

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

**Институт перспективной инженерии
Департамент цифровых, робототехнических систем и электроники**

**ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №4
дисциплины «Программирование на Python»**

Выполнила:
Федорова Дарья Юрьевна
2 курс, группа ИВТ-б-о-24-1,
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.

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2025 г.

Вариант 25

Ссылка на репозиторий: https://github.com/Vishhh123/Rrepozitoriy_4#

Тема: Работа со списками и кортежами в языке Python.

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

Ход работы:

Создайте новый репозиторий

Репозитории содержат файлы проекта и историю версий. У вас есть проект в другом месте? [Импортируйте репозиторий](#).

Обязательные поля отмечены звёздочкой (*).

1 Общая информация

Владелец * / Название репозитория *

Vishhh123 / Rrepozitoriy_4

✓ Rrepozitoriy_4 доступен.

Хорошие названия репозитория должны быть короткими и запоминающимися. Как насчёт [осторожного-изобретателя](#)?

Описание

0 / 350 символов

2 Конфигурация

Выберите видимость *

Выберите, кто сможет просматривать и обновлять этот репозиторий

Публичный

Добавить README

README можно использовать для более подробных описаний. [О README](#)

Вкл

Добавить .gitignore

Файл .gitignore сообщает Git, какие файлы не нужно отслеживать. [Об игнорировании файлов](#)

Питон

Добавить лицензию

Лицензии объясняют, как другие могут использовать ваш код. [О лицензиях](#)

MIT License

Рис. 1 - создание общедоступного репозитория с лицензией MIT, языком программирования Python.

```
C:\Users\user>git clone https://github.com/Vishhh123/Rrepozitoriy_4.git
Cloning into 'Rrepozitoriy_4'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (5/5), done.

C:\Users\user>cd Rrepozitoriy_4

C:\Users\user\Rrepozitoriy_4>git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
```

Рис. 2 - клонирование репозитория на устройство, переход к репозиторию, его проверка.

