Ссылка на урок: https://coreapp.ai/app/player/lesson/67ae685d9fd0ee1384bc8cb3

(для учеников)

Занятие 10

Тема: **Реализация циклических алгоритмов на ЯП Python. Цикл while. Часть 1**

1. Теоретическая часть

На этом занятии мы рассмотрим ещё одну базовую алгоритмическую конструкцию – **цикл** и способы её реализации на языке Python. Как известно из школьного курса информатики, **цикл (повторение)** – это алгоритмическая конструкция, представляющая собой последовательность действий, выполняемых многократно. Последовательность действий, выполняемых многократно, называется ***телом цикла***.

Цикл в языке программирования представляет собой конструкцию, многократно выполняющую одну и ту же группу операторов.

Число повторений (**итераций**) цикла может быть либо задано заранее, либо зависеть от истинности некоторого условия.

В языке программирования Python может быть реализовано два вида цикла:

1) цикл с известным условием продолжения работы (с предусловием) — цикл **while**;
2) цикл с известным количеством повторений (цикл с параметром) — цикл **for**.

Цикл while является часто используемым и универсальным циклом в Python. Полный формат данного цикла:

while <условие>:
    <оператор1>
else:
    <оператор2>

Здесь:
<условие> — логическое выражение, *являющееся условием продолжения работы цикла*;
<оператор1>, <оператор2> — тело цикла.

Часть **else** является необязательной. Блок-схема работы цикла *while* представлена на рисунке 1.

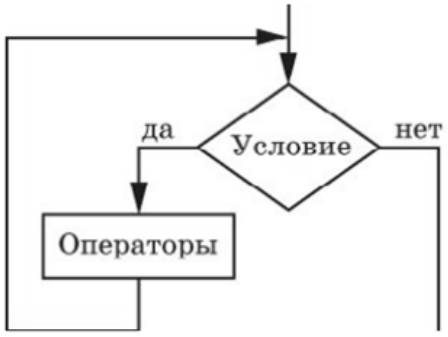


Рис. 1. Блок-схема цикла с предусловием while

Выполнение цикла *while* начинается с проверки условия. Если оно истинно (не равно false), выполняется оператор цикла. Если при первой же проверке выражение в условии равно false, цикл не выполнится ни разу. Если условие в цикле *while* никогда не станет ложным, то не будет причин остановки цикла и программа «зациклится». Чтобы этого не произошло, необходимо организовать момент выхода из цикла, т. е. ложность выражения в условии. Так, например, изменяя значение какой-нибудь переменной в теле цикла, можно довести логическое выражение до ложности. *Обратите внимание, что операторы тела цикла должны быть записаны с отступом.*

2. Примеры программ с использованием цикла while

Пример 1

```
i=5
while i<15:
    print(i)
    i=i+2 #эту команду можно писать так: i+=2
```

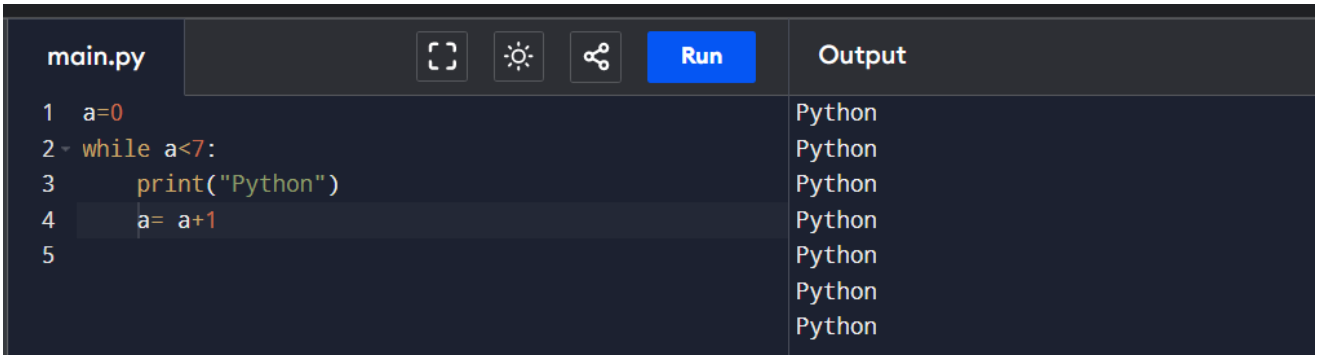
Результат выполнения программы:

В данном примере организован перебор значений переменной **i** с шагом 2. Условие работы цикла: i<15. В теле цикла происходит изменение (увеличение) переменной i, поэтому цикл не будет бесконечным. Цикл выполняется 5 раз: на экран выводится 5 значений переменной **i**, каждое значение больше предыдущего на 2.

