

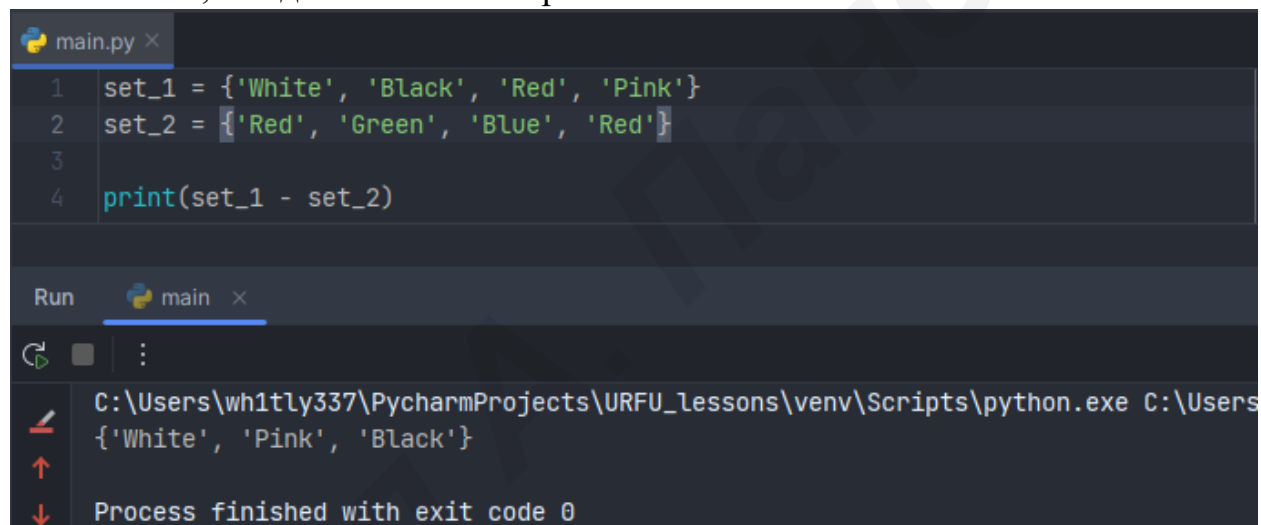
Лабораторная работа 5

ТЕМА 5. Базовые коллекции: множества, списки

Лабораторные задания:

- 1) Друзья предложили вам поиграть в игру “найди отличия и убери повторения (версия для программистов)”. Суть игры состоит в том, что на вход программы поступает два множества, а ваша задача вывести все элементы первого, которых нет во втором. А вы как раз недавно прошли множества и знаете их возможности, поэтому это не составит для вас труда.

P.S. Посмотрите что происходит с повторяющимися значениями в множествах, это достаточно интересно.



```
main.py x
1 set_1 = {'White', 'Black', 'Red', 'Pink'}
2 set_2 = {'Red', 'Green', 'Blue', 'Red'}
3
4 print(set_1 - set_2)

Run main x
C:\Users\wh1tly337\PycharmProjects\URFU_lessons\venv\Scripts\python.exe C:\Users
{'White', 'Pink', 'Black'}
Process finished with exit code 0
```

На скриншоте ниже приведен пример с разными видами повторений в множествах

```
main.py x
1 set_1 = {'White', 'Black', 'Red', 'Pink'}
2 set_2 = {'Red', 'Green', 'Blue', 'Red'}
3 print('1', set_1 - set_2)
4
5 set_1 = {'White', 'Black', 'Red', 'Pink', 'Black', 'White'}
6 set_2 = {'Red', 'Green', 'Blue', 'Red'}
7 print('2', set_1 - set_2)
8
9 set_1 = {'White', 'Black', 'Red', 'Pink', 'Red', 'Red'}
10 set_2 = {'Red', 'Green', 'Red', 'Red', 'Red'}
11 print('3', set_1 - set_2)

Run main x
C:\Users\wh1tly337\PycharmProjects\URFU_lessons\venv\Scripts\python.exe C:\Users
1 {'White', 'Black', 'Pink'}
2 {'White', 'Black', 'Pink'}
3 {'White', 'Black', 'Pink'}
```

- 2) Напишите две одинаковые программы, только одна будет использовать set(), а вторая frozenset() и попробуйте к исходному множеству добавить несколько элементов, например, через цикл.

