Питон, циклы, условия

Хашин С.И.
http://math.ivanovo.ac.ru/dalgebra/Khashin/index.html

Ивановский университет

.

Питон, циклы, условия, файлы

.

Иваново-2023

План

for

```
for i in range(5): print(i, end=' ')      # 0 1 2 3 4
print() # новая строка
for i in range(4,10): print(i, end='-')   # 4-5-6-7-8-9-
print()
for i in range(1,10,3): print(i, end=' ') # 1 4 7
print()
for i in range(10,5,-1): print(i, end=' ')# 10 9 8 7 6
print()
for i in range(10,1,-3): print(i, end=' ')# 10 7 4
```

if

Выравнивание!

```
if num > 100:
    if num < 150:
        print "Число больше ста, но меньше ста пятидесяти"
    elif num <  200:
        print "Число больше ста, но меньше двухсот"
elif num > 50:
    if num < 90:
        print "Число больше пятидесяти, но меньше девяноста"
    else:
        print "Число больше пятидесяти и больше девяноста"
else:
    print "Число меньше пятидесяти"
```

while

```
n=10
while n>3:
    print(n, end=' ')
    n-=1
```

```
Результат: 10 9 8 7 6 5 4
```

Списки

```
xs = [3, 1, 2]    # создаем список
print(len(xs)     # Длина списка, 3
print(xs, xs[2]) # Prints "[3, 1, 2] 2"
print(xs[-1])     # Prints "2"
xs[2] = 'foo'     # elements of different types
print(xs)         # Prints "[3, 1, 'foo']"
xs.append('bar') # Add to the end
print(xs)         # Prints "[3, 1, 'foo', 'bar']"
x = xs.pop()      # Remove and return the last element
print(x, xs)      # Prints "bar [3, 1, 'foo']"
xs.extend(L)      # добавляем в конец xs все элементы из L
x in xs       # Проверка принадлежности элемента списку.
              # Можно проверять принадлежность подстроки ст
x not in xs       # = not x in xs
xs + xs1          # Конкатенация списков
```

Списки

```
 xs*n или n*xs     # n раз повторяем xs.
 min(xs)           # Наименьший элемент xs
 max(xs)           # Наибольший элемент xs
 xs[i:j:d] = t     # Срез от i до j (с шагом d) заменяется на
 del xs[i:j:d]     # Удаление элементов среза из списка
```

цикл:

```
 L = [54,67,48,99,27]
 for v in L: print(v, end=' ')
 > 54 67 48 99 27

 for i, v in enumerate(L): print(i, v)
 > 0 54
 > 1 67
 > 2 48
 > 3 99
 > 4 27
```

Списки

Список из трёх нулей:

```
L = [0] * 3
print(L)          # [0, 0, 0]
L[1]+=1
print(L)          # [0, 1, 0]
L[2] = 29
print(L)          # [0, 1, 29]
L[0] = [2,3,4]
print(L)          # [[2, 3, 4], 1, 29]
```

Список списков

Пример

```
L = [[]] * 3
print(L)          # [[], [], []]
L[1].append(17)
```

Список списков

Пример

```
L = [[]] * 3
print(L)          # [[], [], []]
L[1].append(17)
print(L)          # [[17], [17], [17]]
L[2] = 0
print(L)          # [[17], [17], 0]
```

А как правильно сделать список пустых списков?

```
L = [[] for _ in range(3)]
print(L)          # [[], [], []]
L[1].append(17)
print(L)          # [[], [17], []]
L[2] = 0
print(L)          # [[], [17], 0]
```

Треугольник Паскаля

```
N = 6
L = [[], [], [1,1]]
for i in range(3,N):
    L1 = [1]
    for j in range(1,i-1):
        L1.append(L[-1][j-1] + L[-1][j])
    L1.append(1)
    L.append(L1)
for i,L1 in enumerate(L): print(i, L1)
```

```
0 []
1 []
2 [1, 1]
3 [1, 2, 1]
4 [1, 3, 3, 1]
5 [1, 4, 6, 4, 1]
```

Сортировка списка

```
L = [5,1,9,0,5, 7,1]
print(L.sort())
> None
print(L)
> [0, 1, 1, 5, 5, 7, 9]
L.sort(reverse=True)
print(L)
> [9, 7, 5, 5, 1, 1, 0]
```