Пример 2

```
a=0
while a<7:
    print("Python")
    a=a+1 #эту команду можно писать так: a+=1
```

Результат выполнения программы:



В данном примере организован перебор значений переменной **a** с шагом 1, пока истинно условие a<7. Начальное значение **a** было задано равным 0. Таким образом, цикл выполняется 7 раз: на экран 7 раз выводится слово "Python".

Пример 3. Получить частное **q** и остаток **r** от деления натурального числа **x** на натуральное число **y** без использования операции деления. Выполнить программу для пар значений x, y: (23, 5), (25, 4), (-10, 3).

ПОЯСНЕНИЕ. Операцию деления можно представить как последовательное вычитание делителя из делимого. Вычитать будем до тех пор, пока результат вычитания не станет меньше вычитаемого (делителя). В этом случае количество вычитаний будет равно частному от деления, а последняя разность – остатку от деления. Блок-схема алгоритма решения данной задачи представлена на рис. 1.

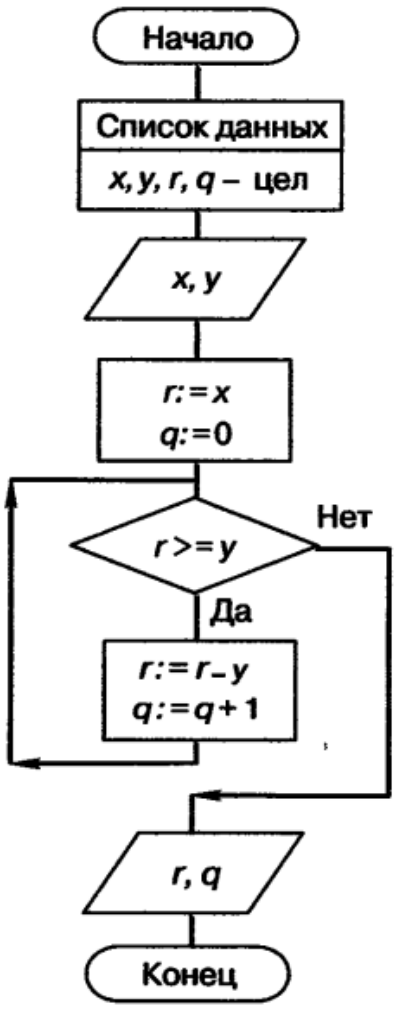


Рис. 1. Получение частного и остатка двух натуральных чисел

Пусть x = 23, y = 5

r	q	r >= y
23	0	Истина
18	1	Истина
13	2	Истина
8	3	Истина
3	4	Ложь

Ответ: q=4, r=3

Пусть x = 25, y = 4

r	q	r >= y
25	0	Истина
21	1	Истина
17	2	Истина
13	3	Истина
9	4	Истина
5	5	Истина
1	6	Ложь

Ответ: q=6, r=1

Пусть x = -10, y = 4

r	q	r >= y
-10	0	Ложь

Ответ: q=0, r = - 10. Как видим, для отрицательных чисел этот алгоритм без использования модуля не работает, и цикл не выполняется ни разу

Решение.

```
print('Частное и остаток')
x = int(input('Введите делимое x>>'))
y = int (input('Введите делитель y>>'))
r = x
q = 0
while r >= y: #цикл выполняется, пока остаток от деления больше или
#равен делителю
     r = r - y #от остатка отнимаем делитель
     q = q + 1 #частное увеличиваем на единицу
```

```
#возвращаемся к условию цикла: если условие стало false (ложным)
#выводим на экран значения q,r
print('Частное q =', q)
print('Остаток r =', r)
```

