16 25 36 49
10	1 4 9
9	1 4 9
4	1 4

Решение

```
s=int(input())
n=1
while n**2<=s:
    print(n**2, end=" ")
    n+=1
```

Задача 2

По данному натуральному числу N найдите наибольшую целую степень двойки, не превосходящую N. Выведите показатель степени и саму степень.

Тесты

Входные данные	Правильный ответ
50	5 32
10	3 8
8	3 8
7	2 4
1	0 1

Решение

```
n = int(input())
i = 1
while 2**i<=n:
    i += 1
print(i-1,2**(i-1))
```

Задача 3

Последовательность состоит из **различных** натуральных чисел и завершается числом 0. Определите значение второго по величине элемента в этой последовательности. Гарантируется, что в последовательности есть хотя бы два элемента.

Решение

```
first_max = int(input())
second_max = int(input())
if first_max < second_max:
    first_max, second_max = second_max, first_max
element = int(input())
while element != 0:
    if element > first_max:
        second_max, first_max = first_max, element
    elif element > second_max:
        second_max = element
    element = int(input())
print(second_max)
```