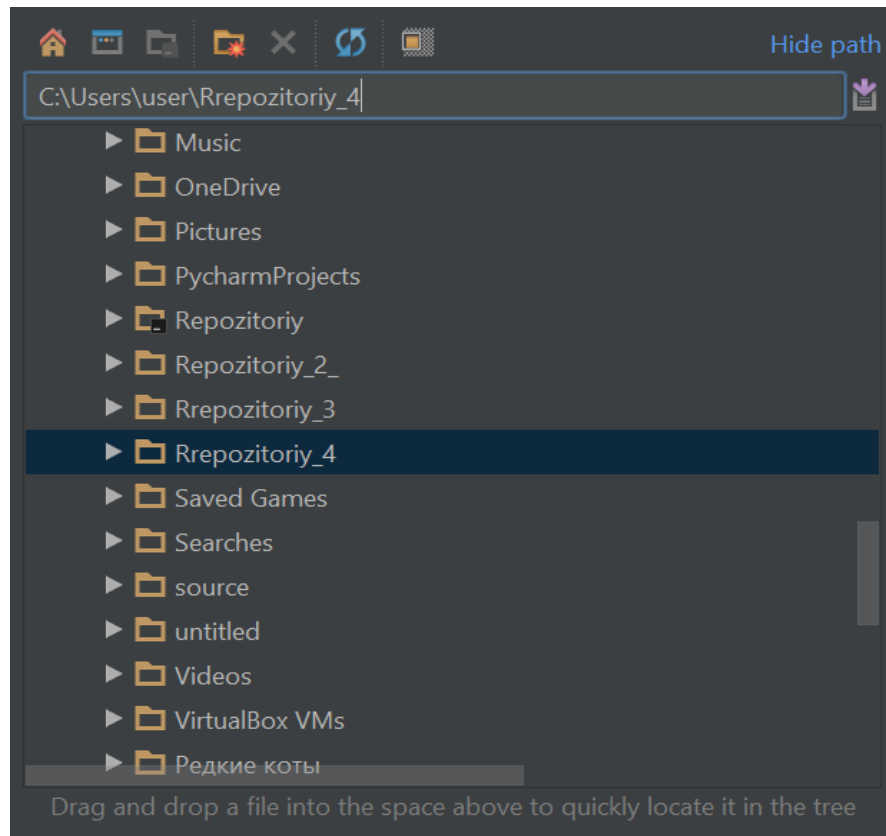


Рис. 3 - создание в PyCharm проекта из существующих файлов (file -> open -> путь к файлу)

```
primer_1.1.py x
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6  if __name__ == '__main__':
7      A = list(map(int, input().split()))
8
9      if len(A) != 10:
10         print("Неверный размер списка", file=sys.stderr)
11         exit(1)
12
13         s = 0
14         for item in A:
15             if abs(item) < 5:
16                 s += item
17
18         print(s)
```

Run: primer_1.1 primer_1.1

"C:\Program Files\Python313\python.exe" C:/Users/user/Rrepozitoriy_4/primer_1.1.py
1 3 2 4 5 6 4 3 8 10
17

Рис. 4 – результат выполнения примера 1.1.

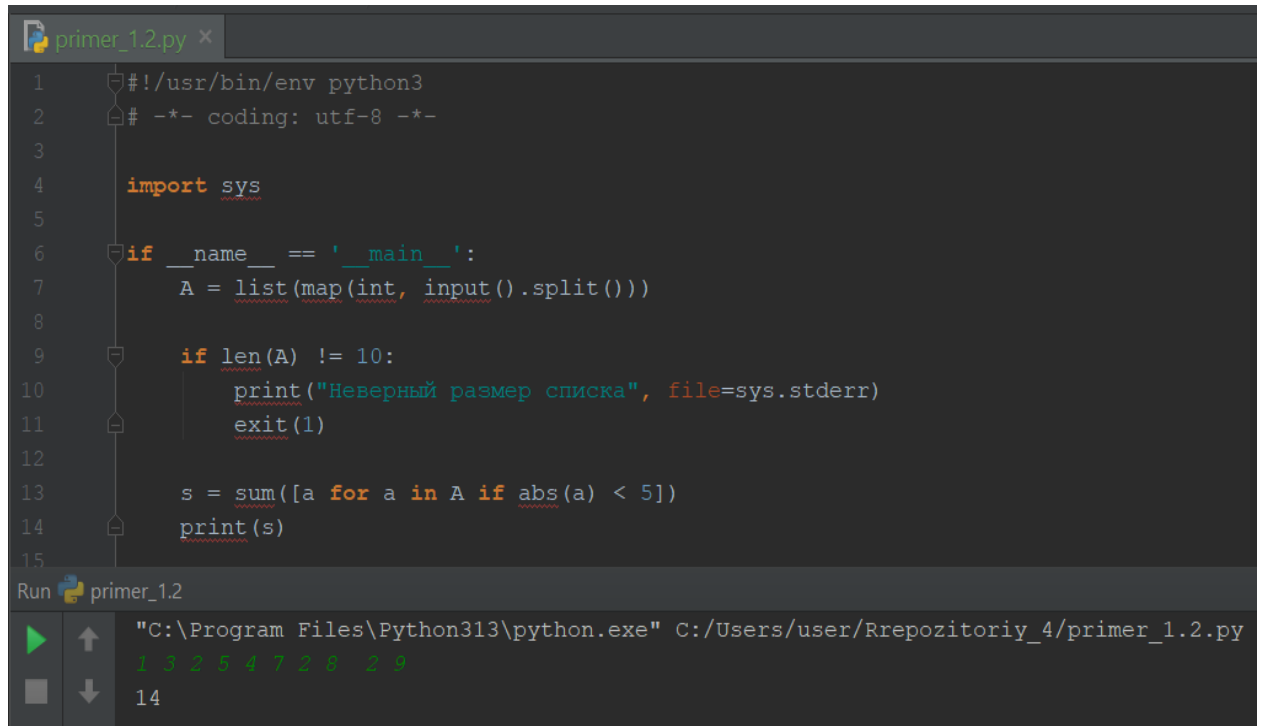
```

C:\Users\user\Rrepozitoriy_4>git add .

C:\Users\user\Rrepozitoriy_4>git commit -m "Код к примеру 1.1"
[main f3678dc] Код к примеру 1.1
1 file changed, 26 insertions(+)
create mode 100644 "\320\237\321\200\320\270\320\274\320\265\321\200\321\213\primer_1.1.py"

```

Рис. 5 - сохранение, добавление коммита.



```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6  if __name__ == '__main__':
7      A = list(map(int, input().split()))
8
9      if len(A) != 10:
10         print("Неверный размер списка", file=sys.stderr)
11         exit(1)
12
13         s = sum([a for a in A if abs(a) < 5])
14         print(s)
15
Run primer_1.2
"C:\Program Files\Python313\python.exe" C:/Users/user/Rrepozitoriy_4/primer_1.2.py
1 3 2 5 4 7 2 8 2 9
14

```

Рис. 6 - результат выполнения примера 1.2

```

C:\Users\user\Rrepozitoriy_4>git add .

C:\Users\user\Rrepozitoriy_4>git commit -m "Код к примеру 1.2"
[main 9a463c8] Код к примеру 1.2
1 file changed, 24 insertions(+)
create mode 100644 "\320\237\321\200\320\270\320\274\320\265\321\200\321\213\primer_1.2.py"

```

Рис. 7 - сохранение, добавление коммита.

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 import sys
5
6 if __name__ == '__main__':
7     a = list(map(int, input().split()))
8
9     if not a:
10        print("Заданный список пуст", file=sys.stderr)
11        exit(1)
12
13    a_min = a_max = a[0]
14    i_min = i_max = 0
15
16    for i, item in enumerate(a):
17        if item < a_min:
18            i_min, a_min = i, item
19
20        if item >= a_max:
21            i_max, a_max = i, item
22
23    if i_min > i_max:
24        i_min, i_max = i_max, i_min
25
26    count = 0
27    for item in a[i_min + 1:i_max]:
28        if item > 0:
29            count += 1
30
31    print(count)
```

Run: primer_2 (1) primer_2.2

"C:\Program Files\Python313\python.exe" C:/Users/user/Rrepozitoriy_4/Примеры/primer_2.2.py

1 2 3 4 5 6 7

1

Рис. 8 - результат выполнения примера 2.

```
C:\Users\user\Rrepozitoriy_4>git add .
warning: in the working copy of '.idea/workspace.xml', LF will be replaced by CRLF the next time Git touches it
C:\Users\user\Rrepozitoriy_4>git commit -m "Код к примеру 2"
[main 2f7e5ba] Код к примеру 2
2 files changed, 39 insertions(+), 3 deletions(-)
create mode 100644 "\320\237\321\200\320\270\320\274\320\265\321\200\321\213/primer_2.py"
```

Рис. 9 - сохранение, добавление коммита.

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 A = ("abc", "abcd", "bcd", "cde")
5
6 for item in A:
7     print(item)
8
9 A = (-1, 3, -8, 12, -20)
10
11 i = 0
12 k = 0
13
14 while i < len(A):
15     if A[i] < 0:
16         k = k + 1
17     i = i + 1
18
19 print("k =", k)
20
21 A = ("abc", "ad", "bcd")
22
23 B = [item * 2 for item in A]
24
25 print("A =", A)
26 print("B =", B)
```

Run: primer_2 (1) primer_3

"C:\Program Files\Python313\python.exe" C:/Users/user/Rrepozitoriy_4/Примеры/primer_3.py

abc
abcd
bcd
cde
k = 3
A = ('abc', 'ad', 'bcd')
B = ['abcabc', 'adad', 'bcdbcd']

Рис. 10 - результат выполнения примера 3.

```
C:\Users\user\Rrepozitoriy_4>git add .
C:\Users\user\Rrepozitoriy_4>git commit -m "Код к примеру 3"
[main eeb4b1d] Код к примеру 3
1 file changed, 29 insertions(+)
create mode 100644 "\320\237\321\200\320\270\320\274\320\265\321\200\321\213\primer_3.py"
```

Рис. 11 - сохранение, добавление коммита.