Результат выполнения программы при разных значениях x, y (интерпретатор Python 3.11.2):

```
File  Edit  Format  Run  Options  Window  Help
print('Частное и остаток')
x = int(input('Введите делимое x>>'))
y = int(input('Введите делитель y>>'))
r = x
q = 0

while r >= y:
    r = r - y
    q += 1

print('Частное q =', q)
print('Остаток r =', r)
```

```
File  Edit  Shell  Debug  Options  Window  Help
Python 3.11.2 (main, Jul 19 2024, 12:24:02) [GCC 12.
2.0] on linux
Type "help", "copyright", "credits" or "license()" f
or more information.
>>>
= RESTART: /home/itcube/Документы/Мои программы на П
итоне/chastnoe and ostatok.py
Частное и остаток
Введите делимое x>>23
Введите делитель y>>5
Частное q = 4
Остаток r = 3
>>>
= RESTART: /home/itcube/Документы/Мои программы на П
итоне/chastnoe and ostatok.py
Частное и остаток
Введите делимое x>>25
Введите делитель y>>4
Частное q = 6
Остаток r = 1
>>>
= RESTART: /home/itcube/Документы/Мои программы на П
итоне/chastnoe and ostatok.py
Частное и остаток
Введите делимое x>>-10
Введите делитель y>>3
Частное q = 0
Остаток r = -10
>>>
                                          Ln: 21  Col: 21
```

Пример 4. Разложить натуральное число **k** на простые множители и вывести их на экран в строку через пробел. Выполнить программу для значений k, равных 8, 27, 99, 54.

ПОЯСНЕНИЕ. Разложить на простые множители — значит представить натуральное число в виде произведения простых множителей (чисел). Например:

20 = 2 × 2 × 5. Простые множители — это простые числа, которые делят натуральное число нацело (без остатка).

Простое — это число, которое делится только на само себя и единицу (минимальное простое число — это 2).

Любое натуральное число **n**, большее единицы, можно разложить на произведение простых чисел (источник – Онлайн школа Skysmart :https://skysmart.ru/articles/mathematic/razlozhenie-chisel-na-prostye-mnozhiteli).

Блок-схема алгоритма решения данной задачи представлена на рис. 2.

