Python

Базовый тренинг

Рузин Алексей
ruzin@me.com

Цель

- Познакомиться с языком
- Научиться самостоятельно разрабатывать приложения

Занятие 2

Введение в Python (продолжение)

Задача 1

Дан список целых чисел, например:
[5, 13, 23, 32, 16, 1].

Требуется напечатать минимальное и максимальное числа из этого списка.

Решение:

```
a = [5, 13, 23, 32, 16, 1]
print(min(a), max(a))
```

Задача 2

Вывести все числа Фибоначи меньше 100.

Числа Фибоначи образуются путем сложения двух предыдущих чисел. При этом первые два числа равны единице: 1, 1, 2, 3, 5, 8, …

Решение:

```
fib = []
while True:
    if len(fib) < 2:
        fib.append(1)
    else:
        fib.append(fib[-1] + fib[-2])
    if fib[-1] >= 100:
        break
    print(fib[-1])
```

Задача 3

Дана строка a=«PYTHON», необходимо напечатать строку, где буквы идут задом наперед.

Решение:

```
a=«PYTHON»
print(a[::-1])
```

Задача 4

Напечатать все простые числа меньше 1000.

Решение:

```
simple = [2]
for x in range(3, 1000):
    is_simple = True
    for t in simple:
        is_simple = x % t
        if not is_simple:
            break
    if is_simple:
        simple.append(x)
print(simple)
```

Задача 5

Напечатать все простые числа Фибоначи меньше 1000.

Решение:

```
# берем массивы fib и simple из предыдущих задач
for x in fib:
    if x in simple:
        print(x)
```

Задача 6

Дана произвольная строка. Найти и напечатать самый длинный фрагмент этой строки, который встречается больше одного раза (фрагменты не должны пересекаться).

Решение:

```
def get_biggest_repetition(a):
    l = len(a)
    cl = len(a)//2
    while cl > 0:
        luft = l - cl*2
        for x in range(1 + luft):
            s1 = a[x:x+cl]
            for y in range(1 - x + luft):
                s2 = a[x+cl+y:x+cl+y+cl]
                if s1 == s2:
                    return s1
        cl -= 1
print(get_biggest_repetition("абвгдежзиклмнопрстуфх") or "нет повторений")
print(get_biggest_repetition("это произвольная строка") or "нет повторений")
```

План

- Словари, множества, кортежи (dict, set, tuple).
- Итерации по словарям, множествам, кортежам.
- Функции.
- Классы.
- Исключения.
- Модули и Пакеты.

Tuple (кортеж)

- в отличие от списка не изменяемые

- a, b, c = (1, 2, 3)

- for i in (1, 2, 3):
    print(i)

Множества

- {}
- |, &, -, ^
- add, update, remove

```
a = {1,2,3}
b = {2,3,4}

print a | b

print a & b

print a - b

print a ^ b
```

Словарь

- {'key': value}
- что может быть ключом
- for … in
  .keys(), .iterkeys(),
  .values(), .itervalues(),
  .items(), .iteritems()

Функции

- определение:
  def func(a):
      return a + 1

- вызов:
  func(3)

- параметры по-умолчанию
  def func(a=3, b=4):
      return a+b

- вызов с параметром по-умолчанию:
  func(b=88)

Классы

- ООП: Инкапсуляция, Наследование, Полиморфизм

- определение:
  class MyClass:
      pass

- конструктор:
  class MyClass:
      def __init__(self):
          super().__init()

Классы

- создание экземпляров:
  a = A()

- обращение к методам:
  a.method()

Исключения

- try:
  …
  except Exception as e:
  print(e)
  else:
  print(‘no exception’)
  finally:
  print(‘finally’)

- raise Exception(‘some message’)

Модули и пакеты

- .py файл - модуль
- каталог с файлом __init__.py - пакет
- **import** my_module
- **from** my_module **import** something

Задача 1

- Есть словарь вида {‘c’: 4, ‘k’: 1, ‘m’: 7,…}. Требуется вывести ключи в порядке увеличения соответствующих им значений.

Задача 2

- Написать функцию вычисления факториала n!
  без использования инструкции if, for, while.
  n! = 1*2*3*…*n

Задача 3

- Написать простой калькулятор вычисляющий выражение для исходной строки вида:
s = «3+7-1/5+3*6».
Все числа состоят из одной цифры. Приоритетов у операций +,-,/,* - т.е. выполняются по мере появления (как если бы мы считали на обычном калькуляторе).
Операции +,-,/,* оформить в виде классов