Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Автоматизированные системы обработки информации и управления»



**Отчет**

**Лабораторная работа № 3**

**По курсу «Разработка интернет приложений»**

**ИСПОЛНИТЕЛЬ:**

Группа ИУ5-55Б

Старых Ф. А.

"14" Декабря 2020 г.

**ПРЕПОДАВАТЕЛЬ:**

Гапанюк Ю.Е.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

"\_\_"\_\_\_\_\_\_\_\_\_\_\_2020 г.

Москва 2020

1. **Общее задание**

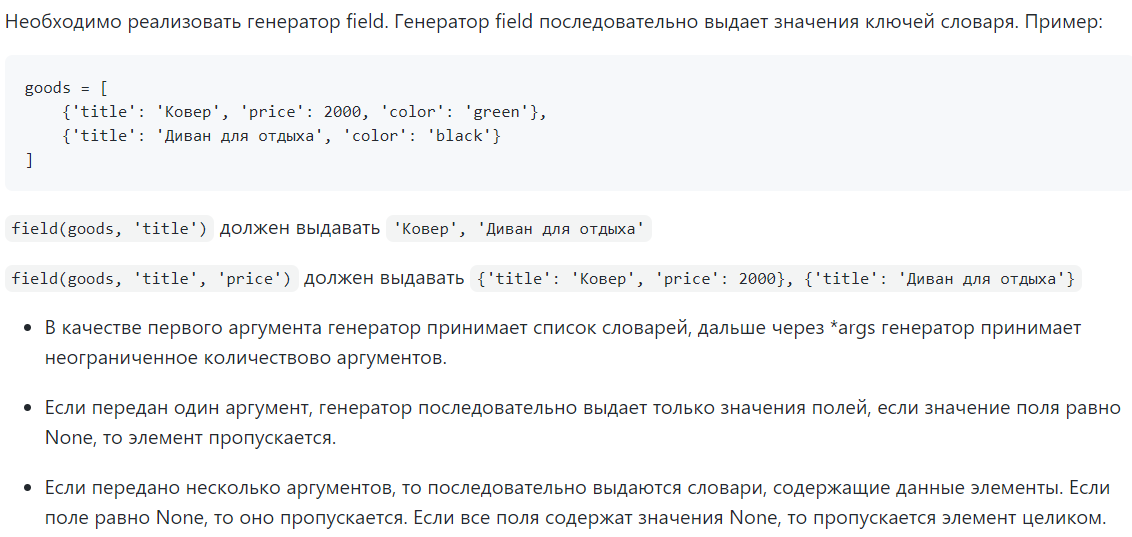
Задание лабораторной работы состоит из решения нескольких задач.

Файлы, содержащие решения отдельных задач, должны располагаться в пакете lab\_python\_fp(Lib). Решение каждой задачи должно раполагаться в отдельном файле.

При запуске каждого файла выдаются тестовые результаты выполнения соответствующего задания.

1. **Задача 1**

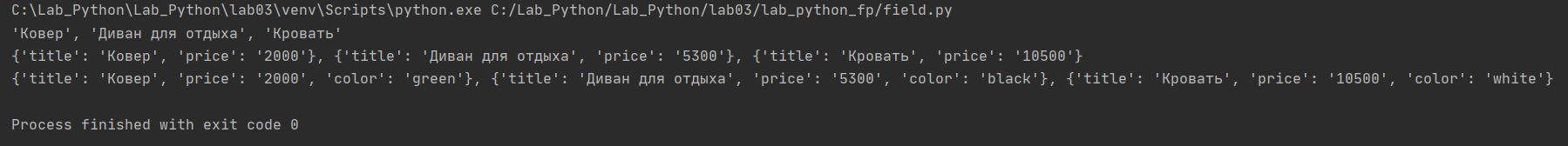
Описание:



Код:

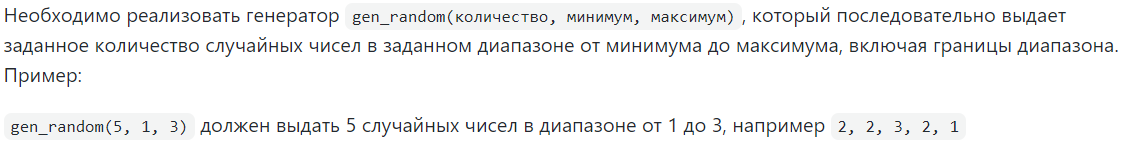
goods = [  
 {'title': 'Ковер', 'price': '2000', 'color': 'green'},  
 {'title': 'Диван для отдыха', 'price': '5300', 'color': 'black'},  
 {'title': 'Кровать', 'price': '10500', 'color': 'white'}  
]  
  
  
def field(items, \*args):  
 assert len(args) > 0, 'Не переданы аргументы полей словаря'  
 count = 0  
 for i in items:  
 count += 1  
 if len(args) == 1:  
 print('\'', end='')  
 print(i.get(\*args), end='')  
 if count < len(items):  
 print('\'', end=', ')  
 else:  
 print('\'')  
 else:  
 k = 0  
 print('{\'', end='')  
 while k < len(args):  
 temp\_args = args[k]  
 k += 1  
 print(temp\_args, end='')  
 print('\': \'', end='')  
 print(i.get(temp\_args), end='')  
 if k < len(args):  
 print('\', \'', end='')  
 else:  
 print('\'', end='')  
 if count < len(items):  
 print('}, ', end='')  
 else:  
 print('}')  
  
  
def main():  
 # field(goods)  
 field(goods, 'title')  
 field(goods, 'title', 'price')  
 field(goods, 'title', 'price', 'color')  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

Скрины:



1. **Задача 2**

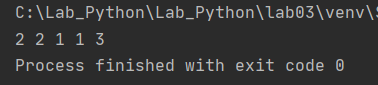
Описание:



Код:

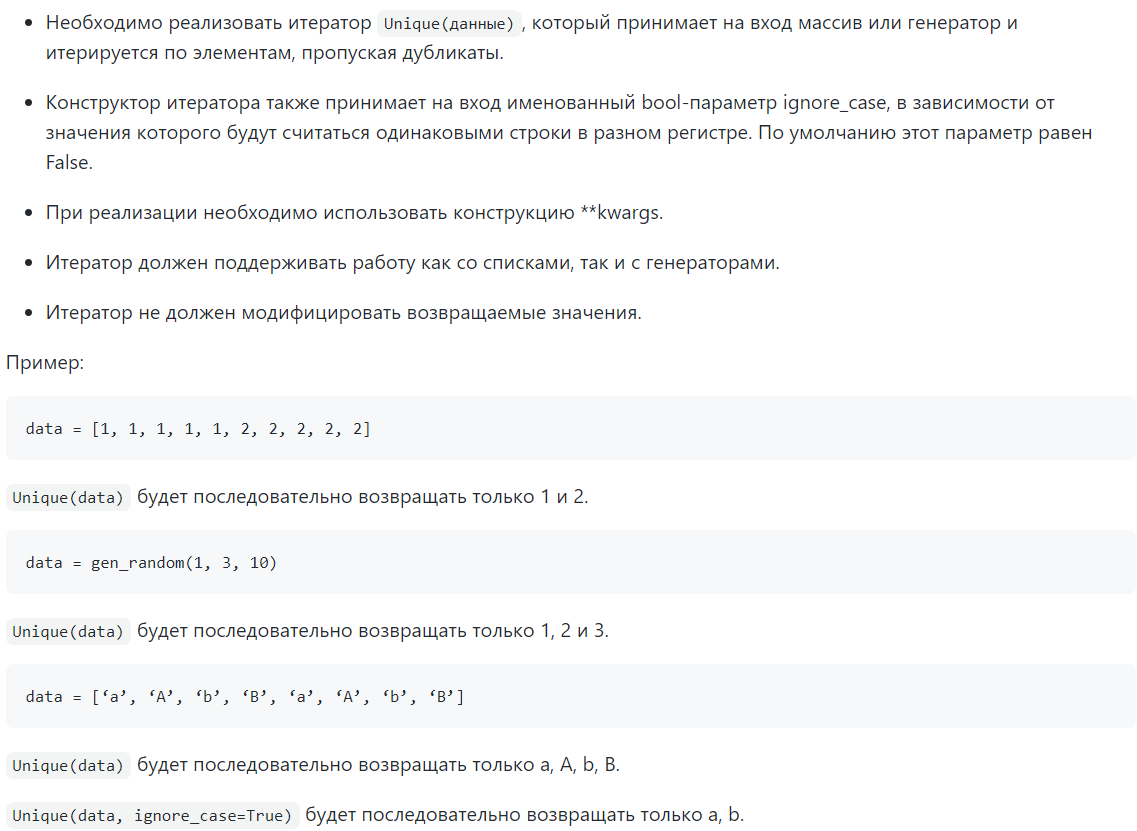
import random  
  
  
def gen\_random(num\_count, begin, end):  
 for i in range(num\_count):  
 yield random.randint(begin, end)  
  
  
def main():  
 gen = gen\_random(5, 1, 3)  
 for i in gen:  
 print(i, end=' ')  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