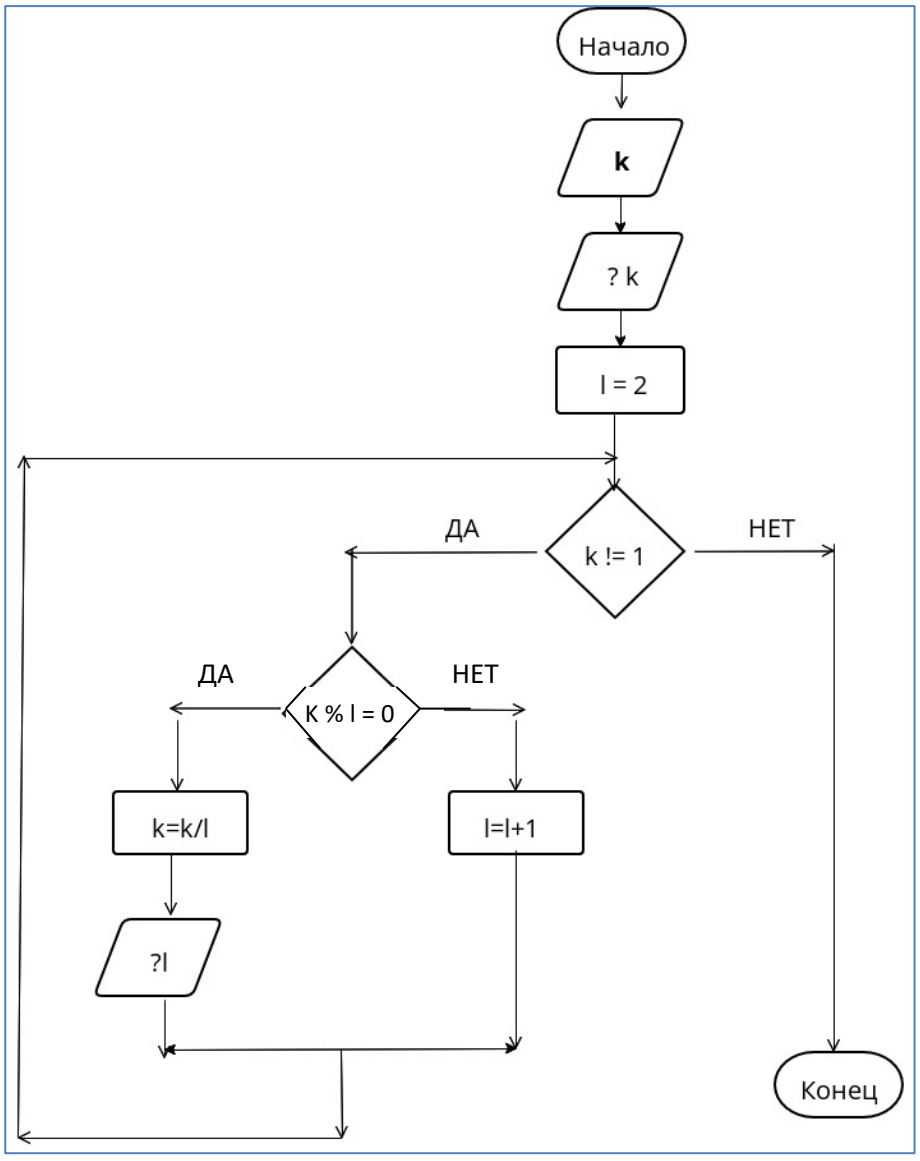


Рис. 2. Разложение натурального числа на простые множители

Пусть k=8

k	l	k % l	На экран
8	2	0	2
4	2	0	2
2	2	0	2
1			

Пусть k=27

k	l	k % l	На экран
27	2	1	
27	3	0	3
9	3	0	3
3	3	0	3
1			

Пусть k=99

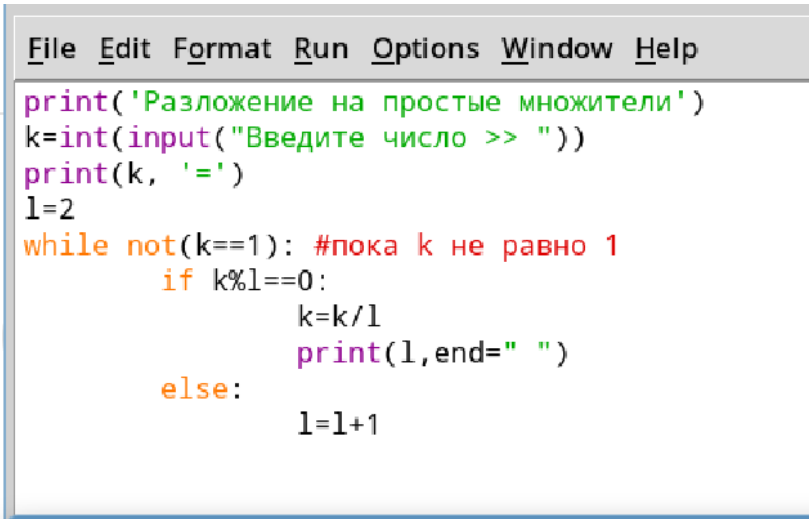
k	l	k % l	На экран
99	2	1	
99	3	0	3
33	3	0	3
11	3	2	
11	4	3	
11	5	1	
11	6	5	
11	7	4	
11	8	3	
11	9	2	
11	10	1	
11	11	0	11