Вариант с set():

```
main.py x
1 a = set('abcdefg')
2 print(a)
3 for i in range(1, 5):
4     a.add(i)
5 print(a)

Run main x
C:\Users\wh1tly337\PycharmProjects\URFU_lessons\venv\Scripts\python.exe C:\Users
{'c', 'a', 'e', 'b', 'd', 'g', 'f'}
{1, 'c', 2, 3, 4, 'a', 'e', 'b', 'd', 'g', 'f'}
Process finished with exit code 0
```

А вот что произойдет, если вы попытаете добавить новый элемент в frozenset():

```
main.py x
1 a = frozenset('abcdefg')
2 print(a)
3 for i in range(1, 5):
4     a.add(i)
5 print(a)

Run main x
C:\Users\wh1tly337\PycharmProjects\URFU_lessons\venv\Scripts\python.exe C:\Users\wh1tly337\PycharmProjects\URFU_lessons\main.py
Traceback (most recent call last):
  File "C:\Users\wh1tly337\PycharmProjects\URFU_lessons\main.py", line 4, in <module>
    a.add(i)
    ^^^^^
AttributeError: 'frozenset' object has no attribute 'add'
frozenset({'b', 'f', 'd', 'a', 'g', 'c', 'e'})

Process finished with exit code 1
```

- 3) На вход в программу поступает список (минимальной длиной 2 символа). Напишите программу, которая будет менять первый и последний элемент списка.

P.S. В Python есть прикольное свойство, благодаря которому эту задачу можно решить более красиво, используя всего 2 строчки кода, если интересно можете самостоятельно найти это решение.

```
main.py x
1 usage
2 def replace(input_list):
3     memory = input_list[0]
4     input_list[0] = input_list[-1]
5     input_list[-1] = memory
6
7     return input_list
8
9 print(replace([1, 2, 3, 4, 5]))

Run main x
C:\Users\wh1tly337\PycharmProjects\URFU_lessons\venv\Scripts\python.exe C:\Users\wh1tly337\PycharmProjects\URFU_lessons\main.py
[5, 2, 3, 4, 1]

Process finished with exit code 0
```

- 4) На вход в программу поступает список (минимальной длиной 10 символов). Напишите программу, которая выводит элементы с индексами от 2 до 6. В программе необходимо использовать “срез”.

```
main.py x
1 a = [12, 54, 32, 57, 843, 2346, 765, 75, 25, 234, 756, 23]
2 print(a[2:6])

Run main x
C:\Users\wh1tly337\PycharmProjects\URFU_lessons\venv\Scripts\python.exe C:\Users\w
[32, 57, 843, 2346]
Process finished with exit code 0
```

- 5) Иван задумался о поиске «бесполезного» числа, полученного из списка. Суть поиска в следующем: он берет произвольный список чисел, находит самое большое из них, а затем делит его на длину списка. Студент пока не придумал, где может пригодиться подобное значение, но ищет у вас помощи в реализации такой функции `useless()`.

```
main.py x
3 usages
1 def useless(lst):
2     return max(lst) / len(lst)
3
4
5 print(useless([3, 5, 7, 3, 33]))
6 print(useless([-12.5, 54, 77.3, 0, -36, 98.2, -63, 21.7, 47, -89.6]))
7 print(useless([-25.8, 86, 12.5, -56, 73.2, 0, 43, -91.5, 65.9, -7]))

Run main x
C:\Users\wh1tly337\PycharmProjects\URFU_lessons\venv\Scripts\python.exe C:\Users\w
6.6
9.82
8.6
Process finished with exit code 0
```

- 6) Ребята не могут определиться каким супергероем они хотят стать. У них есть случайно составленный список супергероев, и вы должны определить кто из ребят будет каким супергероем. Необходимо использовать разделение списков.

```
main.py x
1  superheroes = ['superman', 'spiderman', 'batman']
2
3  nikolay, vasiluy, ivan = superheroes
4
5  print('Николай - ', nikolay)
6  print('Василий - ', vasiluy)
7  print('Иван - ', ivan)

Run  main x
C:\Users\wh1tly337\PycharmProjects\URFU_lessons\venv\Scripts\python.exe C:\Users\wh1tly337\PycharmProjects\URFU_lessons\main.py
Николай - superman
Василий - spiderman
Иван - batman
Process finished with exit code 0
```