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

A = ("Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat")

day = str(input("Enter day: "))

if day in A:
    num = A.index(day)
    print("Number of day =", num + 1)
else:
    num = -1
    print("Wrong day.")

primer_2 (1)  primer_4
"C:\Program Files\Python313\python.exe" C:/Users/user/Rrepozitoriy_4/Примеры/primer_4.py
Enter day: Sat
Number of day = 7
```

Рис. 12 - результат выполнения примера 4.

```
C:\Users\user\Rrepozitoriy_4>git add .
C:\Users\user\Rrepozitoriy_4>git commit -m "Код к примеру 4"
[main f34b5e3] Код к примеру 4
1 file changed, 14 insertions(+)
create mode 100644 "\320\237\321\200\320\270\320\274\320\265\321\200\321\213\primer_4.py"
```

Рис. 13 - сохранение, добавление коммита.

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import sys

if __name__ == '__main__':
    A = tuple(map(int, input().split()))
    if len(A) != 10:
        print("Неверный размер кортежа", file=sys.stderr)
        exit(1)
    s = 0
    for item in A:
        if abs(item) < 5:
            s += item
    print(s)

primer_2 (1)  primer_5
"C:\Program Files\Python313\python.exe" C:/Users/user/Rrepozitoriy_4/Примеры/primer_5.py
1 2 3 4 5 6 7 8 9 10
10
```

Рис. 14 - результат выполнения примера 5.

```

C:\Users\user\Rrepozitoriy_4>git add .
C:\Users\user\Rrepozitoriy_4>git commit -m "Код к примеру 5"
[main 145a4ba] Код к примеру 5
1 file changed, 15 insertions(+)
create mode 100644 "320\237\321\200\320\270\320\274\320\265\321\200\321\213\primer_5.py"

```

Рис. 15 - сохранение, добавление коммита.

```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6  if __name__ == '__main__':
7      A = list(map(int, input().split()))
8      if len(A) != 10:
9          print("Неверный размер списка", file=sys.stderr)
10         exit(1)
11         s = sum(a for a in A if abs(a) < 5)
12         print(s)

```

Run: primer_2 (1) primer_6

"C:\Program Files\Python313\python.exe" C:/Users/user/Rrepozitoriy_4/Примеры/primer_6.py

40

Рис. 16 - результат выполнения примера 6.

```

C:\Users\user\Rrepozitoriy_4>git add .
C:\Users\user\Rrepozitoriy_4>git commit -m "Код к примеру 6"
[main bc2739e] Код к примеру 6
1 file changed, 12 insertions(+)
create mode 100644 "320\237\321\200\320\270\320\274\320\265\321\200\321\213\primer_6.py"

```

Рис. 17 - сохранение, добавление коммита.

25. В ЭВМ по очереди поступают результаты соревнований по плаванию на дистанции 200 м, в которых участвует n спортсменов ($n > 10$). Выдать на экран дисплея лучший результат.

Рис. 18 - индивидуальное задание 1.

```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6  if __name__ == '__main__':
7      print("Введите больше 10 результатов спортсменов через пробел:")
8      result = list(map(int, input().split()))
9
10     if len(result) <= 10:
11         print("Кол-во спортсменов должно быть > 10!", file=sys.stderr)
12         exit(1)
13
14     luchshij_result = min(result)
15
16     print(f"Лучший результат на дистанции 200 м: {luchshij_result} секунд")

```

Run: primer_2 (1) zadanie_1 zadanie_1

"C:\Program Files\Python313\python.exe" C:/Users/user/Rrepozitoriy_4/Задания/zadanie_1.py

Введите больше 10 результатов спортсменов через пробел:

49393 3404 345634 6345345 3456345 2543 63454 34563 64352 2354634 5645

Лучший результат на дистанции 200 м: 2543 секунд

Рис. 19 - результат выполнения индивидуального задания 1.

```
C:\Users\user\Repozitoriy_4>git add .
warning: in the working copy of '.idea/workspace.xml', LF will be replaced by CRLF the next time Git touches it
C:\Users\user\Repozitoriy_4>git commit -m "Код к индивидуальному заданию 1"
[main d549121] Код к индивидуальному заданию 1
2 files changed, 66 insertions(+), 27 deletions(-)
create mode 100644 "\320\227\320\260\320\264\320\260\320\275\320\270\321\217\zadanie_1.py"
```

Рис. 20 - сохранение, добавление коммита.

