

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ**

**Отчет о лабораторной работе №8 по дисциплине основы программной
инженерии**

Выполнил:
Выходцев Егор Дмитриевич,
2 курс, группа ПИЖ-б-о-20-1,

Проверил:
Доцент кафедры инфокоммуникаций,
Воронкин Р.А.

Ставрополь, 2021 г

1. Работа с кортежами в языке Python

Примеры из методических указаний

Что такое кортеж (tuple) в Python?

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

if __name__ == '__main__':
    a = [1, 2, 3]
    print(a)
    a[1] = 15
    print(a)
    b = (1, 2, 3)
    print(b)
    b[1] = 15
```

```
e1 x
C:\Users\Evil\PycharmProjects\LB8\venv\Scripts\python.exe C:/Users/Evil/PycharmPr
[1, 2, 3]
[1, 15, 3]
(1, 2, 3)
Traceback (most recent call last):
  File "C:\Users\Evil\PycharmProjects\LB8\examples\e1.py", line 11, in <module>
    b[1] = 15
TypeError: 'tuple' object does not support item assignment

Process finished with exit code 1
```

Зачем нужны кортежи в Python?

```
e2.py x
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == '__main__':
5     lst = [10, 20, 30]
6     tpl = (10, 20, 30)
7     print(lst.__sizeof__())
8     print(tpl.__sizeof__())
9
```

```
e2 x
C:\Users\Evil\PycharmProjects\LB8\venv\
104
48
Process finished with exit code 0
```

Создание кортежей

```
e3.py x
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 if __name__ == '__main__':
5     a = ()
6     print(type(a))
7     b = tuple()
8     print(type(b))
9     a = (1, 2, 3, 4, 5)
10    print(type(a))
11    print(a)
12    a = tuple([1, 2, 3, 4])
13    print(a)
14    not_a_tuple = (42)
15    tpl = (42,)
16    print(type(not_a_tuple))
17    print(type(tpl))
18
```

```
e3 x
C:\Users\Evil\PycharmProjects\LB8\venv\
<class 'tuple'>
<class 'tuple'>
<class 'tuple'>
(1, 2, 3, 4, 5)
(1, 2, 3, 4)
<class 'int'>
<class 'tuple'>
Process finished with exit code 0
```

Доступ к элементам кортежа

```
e4.py x
1 ▶ #!/usr/bin/env python3
2   # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == '__main__':
5     a = (1, 2, 3, 4, 5)
6     print(a[0])
7     print(a[1:3])
8     a[1] = 3
9
```

```
e4 x
C:\Users\Evil\PycharmProjects\LB8\venv\Scripts\python.exe C:/Users/Evil/PycharmP
Traceback (most recent call last):
  File "C:\Users\Evil\PycharmProjects\LB8\examples\e4.py", line 8, in <module>
    a[1] = 3
TypeError: 'tuple' object does not support item assignment
1
(2, 3)

Process finished with exit code 1
```

Удаление кортежей

```
e5.py x
1 ▶ #!/usr/bin/env python3
2   # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == '__main__':
5     a = (1, 2, 3, 4, 5)
6     del a[0]
7     del a
8     print(a)
9
```

```
e5 x
C:\Users\Evil\PycharmProjects\LB8\venv\Scripts\python.exe C:/Users/Evil/PycharmPro
Traceback (most recent call last):
  File "C:\Users\Evil\PycharmProjects\LB8\examples\e5.py", line 6, in <module>
    del a[0]
TypeError: 'tuple' object doesn't support item deletion

Process finished with exit code 1
```

```
e5 x
C:\Users\Evil\PycharmProjects\LB8\venv\Scripts\python.exe C:/Users/Evil/PycharmPro
Traceback (most recent call last):
  File "C:\Users\Evil\PycharmProjects\LB8\examples\e5.py", line 7, in <module>
    print(a)
NameError: name 'a' is not defined

Process finished with exit code 1
```

Преобразование кортежа в список и обратно

```
e6.py x
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == '__main__':
5     lst = [1, 2, 3, 4, 5]
6     print(type(lst))
7     print(lst)
8     tpl = tuple(lst)
9     print(type(tpl))
10    print(tpl)
11    # Обратная операция также является корректной.
12    tpl = (2, 4, 6, 8, 10)
13    print(type(tpl))
14    print(tpl)
15    lst = list(tpl)
16    print(type(lst))
17    print(lst)
18
```