Скрины:



1. **Задача 3**

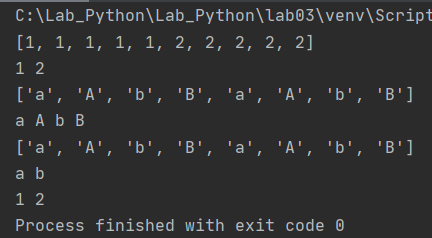
Описание:



Код:

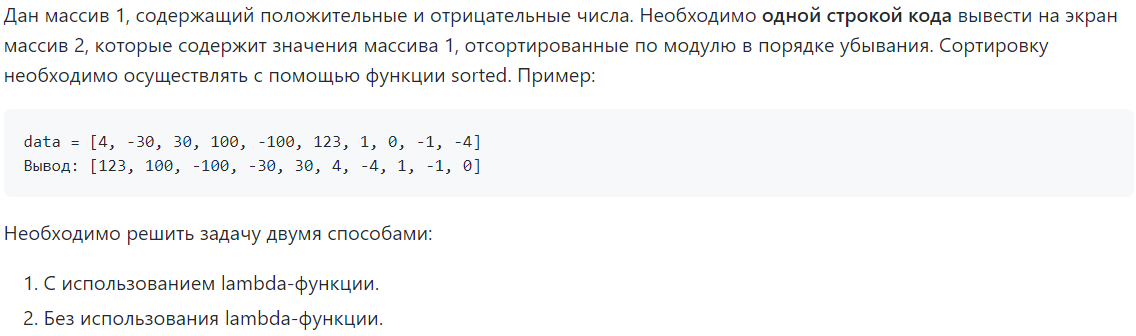
from lab\_python\_fp.gen\_random import gen\_random  
  
  
class Unique(object):  
 def \_\_init\_\_(self, items, \*\*kwargs):  
 self.used\_elements = set()  
 self.items = items  
 self.counter = 0  
 if len(kwargs) != 0:  
 self.ignore\_case = kwargs  
 else:  
 self.ignore\_case = False  
  
 def \_\_next\_\_(self):  
 while True: # как это работает?  
 for item in self.items:  
 temp\_item = item  
 self.counter += 1  
 if (temp\_item not in self.used\_elements) \  
 and not(self.ignore\_case and temp\_item.swapcase() in self.used\_elements):  
 self.used\_elements.add(temp\_item)  
 return temp\_item  
 else:  
 raise StopIteration  
  
 def \_\_iter\_\_(self):  
 return self  
  
  
def main():  
 data1 = [1, 1, 1, 1, 1, 2, 2, 2, 2, 2]  
 print(data1)  
 itr1 = Unique(data1)  
 for i1 in itr1:  
 print(i1, end=' ')  
 print('\n', end='')  
 data2 = ['a', 'A', 'b', 'B', 'a', 'A', 'b', 'B']  
 print(data2)  
 itr2 = Unique(data2)  
 for i2 in itr2:  
 print(i2, end=' ')  
 print('\n', end='')  
 print(data2)  
 itr3 = Unique(data2, ignor\_case=True)  
 for i3 in itr3:  
 print(i3, end=' ')  
 print('\n', end='')  
 data3 = gen\_random(5, 1, 3)  
 itr4 = Unique(data3)  
 for i4 in itr4:  
 print(i4, end=' ')  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

Скрины:



1. **Задача 4**

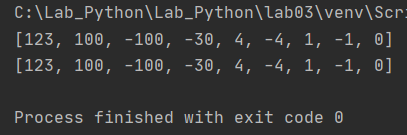
Описание:



Код:

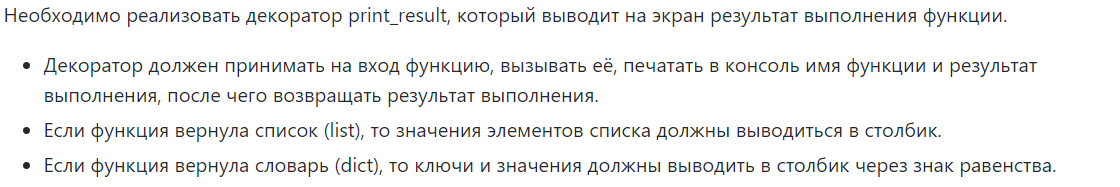
def sort(x):  
 return abs(x)  
  
  
def main():  
 data = [4, -30, 100, -100, 123, 1, 0, -1, -4]  
  
 result = sorted(data, key=sort, reverse=True)  
 print(result)  
  
 result\_with\_lambda = sorted(data, key=lambda x: abs(x), reverse=True)  
 print(result\_with\_lambda)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

Скрины:



1. **Задача 5**

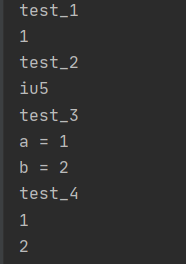
Описание:



Код:

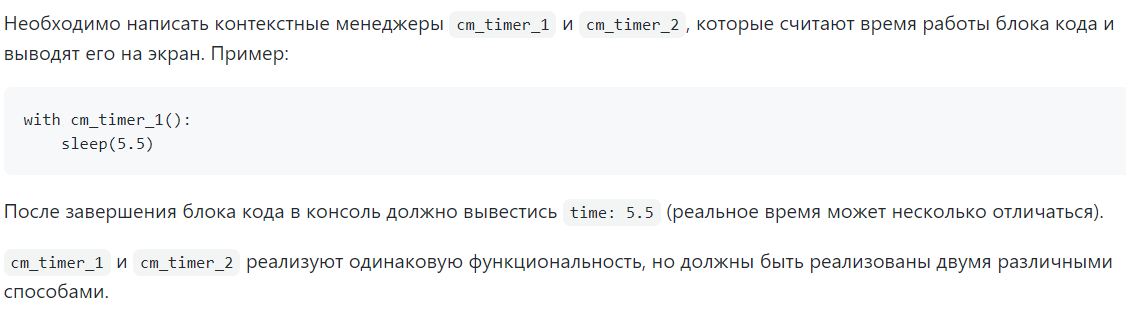
def print\_result(func\_to\_decorate):  
  
 def decorated\_func():  
 print(func\_to\_decorate.\_\_name\_\_)  
 result = func\_to\_decorate()  
 if type(result) is list:  
 for i in result:  
 print(i)  
 elif type(result) is dict:  
 for i in result:  
 print(i, result.get(i), sep=' = ')  
 else:  
 print(result)  
  
 return decorated\_func()  
  
  
@print\_result  
def test\_1():  
 return 1  
  
  
@print\_result  
def test\_2():  
 return 'iu5'  
  
  
@print\_result  
def test\_3():  
 return {'a': 1, 'b': 2}  
  
  
@print\_result  
def test\_4():  
 return [1, 2]  
  
  
def main():  
 print('!!!!!!!!')  
 test\_1()  
 test\_2()  
 test\_3()  
 test\_4()  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

Скрины:



1. **Задача 6**

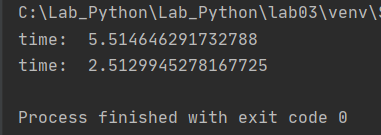
Описание:



Код:

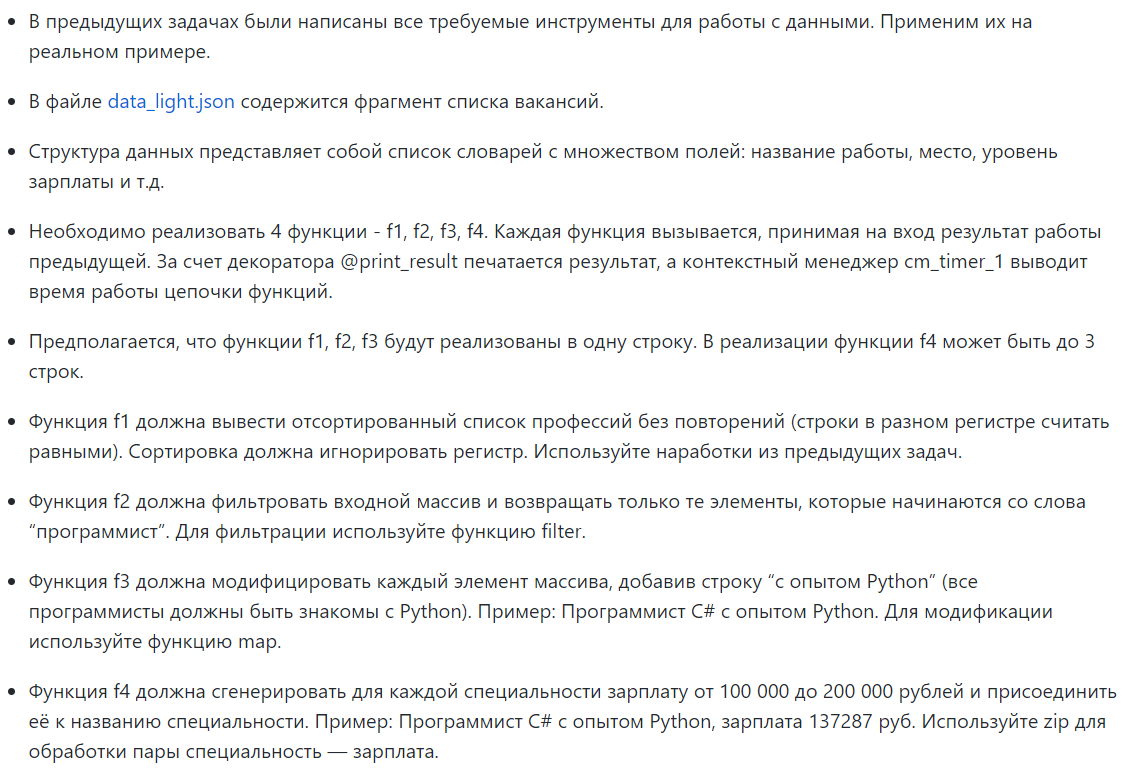
import time  
from contextlib import contextmanager  
  
  
class cm\_timer\_1:  
  
 def \_\_init\_\_(self):  
 self.begin\_time = time.time()  
  
 def \_\_enter\_\_(self):  
 pass  
  
 def \_\_exit\_\_(self, exc\_type, exc\_val, exc\_tb):  
 if exc\_type is not None:  
 print(exc\_type, exc\_val, exc\_tb)  
 else:  
 print('time: ', time.time() - self.begin\_time)  
  
  
@contextmanager  
def cm\_timer\_2():  
 begin\_time = time.time()  
 yield 1  
 print('time: ', time.time() - begin\_time)  
  
  
def main():  
 with cm\_timer\_1():  
 time.sleep(5.5)  
  
 with cm\_timer\_2():  
 time.sleep(2.5)  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

Скрины:



1. **Задача 7**

Описание:



Код:

from lab\_python\_fp.cm\_timer import cm\_timer\_1  
from lab\_python\_fp.print\_result import print\_result  
from lab\_python\_fp.unique import Unique  
from lab\_python\_fp.field import field  
from lab\_python\_fp.gen\_random import gen\_random  
import re  
import json  
import sys  
  
  
path = 'data\_light.json'  
  
  
with open(path) as f:  
 data = json.load(f)  
  
  
@print\_result  
def f1(arg):  
 return Unique(field(arg, 'job-name'), ignore\_case=True)  
  
  
@print\_result  
def f2(arg):  
 return filter(lambda x: re.search('Программист', x) or re.search('программист', x), arg)  
  
  
@print\_result  
def f3(arg):  
 return list(map(lambda x: x + ' с опытом Python', arg))  
  
  
@print\_result  
def f4(arg):  
 price = gen\_random(len(arg), 100000, 200000)  
 res = list(zip(arg, (list(map(lambda x: ', зарплата ' + x + ' руб', ''.join(str(list(price)))[1:-1].split(', '))))))  
 return [''.join(i) for i in res]  
  
  
def main():  
 with cm\_timer\_1():  
 f4(f3(f2(f1(data))))  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

Скрины:

