

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №7
дисциплины «Программирование на Python»

Выполнил:
Данилецкий Дмитрий Витальевич
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств
вычислительной техники и
автоматизированных систем», очная
форма обучения

(подпись)

Руководитель практики:
Воронкин Р. А., канд. технических
наук, доцент кафедры
инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

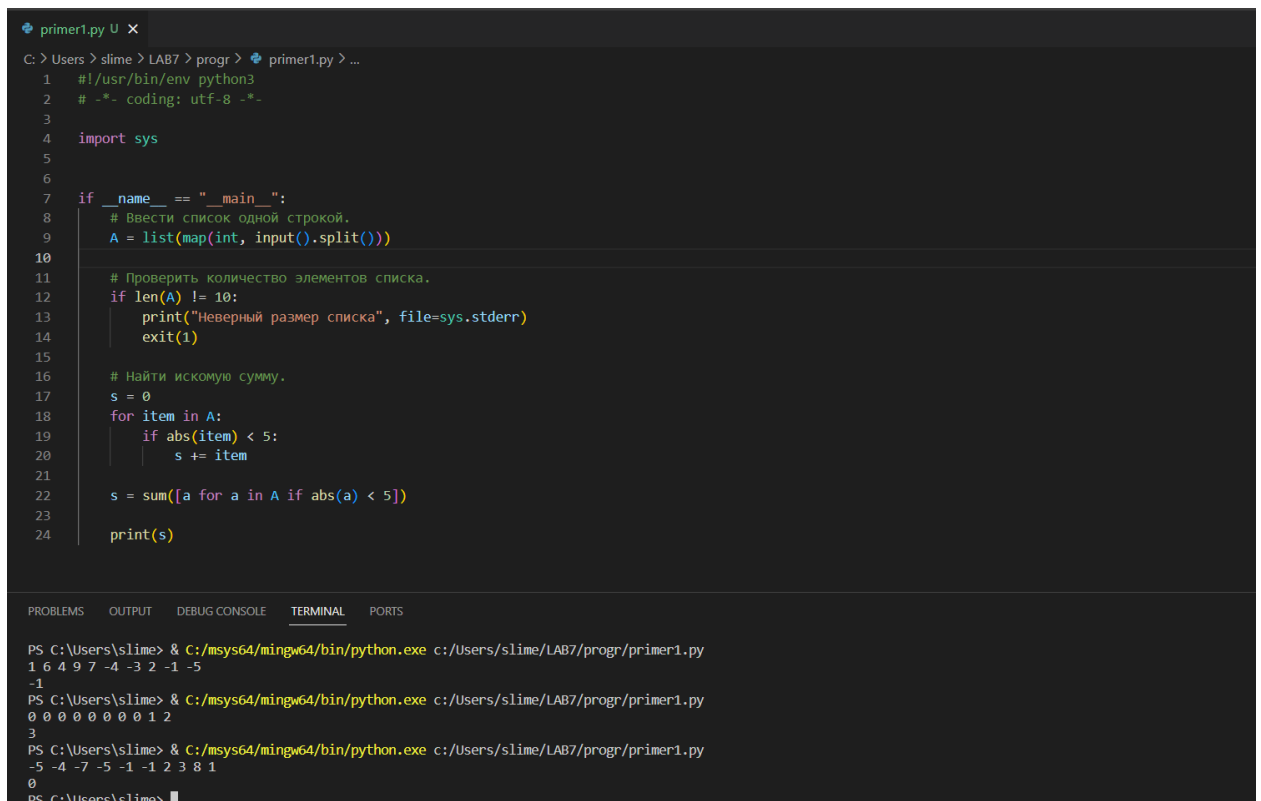
Ставрополь, 2023 г.

Тема: Работа со списками в языке Python

Цель: приобретение навыков по работе со списками при написании программ с помощью языка программирования Python версии 3.x.

Ход работы

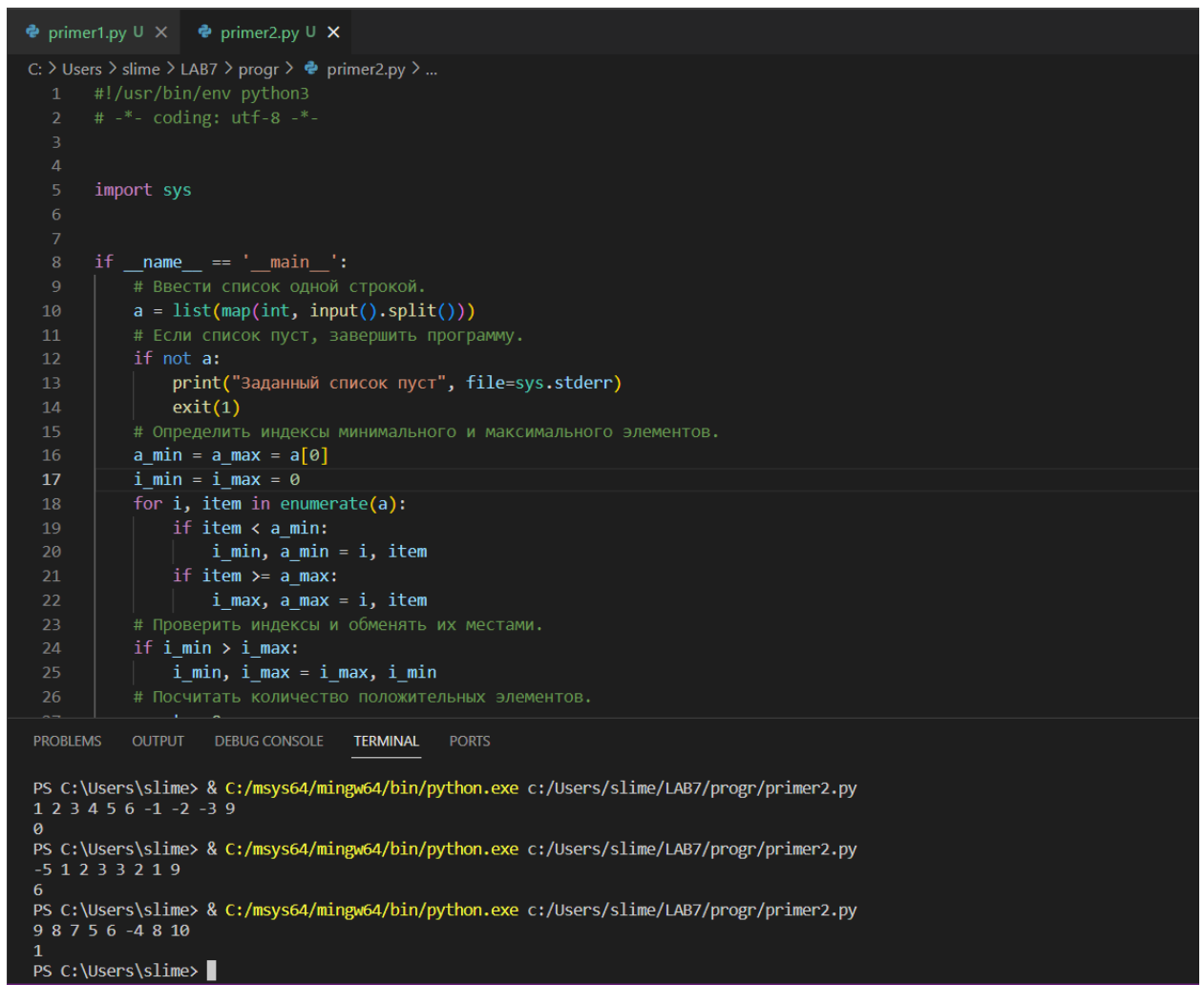
1. Создал общедоступный репозиторий на GitHub, в котором использована лицензия MIT и язык программирования Python. Выполнил клонирование созданного репозитория.
2. Дополнил файл .gitignore необходимыми правилами.
3. Организовал созданный репозиторий в соответствие с моделью ветвления git-flow.
4. Проработал примеры лабораторной работы. Создал для каждого примера отдельный модуль языка Python. Привел в отчете скриншоты результатов выполнения каждой из программ примеров при различных исходных данных вводимых с клавиатуры.



```
primer1.py U X
C: > Users > slime > LAB7 > progr > primer1.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6
7  if __name__ == "__main__":
8      # Ввести список одной строкой.
9      A = list(map(int, input().split()))
10
11     # Проверить количество элементов списка.
12     if len(A) != 10:
13         print("Неверный размер списка", file=sys.stderr)
14         exit(1)
15
16     # Найти искомую сумму.
17     s = 0
18     for item in A:
19         if abs(item) < 5:
20             s += item
21
22     s = sum([a for a in A if abs(a) < 5])
23
24     print(s)

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS C:\Users\slime> & C:/msys64/mingw64/bin/python.exe c:/Users/slme/LAB7/progr/primer1.py
1 6 4 9 7 -4 -3 2 -1 -5
-1
PS C:\Users\slime> & C:/msys64/mingw64/bin/python.exe c:/Users/slme/LAB7/progr/primer1.py
0 0 0 0 0 0 0 1 2
3
PS C:\Users\slime> & C:/msys64/mingw64/bin/python.exe c:/Users/slme/LAB7/progr/primer1.py
-5 -4 -7 -5 -1 -1 2 3 8 1
0
PS C:\Users\slime>
```