Сортировка списка строк L по 3-му символу:

```
L = ['abcd', 'eeeeee', '0123456', 'saro8s', 'lk;a']
L = sorted(L, key=lambda el: el[3:])
print(L)
> ['0123456', 'lk;a', 'abcd', 'eeeeee', 'saro8s']
```

Списки по условию

```
L = [i**3 for i in range(50) if i%5==0 and i%3 != 0]
> [125, 1000, 8000, 15625, 42875, 64000]
```

Чтение файла

Создайте файл 00.txt:

```
with open('00.txt', 'w') as f:
    f.write(''' Строка-заголовок
 a 2.3 6 8
 b   9  8   0
 c 1.1 2 2''')
```

Теперь его прочитаем:

```
letters, matrix = [], []
with open('00.txt', 'r') as f:
    f.readline()
    for line in f:
        ww = line.split()
        letters.append(ww[0].strip())
        matrix.append(list(map(float, ww[1:])))
for i, v in enumerate(matrix): # вывод результата
    print(i, letters[i], v)
```

Задание

Создайте файл с таблицей синусов и косинусов от 0 до 6.28 с шагом 0.01

```
x    sin(x)
0.00 0.000000
0.01 0.010000
0.02 0.019999
0.03 0.029996
0.04 0.039989
0.05 0.049979
0.06 0.059964
0.07 0.069943
0.08 0.079915
0.09 0.089879
...
```

Задание

Создайте файл с таблицей синусов и косинусов от 0 до 6.28 с шагом 0.01

```
import math
with open('sin.txt', 'w') as f:
  f.write(' x    sin(x)\n') # строка заголовка
  for ix in range(628):
    x = ix/100
    f.write(f'{x:5.2f} {math.sin(x):8.6f}\n')
```

Файлы в папке

```
import os
# так НЕЛЬЗЯ!:
files = os.listdir(r'D:\w\IVGU\slides\Python\')
# список файлов в папке
files = os.listdir(r'D:\w\IVGU\slides\Python')
print(files)
> ['for_list.aux', 'for_list.log', 'for_list.nav', ...]
for filename in files:
    if filename[-4:]=='.pdf': print(filename)
> for_list.pdf }
> intro.pdf
> numb_th.pdf
```

Задание

Задание 1. Напечатайте список файлов *.py в вашей текущей папке.

Задание 2. Напечатайте список файлов *.py в папке C:/Program Files/Python/Lib/idlelib/

Гипотеза 3n+1

Чётные: $n \to n/2$

Нечётные: $n \to 3n+1$

$7 \to 22 \to 11 \to 34 \to 17 \to 52 \to 26 \to 13 \to 40 \cdots \to 5 \to 16 \ldots 1$

```
def f31(n):
    k = 0
    while True:
        while not n & 1: n //= 2
        if n==1: break
        n = 3*n+1
        k += 1
    return k

for n in range(1,30):
    print(n, f31(n))
```

Задание. Найти наибольшее f31 при n< 1 млн.

Счастливые билеты

Билет $(a_0a_1a_2, b_0b_1b_2)$ назвывается p-счастливым, если

$$a_0^p + a_1^p + a_2^p = b_0^p + b_1^p + b_2^p.$$

Тривиальное счастье, когда цифры одинаковые, но в другом порядке не считаем.

```
p=3
# a1<b1,  a1<=a2<=a3,  b1<=b2<=b3
for a1 in range(0,9):
 for a2 in range(a1,10):
   for a3 in range(a2,10):
     L = a1**p + a2**p + a3**p
     for b1 in range(a1+1,10):
       for b2 in range(b1,10):
         for b3 in range(b2,10):
           R = b1**p + b2**p + b3**p
           if L==R: print(a1,a2,a3, b1,b2,b3)
```

$16/64 = 1/4$

Некоторых дроби, например 16/64 очень легко сократить: достаточно вычеркнуть одинаковые цифры: $16/64 = 1/4$.
Проверим, всегда ли так: $98/49 = 8/4$.
Найти все остальные такие дроби из двузначных чисел.
Тривиальные случаи (типа 77/77=7/7) не рассматриваем.
Более сложно: найти аналогичные дроби с трёхзначными числами.