```
e6 x
C:\Users\Evil\PycharmProjects\LB8\venv\Script
<class 'list'>
[1, 2, 3, 4, 5]
<class 'tuple'>
(1, 2, 3, 4, 5)
<class 'tuple'>
(2, 4, 6, 8, 10)
<class 'list'>
[2, 4, 6, 8, 10]

Process finished with exit code 0
```

Деструктуризация

```
e7.py x
1 > #!/usr/bin/env python3
2 > # -*- coding: utf-8 -*-
3
4 > if __name__ == '__main__':
5     name_and_age = ('Bob', 42)
6     (name, age) = name_and_age
7     print(f"name = {name}")
8     print(f"age = {age}")
9     (a,) = (42,)
10    print(f"a = {a}")
11
```

```
e7 x
C:\Users\Evil\PycharmProjects\LB8\venv
name = Bob
age = 42
a = 42

Process finished with exit code 0
```

Кортежи, множественное присваивание и обмен значениями

```
e8.py x
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == '__main__':
5     (a, b, c) = (1, 2, 3)
6     print(f"a = {a}")
7     print(f"b = {b}")
8     print(f"c = {c}")
9     a = 100
10    b = 'foo'
11    (a, b) = (b, a)
12    print(f"a = {a}")
13    print(f"b = {b}")
14
```

```
e8 x
C:\Users\Evil\PycharmProjects\LB8\venv\
a = 1
b = 2
c = 3
a = foo
b = 100

Process finished with exit code 0
```

Создание кортежа из итерированного объекта

```
e9.py x
1  ▶  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  ▶  if __name__ == '__main__':
5      # Операция tuple()
6      # 1. Создание кортежа из слова 'Hello'
7      d = tuple('Hello') # d = ('H', 'e', 'l', 'l', 'o')
8      # 2. Создание кортежа из списка
9      # Заданный список
10     lst = [2, "abc", 3.88]
11     # Создать кортеж
12     e = tuple(lst) # e = (2, 'abc', 3.88)
13     # 3. Создание кортежа из другого кортежа
14     f = tuple((3, 2, 0, -5)) # f = (3, 2, 0, -5)
15     print(f"d = {d}")
16     print(f"e = {e}")
17     print(f"f = {f}")
18
```

```
e9 x
C:\Users\Evil\PycharmProjects\LB8\venv
d = ('H', 'e', 'l', 'l', 'o')
e = (2, 'abc', 3.88)
f = (3, 2, 0, -5)

Process finished with exit code 0
```

Операция T[i:j]. Взятие среза в кортеже


```
e10.py x
1  ▶  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  ▶  if __name__ == '__main__':
5      # Операция [i:j] - взятие среза
6      # 1. Кортеж, содержащий целые числа
7      A = (0, 1, 2, 3)
8      item = A[0:2] # item = (0, 1)
9      print(f"item = {item}")
10     # 2. Кортеж, содержащий список
11     A = (2.5, ['abcd', True, 3.1415], 8, False, 'z')
12     item = A[1:3] # item = (['abcd', True, 3.1415], 8)
13     print(f"item = {item}")
14     # 3. Кортеж, содержащий вложенный кортеж
15     A = (3, 8, -11, "program")
16     B = ("Python", A, True)
17     item = B[:3] # item = ('Python', (3, 8, -11, 'program'), True)
18     print(f"item = {item}")
19     item = B[1:] # item = ((3, 8, -11, 'program'), True)
20     print(f"item = {item}")
21
```

```
e10 x
C:\Users\Evil\PycharmProjects\LB8\venv\Scripts\python.exe
item = (0, 1)
item = (['abcd', True, 3.1415], 8)
item = ('Python', (3, 8, -11, 'program'), True)
item = ((3, 8, -11, 'program'), True)
Process finished with exit code 0
```

Конкатенация

```
e11.py x
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == '__main__':
5     # Кортежи. Конкатенация +
6     # Конкатенация двух кортежей
7     A = (1, 2, 3)
8     B = (4, 5, 6)
9     C = A + B # C = (1, 2, 3, 4, 5, 6)
10    print(f"C = {C}")
11    # Конкатенация кортежей со сложными объектами
12    D = (3, "abc") + (-7.22, ['a', 5]) # D = (3, 'abc', -7.22, ['a', 5])
13    print(f"D = {D}")
14    # Конкатенация трех кортежей
15    A = ('a', 'aa', 'aaa')
16    B = A + (1, 2) + (True, False) # B = ('a', 'aa', 'aaa', 1, 2, True, False)
17    print(f"B = {B}")
18
```