6. В списке, состоящем из целых элементов, вычислить:

1. номер максимального элемента списка;
2. произведение элементов списка, расположенных между первым и вторым нулевыми элементами.

Преобразовать список таким образом, чтобы в первой его половине располагались элементы, стоявшие в нечетных позициях, а во второй половине - элементы, стоявшие в четных позициях.

Рис. 21 - индивидуальное задание 2.

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import sys

if __name__ == '__main__':
    print("Введите целые числа через пробел:")
    a = list(map(int, input().split()))

    if not a:
        print("Список пуст!", file=sys.stderr)
        exit(1)

    print(f"\nИсходный список: {a}")

    #Номер максимального элемента
    max_index = a.index(max(a))
    print(f"Номер максимального элемента: {max_index} (значение {a[max_index]})")

    #Произведение между нулевыми элементами
    if a.count(0) >= 2:
        first_zero = a.index(0)
        second_zero = a.index(0, first_zero + 1)

        product = 1
        for q in a[first_zero + 1:second_zero]:
            product = q * product

        print(f"Произведение между нулевыми элементами = {product}")
    else:
        print("Недостаточно нулевых элементов")

    #Преобразование списка
    pervaya_polovina = [a[i] for i in range(len(a)) if i % 2 == 1]
    vtoraya_polovina = [a[i] for i in range(len(a)) if i % 2 == 0]
    spisok = pervaya_polovina + vtoraya_polovina

    print(f"Преобразованный список: {spisok}")
```

primer_2 (1) zadanie_1 zadanie_2

Исходный список: [1, 2, 3, 4, 0, 4, 7, 3, 2, 6, 7, 0, 1]
 Номер максимального элемента: 6 (значение 7)
 Произведение между нулевыми элементами = 7056
 Преобразованный список: [2, 4, 4, 3, 6, 0, 1, 3, 0, 7, 2, 7, 1]

Рис. 22 - результат выполнения индивидуального задания 2.

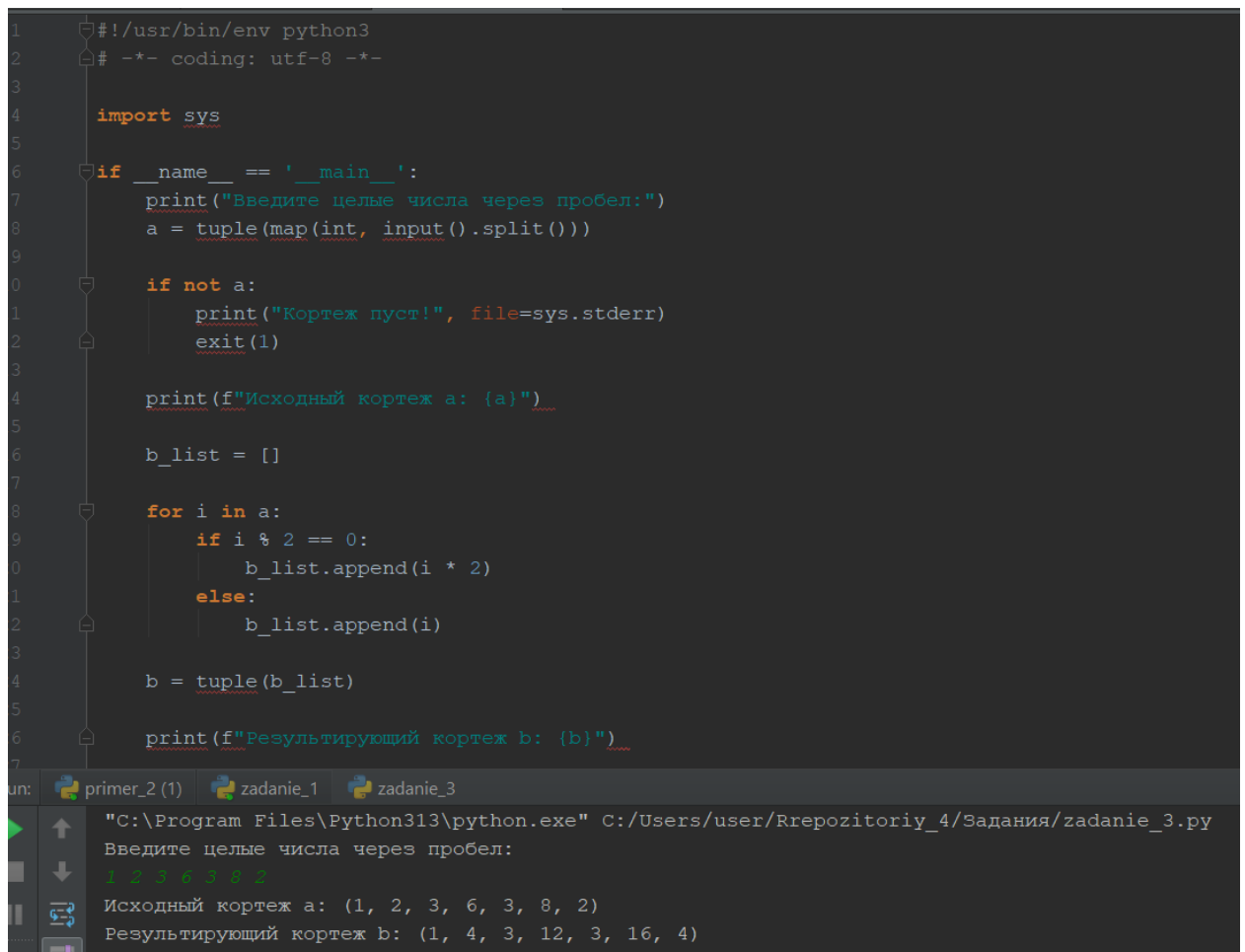

```
C:\Users\user\Rrepozitoriy_4>git add .
warning: in the working copy of '.idea/workspace.xml', LF will be replaced by CRLF the next time Git touches it

C:\Users\user\Rrepozitoriy_4>git commit -m "Код к индивидуальному заданию 2"
[main d59472a] Код к индивидуальному заданию 2
2 files changed, 109 insertions(+), 35 deletions(-)
create mode 100644 "\320\227\320\260\320\264\320\260\320\275\320\270\321\217\zadanie_2.py"
```

Рис. 23 - сохранение, добавление коммита.

25. Из элементов кортежа a , заполненного целыми числами, сформировать кортеж b того же размера по правилу: четные элементы кортежа a удвоить, нечетные оставить без изменения.

Рис. 24 - индивидуальное задание 3.



```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6  if __name__ == '__main__':
7      print("Введите целые числа через пробел:")
8      a = tuple(map(int, input().split()))
9
10     if not a:
11         print("Кортеж пуст!", file=sys.stderr)
12         exit(1)
13
14     print(f"Исходный кортеж a: {a}")
15
16     b_list = []
17
18     for i in a:
19         if i % 2 == 0:
20             b_list.append(i * 2)
21         else:
22             b_list.append(i)
23
24     b = tuple(b_list)
25
26     print(f"Результирующий кортеж b: {b}")
```

un: primer_2 (1) zadanie_1 zadanie_3

"C:\Program Files\Python313\python.exe" C:/Users/user/Rrepozitoriy_4/Задания/zadanie_3.py

Введите целые числа через пробел:

1 2 3 6 3 8 2

Исходный кортеж a: (1, 2, 3, 6, 3, 8, 2)

Результирующий кортеж b: (1, 4, 3, 12, 3, 16, 4)

Рис. 25 - результат выполнения индивидуального задания 3.