Рисунок 1. Результат работы программы из примера 1



```
primer1.py U x primer2.py U x
C: > Users > slime > LAB7 > progr > primer2.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  import sys
6
7
8  if __name__ == '__main__':
9      # Ввести список одной строкой.
10     a = list(map(int, input().split()))
11     # Если список пуст, завершить программу.
12     if not a:
13         print("Заданный список пуст", file=sys.stderr)
14         exit(1)
15     # Определить индексы минимального и максимального элементов.
16     a_min = a_max = a[0]
17     i_min = i_max = 0
18     for i, item in enumerate(a):
19         if item < a_min:
20             i_min, a_min = i, item
21         if item >= a_max:
22             i_max, a_max = i, item
23     # Проверить индексы и обменять их местами.
24     if i_min > i_max:
25         i_min, i_max = i_max, i_min
26     # Посчитать количество положительных элементов.
27
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\slime> & C:/msys64/mingw64/bin/python.exe c:/Users/slme/LAB7/progr/primer2.py
1 2 3 4 5 6 -1 -2 -3 9
0
PS C:\Users\slime> & C:/msys64/mingw64/bin/python.exe c:/Users/slme/LAB7/progr/primer2.py
-5 1 2 3 3 2 1 9
6
PS C:\Users\slime> & C:/msys64/mingw64/bin/python.exe c:/Users/slme/LAB7/progr/primer2.py
9 8 7 5 6 -4 8 10
1
PS C:\Users\slime>
```

Рисунок 2. Результат работы программы из примера 2

6. Выполнил индивидуальные задания, согласно варианту 8. Привёл в отчете скриншоты работы программ.

Задание 1. В заданном списке подсчитать число нулевых элементов и вывести на экран их индексы.

```
C: > Users > slime > LAB7 > progr > Ind1.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6
7  if __name__ == '__main__':
8
9      a = list(map(int, input().split()))
10     if not a:
11         print("Заданный список пуст", file=sys.stderr)
12         exit(1)
13
14     count_zeros = 0
15
16     for index, value in enumerate(a):
17         if value == 0:
18             count_zeros += 1
19             print("Нулевой элемент найден на индексе:", index)
20
21     print("Общее количество нулевых элементов:", count_zeros)
22
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\slime> & C:/msys64/mingw64/bin/python.exe c:/Users/slime/LAB7/progr/Ind1.py
0 1 2 0 3 1 0 0 0 4 7 9 0
Нулевой элемент найден на индексе: 0
Нулевой элемент найден на индексе: 3
Нулевой элемент найден на индексе: 6
Нулевой элемент найден на индексе: 7
Нулевой элемент найден на индексе: 8
Нулевой элемент найден на индексе: 12
Общее количество нулевых элементов: 6
PS C:\Users\slime>
```

Рисунок 3. Результат работы программы из 1 индивидуального задания

Задание 2. В списке, состоящем из вещественных элементов, вычислить:

- 1) максимальный по модулю элемент списка;
- 2) сумму элементов списка, расположенных между первым и вторым положительными элементами.

```
C: > Users > slime > LAB7 > progr > Ind2.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6
7  if __name__ == '__main__':
8
9      a = list(map(float, input().split()))
10     if not a:
11         print("Заданный список пуст", file=sys.stderr)
12         exit(1)
13
14     max_el = max(a, key=abs)
15     print("Максимальный по модулю элемент:", max_el)
16
17     positivs = [i for i, x in enumerate(a) if x > 0]
18
19     if len(positivs) >= 2:
20         start_in = positivs[0]
21         end_in = positivs[1]
22         sum_between = sum(value for idx, value in enumerate(
23             a[start_in + 1:end_in], start=start_in + 1)
24         )
25         print("Сумма элементов:", sum_between)
26     else:
27         print("Недостаточно положительных элементов в списке.")
28
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\slime> & C:/msys64/mingw64/bin/python.exe c:/Users/slime/LAB7/progr/Ind2.py
1.2 -9.7 -3.3 5 4 9 11 -4
Максимальный по модулю элемент: 11.0
Сумма элементов: -13.0
PS C:\Users\slime>
```

Рисунок 5. Результат работы программы из 2 индивидуального задания

Контрольные вопросы:

1. Что такое списки в языке Python?