```
e11 x
C:\Users\Evil\PycharmProjects\LB8\venv\Scripts
C = (1, 2, 3, 4, 5, 6)
D = (3, 'abc', -7.22, ['a', 5])
B = ('a', 'aa', 'aaa', 1, 2, True, False)
Process finished with exit code 0
```

Повторение

```
e12.py x
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == '__main__':
5     # Кортежи. Повторение *
6     # Кортеж, который содержит простые числа
7     A = (1, 2, 3) * 3 # A = (1, 2, 3, 1, 2, 3, 1, 2, 3)
8     # Кортеж, который содержит вложенные объекты
9     B = ("ab", ["1", "12"]) * 2 # A=('ab', ['1','12'], 'ab', ['1','12'])
10    print(f"A = {A}")
11    print(f"B = {B}")
12
```

```
e12 x
C:\Users\Evil\PycharmProjects\LB8\venv\Script
A = (1, 2, 3, 1, 2, 3, 1, 2, 3)
B = ('ab', ['1', '12'], 'ab', ['1', '12'])

Process finished with exit code 0
```

Обход кортежа в цикле

```
e13.py x
1  ▶  #!/usr/bin/env python3
2      # -*- coding: utf-8 -*-
3
4  ▶  if __name__ == '__main__':
5      # Обход кортежа в цикле
6      # 1. Цикл for
7      # Заданный кортеж
8      A = ("abc", "abcd", "bcd", "cde")
9      # Вывести все элементы кортежа
10     for item in A:
11         print(item)
12     # 2. Цикл while
13     # Исходный кортеж - целые числа
14     A = (-1, 3, -8, 12, -20)
15     # Вычислить количество положительных чисел
16     i = 0
17     k = 0 # количество положительных чисел
18     while i < len(A):
19         if A[i] < 0:
20             k = k + 1
21         i = i + 1
22     # Вывести результат
23     print("k = ", k)
24     # 3. Обход в цикле for
25     # Заданный кортеж, содержащий строки
26     A = ("abc", "ad", "bcd")
27     # Сформировать новый список из элементов кортежа A,
28     # в новом списке B, каждый элемент удваивается
29     B = [item * 2 for item in A]
30     print("A = ", A)
31     print("B = ", B)
32
```

```
e13 x
C:\Users\Evil\PycharmProjects\LB8\venv\Scripts\python.exe
abc
abcd
bcd
cde
k = 3
A = ('abc', 'ad', 'bcd')
B = ['abcabc', 'adad', 'bcdbcd']

Process finished with exit code 0
```

Операция in. Проверка вхождения элемента в кортеж

```
e14.py x
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 if __name__ == '__main__':
5     # Проверка вхождения элемента в кортеж
6     # Оператор in
7     # Заданный кортеж, который содержит строки
8     A = ("abc", "abcd", "bcd", "cde")
9     # Ввести элемент
10    item = str(input("s = "))
11    if item in A:
12        print(item, " in ", A, " = True")
13    else:
14        print(item, " in ", A, " = False")
15
```

```
e14 x
C:\Users\Evil\PycharmProjects\LB8\venv\Scripts\python.exe
s = abc
abc in ('abc', 'abcd', 'bcd', 'cde') = True

Process finished with exit code 0
```

```
e14 x
C:\Users\Evil\PycharmProjects\LB8\venv\Scripts\python.
s = fff
fff in ('abc', 'abcd', 'bcd', 'cde') = False

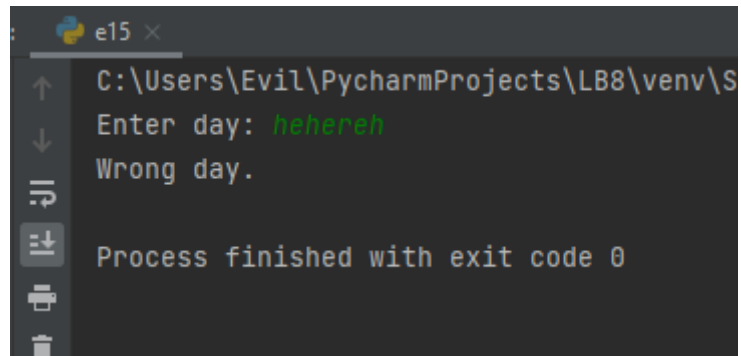
Process finished with exit code 0
```

Метод index(). Поиск позиции элемента в кортеже