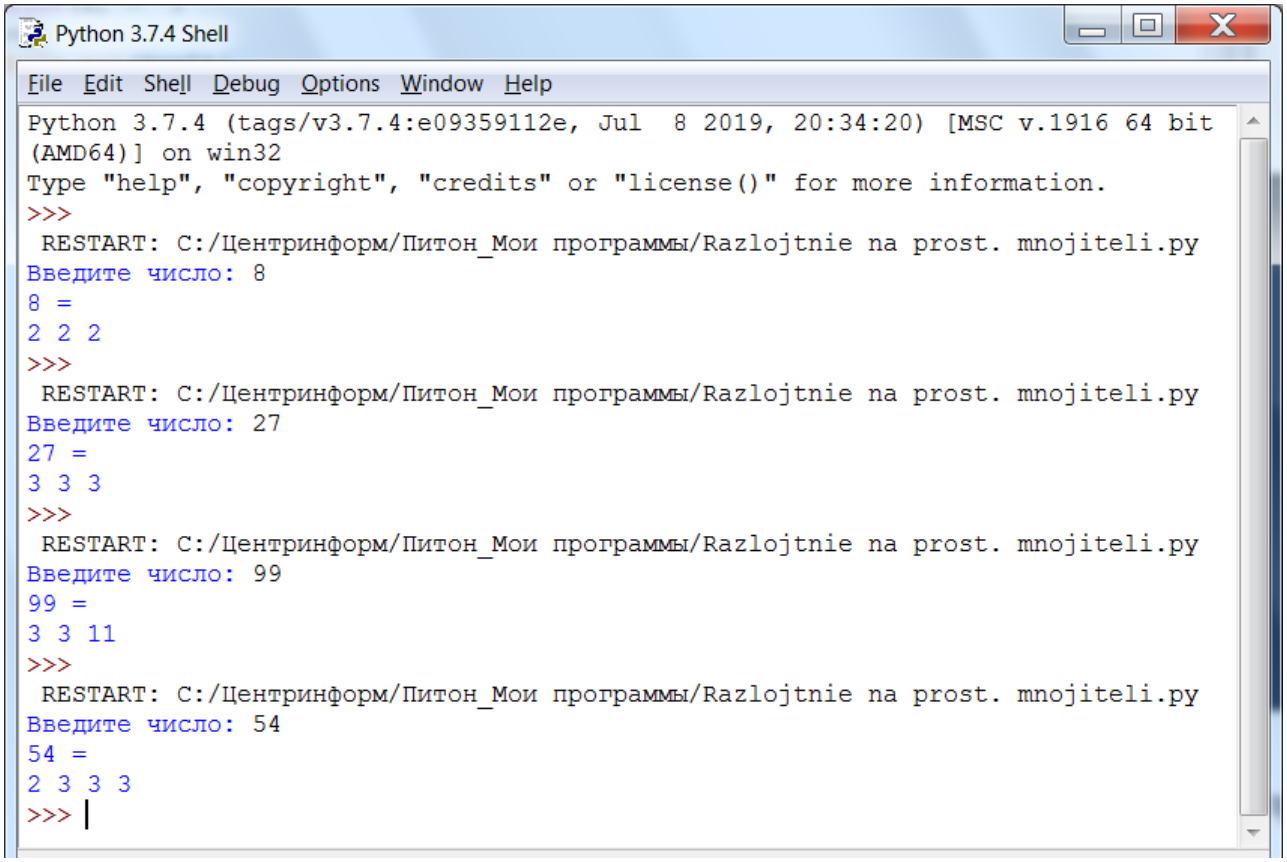
Решение.

```
print('Разложение на простые множители')
k=int(input("Введите число: "))
```

```
print(k,'=')
l=2 #присваиваем переменной l значение 2 (минимальное простое число)
while not(k==1): #пока k не равно 1
    if k%l==0:#проверяем, является ли переменная l простым множителем,
#если Да, то
        k=k/l #число k делим на простой множитель
        print(l,end=" ")#выводим на экран простой множитель
    else: #иначе
        l+=1#значение переменной l увеличиваем на единицу
#возвращаемся к условию цикла
```

Результат выполнения программы при разных значениях k (интерпретатор Python 3.7.4):





2. Просмотр учебного видеоролика «Цикл while» (с целью расширения знаний, наглядного представления и закрепления материала):

https://rutube.ru/video/b88cdc659e32186e3c35bc07752f59c8/?r=plwd (длина ролика: 9:55).

3. Закрепление

ЗАДАНИЕ 1.

Что будет выведено на экран в результате работы следующих программ?

а)	б)	в)

s = 0
i = 0
while i < 5:
  i += 1
  s += i
	s = 0
	i = 0
	while i < 5:
	  i += 1
	s += i
		s = 0
		i = 2
		while i > 1:
		  s = s + 1 / i
		  i = i - 1

ЗАДАНИЕ 2.

Написать программу, рассчитывающую среднее арифметическое группы оценок (средний балл). Количество оценок заранее не известно. Для завершения ввода введите любую оценку со знаком минус.

!!! Решение к заданиям 1, 2 будет на следующей странице. Но сначала попробуйте решить самостоятельно.

РЕШЕНИЕ

Задание 1.