Список – это структура данных для хранения объектов различных типов. Размер списка не статичен, его можно изменять. Список по своей природе является изменяемым типом данных. Переменная, определяемая как список, содержит ссылку на структуру в памяти, которая в свою очередь хранит ссылки на какие-либо другие объекты или структуры.

2. Как осуществляется создание списка в Python?

Для создания списка нужно заключить элементы в квадратные скобки: `list_1 = [1, 2, 3, 4,]`. Так же при помощи `list()`: `list_2 = list (1, 2, 3, 4)`

3. Как организовано хранение списков в оперативной памяти? При его создании в памяти резервируется область («контейнер»), в котором хранятся ссылки на другие элементы в памяти. Содержимое контейнера можно менять в отличие от чисел или строк.

4. Каким образом можно перебрать все элементы списка?

Все элементы списка можно перебрать разными способами, при помощи цикла `for` и с помощью `range(len(list))`: `for i in range(len(list)): ...` или при помощи также цикла `for` и с помощью `enumerate(a)`: `for i, item in enumerate(a):`

5. Какие существуют арифметические операции со списками?

Объединение списков при помощи оператора сложения («+») и операция повторения при помощи оператора умножения («*»).

6. Как проверить есть ли элемент в списке?

Проверить есть ли заданный элемент в списке может оператор `in`, если нет заданного элемента `not in`

7. Как определить число вхождений заданного элемента в списке?

Число вхождений заданного элемента в списке может определить метод `count`, который в качестве аргумента принимает искомый элемент.

8. Как осуществляется добавление (вставка) элемента в список?

Добавление элемента в список может осуществляется при помощи метода `append(a)`, который добавляет элемент `a` в конец списка, также можно добавить больше одного элемента методом `extend(a)`.

9. Как выполнить сортировку списка?

Отсортировать массив можно при помощи метода `sort()`, чтобы отсортировать по «убыванию», `sort(reverse=True)`.

10. Как удалить один или несколько элементов из списка?

Удалить элемент по его индексу может метод `pop(i)`.

Удалить элемент по его значению может метод `remove()`.

Оператор `del` может удалять также как метод `remove`, но сразу несколько элементов: `del list[1:3]`.

11. Что такое списковое включение и как с его помощью осуществлять обработку списков?

Списковое включение – способ построения списков, пример: `a = [i for i in range(10)]`. В этом примере создастся массив из 10 элементов: от 0 до 9. Также при помощи данного способа можно осуществлять обработку списков: с элементом `i` проводить арифметические операции, и после цикла писать условие вхождения в список.

12. Как осуществляется доступ к элементам списков с помощью срезов?

С помощью срезов доступ к элементам списков осуществляется так:

- 1) `list[:]` – копия списка;
- 2) `list[0:n]` – первые `n` элементы;
- 3) `list[n:m]` – получить элементы с `n+1` по `m`;
- 4) `list[:n]` – взять элементы списка с шагом `n`;
- 5) `list[n:m:s]` – взять элементы с `n+1` по `m` с шагом `s`.

13. Какие существуют функции агрегации для работы со списками?

Функции агрегации: получить число элементов: `len()`, получить минимальный элемент списка: `min()`, получить максимальный элемент списка: `max()`, получить сумму элементов списка: `sum()`.

14. Как создать копию списка?

Создать копию списка можно при помощи среза `a = b[:]` и при помощи метода `copy()`.

15. Самостоятельно изучите функцию `sorted` языка Python. В чем ее отличие от метода `sort` списков?

Функция `sorted()` возвращает новый отсортированный список, оставляя исходный список неизменным. Она принимает список (или другую итерируемую последовательность) в качестве аргумента и возвращает новый список, содержащий отсортированные элементы. Основное отличие между `sorted()` и `sort()` заключается в том, что `sorted()` возвращает новый отсортированный список, оставляя исходный список неизменным, в то время как `sort()` изменяет сам список, сортируя его элементы на месте

Вывод: в ходе выполнения лабораторной работы были приобретены навыки по работе со списками при написании программ с помощью языка программирования Python версии 3.x.