```
e15.py x
1 > #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 > if __name__ == '__main__':
5     # Метод index - определяет позицию (индекс) элемента в кортеже
6     # Заданный кортеж
7     A = ("Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat")
8     # Запрос к вводу названия дня недели
9     day = str(input("Enter day: "))
10    # Корректно вычислить индекс
11    if day in A: # проверка, есть ли строка day в кортеже A
12        num = A.index(day)
13        print("Number of day = ", num + 1)
14    else:
15        num = -1
16        print("Wrong day.")
17
```

```
e15 x
C:\Users\Evil\PycharmProjects\LB8\venv\
Enter day: Sun
Number of day = 1

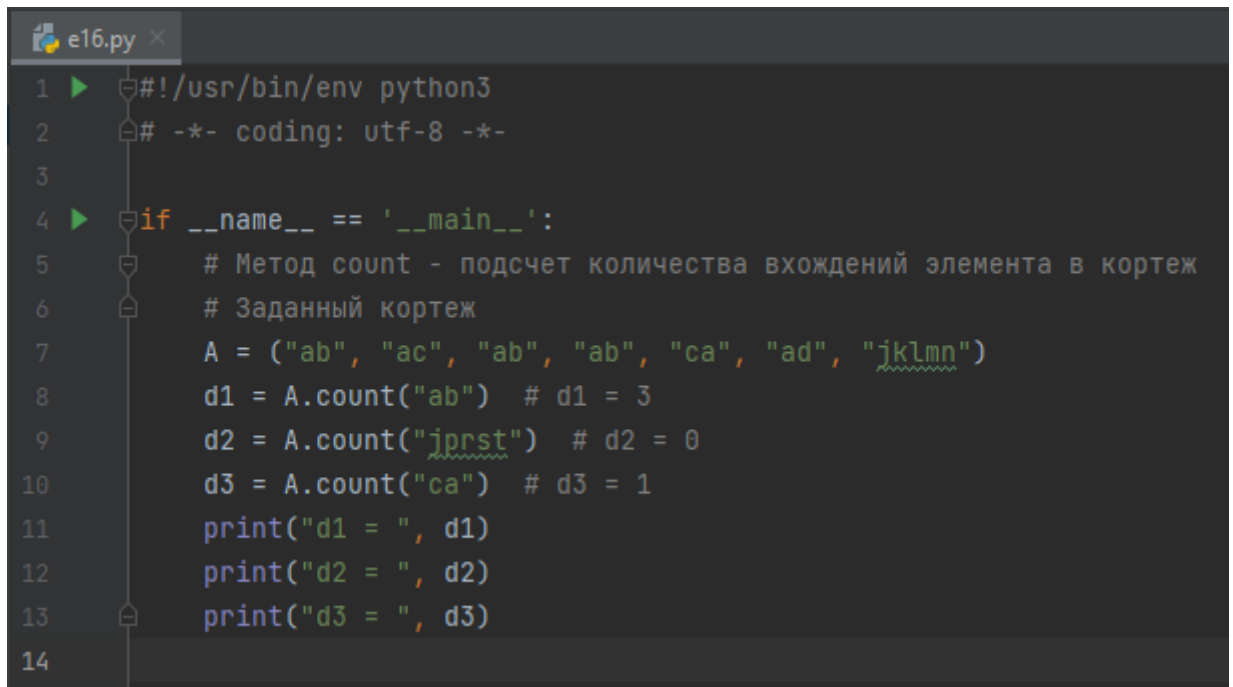
Process finished with exit code 0
```



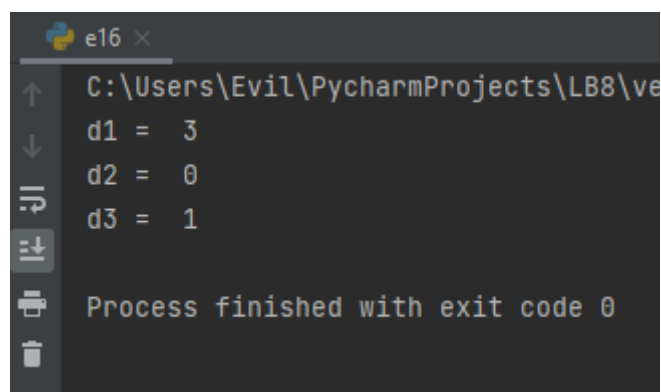
```
e15 x
C:\Users\Evil\PycharmProjects\LB8\venv\Scripts\python.exe
Enter day: hehereh
Wrong day.

Process finished with exit code 0
```

Метод count(). Количество вхождений элемента в кортеж



```
e16.py x
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  if __name__ == '__main__':
5      # Метод count - подсчет количества вхождений элемента в кортеж
6      # Заданный кортеж
7      A = ("ab", "ac", "ab", "ab", "ca", "ad", "jklmn")
8      d1 = A.count("ab") # d1 = 3
9      d2 = A.count("jprst") # d2 = 0
10     d3 = A.count("ca") # d3 = 1
11     print("d1 = ", d1)
12     print("d2 = ", d2)
13     print("d3 = ", d3)
14
```



```
e16 x
C:\Users\Evil\PycharmProjects\LB8\venv\Scripts\python.exe
d1 = 3
d2 = 0
d3 = 1

Process finished with exit code 0
```

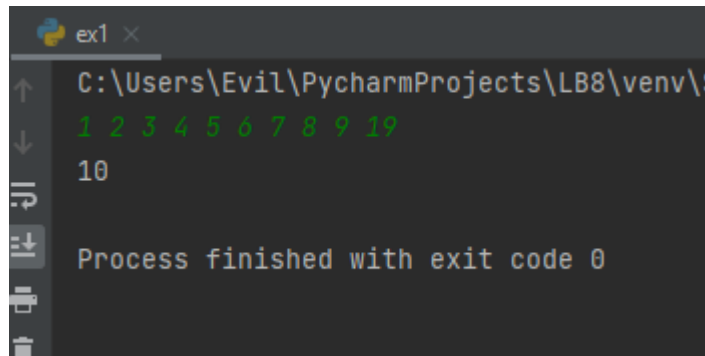
1.1 Пример 1 (рис. 1, 2).

```

1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 import sys
5
6 ▶ if __name__ == '__main__':
7     # Ввести список одной строкой.
8     A = tuple(map(int, input().split()))
9     # Проверить количество элементов списка.
10    if len(A) != 10:
11        print("Неверный размер списка", file=sys.stderr)
12        exit(1)
13    # Найти искомую сумму.
14    s = sum(a for a in A if abs(a) < 5)
15    print(s)
16

```

Рисунок 1 – Код примера



```

ex1 x
C:\Users\Evil\PycharmProjects\LB8\venv\
1 2 3 4 5 6 7 8 9 19
10
Process finished with exit code 0

```

Рисунок 2 – Пример работы программы

Решение индивидуальных заданий

Вариант 5

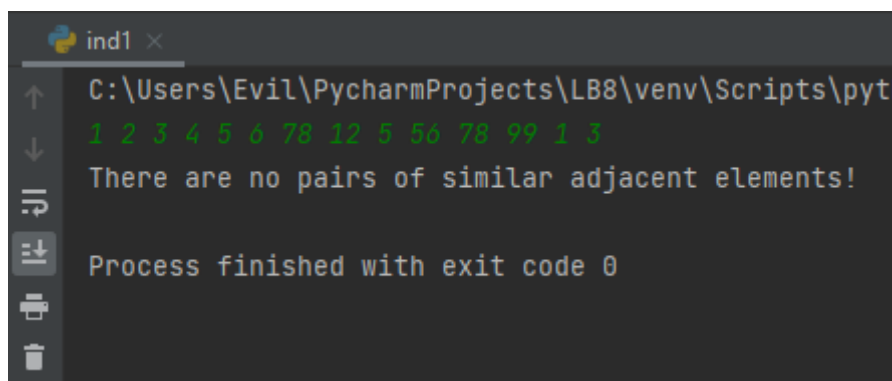
1.2 Индивидуальное задание №1 (рис. 3

