**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Радиотехнический»

Кафедра «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по лабораторной работе №2

«**Функциональные возможности языка Python.**»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы РТ5-31Б: |  | преподаватель кафедры ИУ5 |
| Молодцов М.В. |  | Гапанюк Ю. Е. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2024 г.

Постановка задачи

1. Необходимо реализовать генератор field. Генератор field последовательно выдает значения ключей словаря.
2. Необходимо реализовать генератор gen\_random(количество, минимум, максимум), который последовательно выдает заданное количество случайных чисел в заданном диапазоне от минимума до максимума, включая границы диапазона.
3. Необходимо реализовать итератор Unique(данные), который принимает на вход массив или генератор и итерируется по элементам, пропуская дубликаты.Конструктор итератора также принимает на вход именованный bool-параметр ignore\_case, в зависимости от значения которого будут считаться одинаковыми строки в разном регистре. По умолчанию этот параметр равен False.При реализации необходимо использовать конструкцию \*\*kwargs.Итератор должен поддерживать работу как со списками, так и с генераторами.Итератор не должен модифицировать возвращаемые значения.
4. Дан массив 1, содержащий положительные и отрицательные числа. Необходимо **одной строкой кода** вывести на экран массив 2, которые содержит значения массива 1, отсортированные по модулю в порядке убывания. Сортировку необходимо осуществлять с помощью функции sorted.
5. Необходимо реализовать декоратор print\_result, который выводит на экран результат выполнения функции.
6. Необходимо написать контекстные менеджеры cm\_timer\_1 и cm\_timer\_2, которые считают время работы блока кода и выводят его на экран
7. В предыдущих задачах были написаны все требуемые инструменты для работы с данными. Применим их на реальном примере

Текст программы

Файл «field.py»

def field(items, \*args):  
 assert len(args) > 0  
  
 for item in items:  
 if len(args) == 1:  
 v = item.get(args[0])  
 if v is not None:  
 yield v  
  
 else:  
 filtered\_item = {key: item.get(key) for key in args if item.get(key) is not None}  
 if filtered\_item:  
 yield filtered\_item  
  
def main():  
 print("#############1")  
 goods = [  
 {'title': 'Ковер', 'price': 2000, 'color': 'green'},  
 {'title': 'Диван для отдыха', 'color': 'black'}  
 ]  
 print(list(field(goods, 'title')))  
 print(list(field(goods, 'title', 'price')))  
 print()  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

файл «gen**\_**random.py»

import random  
  
def gen\_random(num\_count, begin, end):  
 for i in range(num\_count):  
 yield random.randint(begin,end)

файл «unique.py»

class Unique:  
 def \_\_init\_\_(self, items, \*\*kwargs):  
 self.items = iter(items)  
 self.seen = set()  
 self.ignore\_case = kwargs.get('ignore\_case', False)  
  
 def \_\_iter\_\_(self):  
 return self  
  
 def \_\_next\_\_(self):  
 for item in self.items:  
 key = item.lower() if self.ignore\_case and isinstance(item, str) else item  
 if key not in self.seen:  
 self.seen.add(key)  
 return item  
  
 raise StopIteration

файл «sort.py»

data = [4, -30, 30, 100, -100, 123, 1, 0, -1, -4]  
print(sorted(data, key = abs, reverse=True))  
print(sorted(data, key= lambda x: abs(x), reverse=True))  
print()

файл «print\_result.py»

def print\_result(func):  
 def wrapper(\*args, \*\*kwargs):  
 result = func(\*args, \*\*kwargs)  
 if isinstance(result, list):  
 for item in result:  
 print(item)  
 elif isinstance(result, dict):  
 for key, v in result.items():  
 print(f"{key} = {v}")  
 else:  
 print(result)  
 return result  
 return wrapper  
  
@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]

файл «cm\_timer.py»

import time  
from contextlib import contextmanager  
  
class cm\_timer\_1:  
 def \_\_enter\_\_(self):  
 self.start\_time = time.time()  
 return(self)  
  
 def \_\_exit\_\_(self, exc\_type, exc\_val, traceback):  
 elapsed\_time = time.time() - self.start\_time  
 print(f"time: {elapsed\_time:.2f}")  
  
@contextmanager  
def cm\_timer\_2():  
 start\_time = time.time()  
 yield  
 elapsed\_time = time.time() - start\_time  
 print(f"time: {elapsed\_time:.2f}")

файл «process\_data.py»

import json  
import sys  
  
from lab\_python\_fp.print\_result import print\_result  
from lab\_python\_fp.cm\_timer import cm\_timer\_1

файл «main.py»

from lab\_python\_fp.field import field  
from lab\_python\_fp.gen\_random import gen\_random  
from lab\_python\_fp.unique import Unique  
from lab\_python\_fp.print\_result import print\_result  
from lab\_python\_fp.cm\_timer import cm\_timer\_1  
import json  
import sys  
  
def main():  
 print("#############1")  
 goods = [  
 {'title': 'Ковер', 'price': 2000, 'color': 'green'},  
 {'title': 'Диван для отдыха', 'color': 'black'}  
 ]  
 print(list(field(goods, 'title')))  
 print(list(field(goods, 'title', 'price')))  
 print()  
  
 print("##########2")  
 print(list(gen\_random(5,1,3)))  
 print()  
  
 print("##########3")  
 data = [1,1,2,3,2]  
 print(list(Unique(data)))  
 print()  
  
 print("##########4")  
 data = [4, -30, 30, 100, -100, 123, 1, 0, -1, -4]  
 print(sorted(data, key = abs, reverse=True))  
 print(sorted(data, key= lambda x: abs(x), reverse=True))  
 print()  
  
 print("##########5")  
  
 @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]  
  
 test\_1()  
 test\_2()  
 test\_3()  
 test\_4()  
 print()  
  
 print("#######6")  
 with cm\_timer\_1():  
 import time  
 time.sleep(1)  
 print()  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

Экранные формы с примерами выполнения программы