а) i = 5, s = 15	б) i = 5, s = 5	в) i = 1, s = 0,5

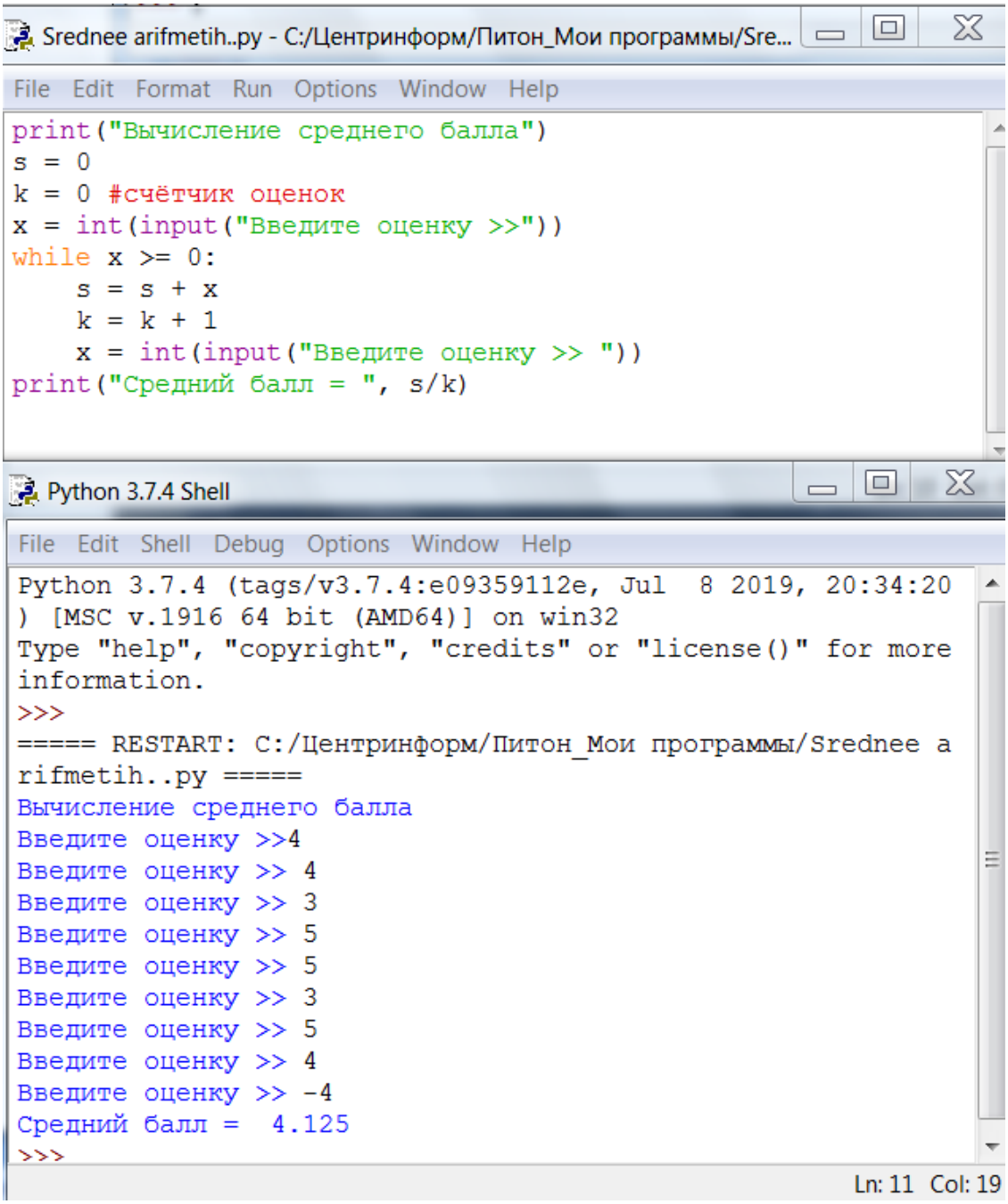
Примечание. В примере под буквой **б** вычисление суммы выполняется за пределами цикла, то есть она не накапливается.

Задание 2.

Решение.

```
print("Вычисление среднего балла")
s = 0 #сумма чисел изначально равна 0
k = 0 #счётчик оценок
x = int(input("Введите оценку>>"))
while x >= 0: #цикл while будет выполняться, пока введённая оценка
#будет положительным числом
    s = s + x #прибавляем введённую оценку к предыдущему
#значению суммы
    k = k + 1 #увеличиваем счётчик оценок на 1
    x = int(input("Введите оценку> ")) #запрашиваем ввод
следующей оценки
print("Средний балл =  ", s/k) #выводим среднюю оценку
```

Результат выполнения программы:



Платформа Programiz