```
C:\Users\user\Rrepozitoriy_4>git add .
warning: in the working copy of '.idea/workspace.xml', LF will be replaced by CRLF the next time Git touches it

C:\Users\user\Rrepozitoriy_4>git commit -m "Код к индивидуальному заданию 3"
[main ba98893] Код к индивидуальному заданию 3
3 files changed, 81 insertions(+), 45 deletions(-)
create mode 100644 "\320\227\320\260\320\264\320\260\320\275\320\270\321\217\zadanie_3.py"

C:\Users\user\Rrepozitoriy_4>git push
Enumerating objects: 22, done.
Counting objects: 100% (22/22), done.
Delta compression using up to 24 threads
Compressing objects: 100% (18/18), done.
Writing objects: 100% (19/19), 4.18 KiB | 1.39 MiB/s, done.
Total 19 (delta 10), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (10/10), completed with 3 local objects.
To https://github.com/Vishhh123/Rrepozitoriy_4.git
74a9979..ba98893 main -> main
```

Рис. 26 - сохранение, добавление коммита, отправка на github.

Ответы на контрольные вопросы:

1. Что такое списки в языке Python?

Коллекция упорядоченных элементов, изменяемая (мутабельная), может содержать разные типы данных.

2. Как осуществляется создание списка в Python?

`a = [1, 2, 3]`, `b = list(range(5))`, `c = []` (пустой список).

3. Как организовано хранение списков в оперативной памяти?

Хранятся как динамические массивы ссылок на объекты, память перераспределяется при необходимости.

4. Каким образом можно перебрать все элементы списка?

`for item in lst:` или `for i in range(len(lst)):`.

5. Какие существуют арифметические операции со списками?

Конкатенация (+), повторение (*), сравнение (==, <, >).

6. Как проверить есть ли элемент в списке?

`if element in lst:`.

7. Как определить число вхождений заданного элемента в списке?

`lst.count(element)`.

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

`.append(item)` — в конец, `.insert(index, item)` — в позицию.

9. Как выполнить сортировку списка?

`lst.sort()` — изменяет список, `sorted(lst)` — возвращает новый список.

10. Как удалить один или несколько элементов из списка?

`.remove(value)`, `del lst[index]`, `.pop(index)`, срезом `lst[2:5] = []`.

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

`[x*2 for x in lst if x > 0]` — компактный синтаксис создания списка.

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

`lst[start:end:step]`, пример: `lst[1:5]`, `lst[::-1]` — реверс.

13. Какие существуют функции агрегации для работы со списками?

`len()`, `sum()`, `min()`, `max()`, `any()`, `all()`.

14. Как создать копию списка?

`copy = lst.copy()` или `copy = list(lst)` или `copy = lst[:]`.

15. Функция `sorted` vs метод `sort`:

`sorted(lst)` возвращает новый отсортированный список, не изменяя оригинал; `lst.sort()` сортирует сам список на месте.

16. Что такое кортежи в языке Python?

Коллекция упорядоченных элементов, неизменяемая (иммутабельная).

17. Каково назначение кортежей в языке Python?

Хранение фиксированных данных, использование как ключи словарей, возврат нескольких значений из функции.

18. Как осуществляется создание кортежей?

`t = (1, 2, 3)`, `t = tuple([1, 2, 3])`, `t = 1, 2, 3` (без скобок).

19. Как осуществляется доступ к элементам кортежа?

По индексу: `t[0]`, срезы: `t[1:3]`.

20. Зачем нужна распаковка (деструктуризация) кортежа?

`x, y, z = t` — присвоить элементы переменным.

21. Какую роль играют кортежи в множественном присваивании?

`a, b = b, a` — обмен значений без временной переменной.

22. Как выбрать элементы кортежа с помощью среза?

Так же как у списков: `t[1:4]`, `t[::-2]`.

23. Как выполняется конкатенция и повторение кортежей?

$(1, 2) + (3, 4) = (1, 2, 3, 4)$, $(1, 2) * 3 = (1, 2, 1, 2, 1, 2)$.

24. Как выполняется обход элементов кортежа?

for item in t:

25. Как проверить принадлежность элемента кортежу?

if element in t:

26. Какие методы работы с кортежами Вам известны?

.count(), .index().

27. Допустимо ли использование функций агрегации таких как len(), sum() и т. д. при работе с кортежами?

Да, все эти функции работают с кортежами.

28. Как создать кортеж с помощью спискового включения?

tuple(x*2 for x in range(5)).

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