- 7) Вовочка, насмотревшись передачи “Слабое звено” решил написать программу, которая также будет находить самое слабое звено (минимальный элемент) и удалять его, только делать он это хочет не с людьми, а со списком. Помогите Вовочке с реализацией программы. Подсказка: для этого вам необходимо отсортировать список и удалить значение при помощи `pop()`.

```
main.py x
1  a = [-25.8, 86, 12.5, -56, 73.2, 0, 43, -91.5, 65.9, -7]
2  a.sort()
3  print('Отсортированный список:\n', a)
4  a.pop(0)
5  print('Отсортированный список без наименьшего элемента:\n', a)

Run  main x
C:\Users\wh1tly337\PycharmProjects\URFU_lessons\venv\Scripts\python.exe C:\Users\wh1tly337\PycharmProjects\URFU_lessons\main.py
Отсортированный список:
[-91.5, -56, -25.8, -7, 0, 12.5, 43, 65.9, 73.2, 86]
Отсортированный список без наименьшего элемента:
[-56, -25.8, -7, 0, 12.5, 43, 65.9, 73.2, 86]
Process finished with exit code 0
```

- 8) Михаил решил создать большой n-мерный список, для этого он случайным образом создал несколько списков, состоящих минимум из 3, а максимум из 10 элементов и поместил их в один большой список. Он также как и Иван не знает зачем ему это сейчас нужно, но надеется на то, что это пригодится ему в будущем.

```
main.py ×
1  from random import randint
2
3
4  1 usage
5  def list_maker():
6      a = [randint(1, 100)] * randint(3, 10)
7      return a
8
9  if __name__ == '__main__':
10     result = []
11     for i in range(randint(1, 5)):
12         result.append(list_maker())
13
14     print(result)
```

```
Run  main ×
C:\Users\wh1tly337\PycharmProjects\URFU_lessons\venv\Scripts\python.exe C:\Users\w
[[36, 36, 36, 36, 36, 36, 36], [41, 41, 41, 41, 41, 41, 41]]
Process finished with exit code 0
```

9) Вы работаете в ресторане и отвечает за статистику покупок, ваша задача сравнить между собой заказы покупателей, которые указаны в разном порядке. Реализуйте функцию `superset()`, которая принимает 2 множества. Результат работы функции: вывод в консоль одного из сообщений в зависимости от ситуации:

- 1 - «Супермножество не обнаружено»
- 2 – «Объект {X} является чистым супермножеством»
- 3 – «Множества равны»

```
main.py x
4 usages
1 def superset(set_1, set_2):
2     if set_1 > set_2:
3         print(f'Объект {set_1} является чистым супермножеством')
4     elif set_1 == set_2:
5         print(f'Множества равны')
6     elif set_1 < set_2:
7         print(f'Объект {set_2} является чистым супермножеством')
8     else:
9         print('Супермножество не обнаружено')
10
11
12 if __name__ == '__main__':
13     superset({1, 8, 3, 5}, {3, 5})
14     superset({1, 8, 3, 5}, {5, 3, 8, 1})
15     superset({3, 5}, {5, 3, 8, 1})
16     superset({90, 100}, {3, 5})

Run main x
/usr/local/bin/python3.11 /Users/user/PycharmProjects/URFU_lessons/main.py
Объект {8, 1, 3, 5} является чистым супермножеством
Множества равны
Объект {8, 1, 3, 5} является чистым супермножеством
Супермножество не обнаружено
Process finished with exit code 0
```

- 10) Предположим, что вам нужно разобрать стопку бумаг, но нужно начать работу с нижней, “переверните стопку”. Вам дан произвольный список. Представьте его в обратном порядке. Программа должна занимать не более двух строк в редакторе кода.

```
main.py x
1 my_list = [2, 5, 8, 3]
2 print(my_list[::-1])

Run main x
/usr/local/bin/python3.11 /Users/user/PycharmProjects/URFU_lessons/main.py
[3, 8, 5, 2]
Process finished with exit code 0
```