```
ex1.py x ind1.py x
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5 ▶ if __name__ == '__main__':
6     A = tuple(map(int, input().split()))
7     i = 0
8     for index, el in enumerate(A):
9         k = A.count(el)
10        if k == 2:
11            if len(A) > index + 1:
12                if A[index + 1] == el:
13                    i = index + 1
14                    break
15            else:
16                continue
17        if i:
18            for index, el in enumerate(A):
19                if i < index < len(A):
20                    print(el, end=' ')
21        else:
22            print("There are no pairs of similar adjacent elements!")
23
```

Рисунок 3 – Код программы

```
ind1 x
C:\Users\Evil\PycharmProjects\LB8\venv\Sc
1 2 3 4 5 5 6 81 89 12 3 3 45 671
6 81 89 12 3 3 45 671
Process finished with exit code 0
```

Рисунок 4 – Пример работы программы при наличии двух пар соседних элементов



```
ind1 x
C:\Users\Evil\PycharmProjects\LB8\venv\Scripts\python
1 2 3 4 5 6 78 12 5 56 78 99 1 3
There are no pairs of similar adjacent elements!
Process finished with exit code 0
```

Рисунок 5 – Вывод программы при наличии пар одинаковых, но НЕ СОСЕДНИХ элементов

2. Ответы на контрольные вопросы

1. Что такое кортежи в языке Python?

Кортеж (tuple) – это неизменяемая структура данных, которая по своему подобию очень похожа на список.

2. Каково назначение кортежей в языке Python?

Существует несколько причин, по которым стоит использовать кортежи вместо списков. Одна из них – это обезопасить данные от случайного изменения. Если мы получили откуда-то массив данных, и у нас есть желание поработать с ним, но при этом непосредственно менять данные мы не собираемся, тогда, это как раз тот случай, когда кортежи придутся как нельзя кстати. Используя их в данной задаче, мы дополнительно получаем сразу несколько бонусов – во-первых, это экономия места. Дело в том, что кортежи в памяти занимают меньший объем по сравнению со списками. Во-вторых – прирост производительности, который связан с тем, что кортежи работают быстрее, чем списки (т. е. на операции перебора элементов и т. п. будет тратиться меньше времени). Важно также отметить, что кортежи можно использовать в качестве ключа у словаря.

3. Как осуществляется создание кортежей?

Для создания пустого кортежа можно воспользоваться одной из следующих команд:

```
a = ()
```

```
b = tuple()
```

Кортеж с заданным содержанием создается также как список, только вместо квадратных скобок используются круглые.

4. Как осуществляется доступ к элементам кортежа?

Доступ к элементам кортежа осуществляется также как к элементам списка — через указание индекса.

```
a = (1, 2, 3, 4, 5)
```

```
print(a[0])
```

5. Зачем нужна распаковка (деструктуризация) кортежа?

Дело в том, что кортежи часто содержат значения разных типов, и помнить, по какому индексу что лежит — очень непросто. Но есть способ лучше! Как мы кортеж собираем, так его можно и разобрать:

```
name_and_age = ('Bob', 42)
```

```
(name, age) = name_and_age
```

```
name # 'Bob'
```

```
age # 42
```

6. Какую роль играют кортежи в множественном присваивании?

Используя множественное присваивание, можно провернуть интересный трюк: обмен значениями между двумя переменными.

```
(a, b) = (b, a)
```

7. Как выбрать элементы кортежа с помощью среза?

Общая форма операции взятия среза для кортежа следующая:

$$T2 = T1[i:j]$$

T2 – новый кортеж, который получается из кортежа T1;

T1 – исходный кортеж, для которого происходит срез;

i, j – соответственно нижняя и верхняя границы среза.

Фактически берутся во внимание элементы, лежащие на позициях i, i+1, ..., j-1. Значение j определяет позицию за последним элементом среза.

8. Как выполняется конкатенация и повторение кортежей?

$$T3 = T1 + T2 \text{ – Конкатенация}$$
$$T2 = T1 * n \text{ – Повторение}$$

9. Как выполняется обход элементов кортежа?

Элементы кортежа можно последовательно просмотреть с помощью операторов цикла while или for.

10. Как проверить принадлежность элемента кортежу?

С помощью оператора «in»

11. Какие методы работы с кортежами Вам известны?

index() и count()

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

Да, т.к они не изменяют элементы в самом кортеже.

13. Как создать кортеж с помощью спискового включения.

$$A = \text{tuple}(\text{map}(\text{int}, \text{input}().\text{split}()))$$