

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное учреждение
высшего образования**

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития

Кафедра информационных систем и технологий

Отчет по лабораторной работе №10.

Дисциплина: «Основы программной инженерии»

Выполнил:

Студент группы ПИЖ-б-о-22-1,

направление подготовки: 09.03.04

«Программная инженерия»

ФИО: Джараян Арег Александрович

Проверил:

Воронкин Р. А.

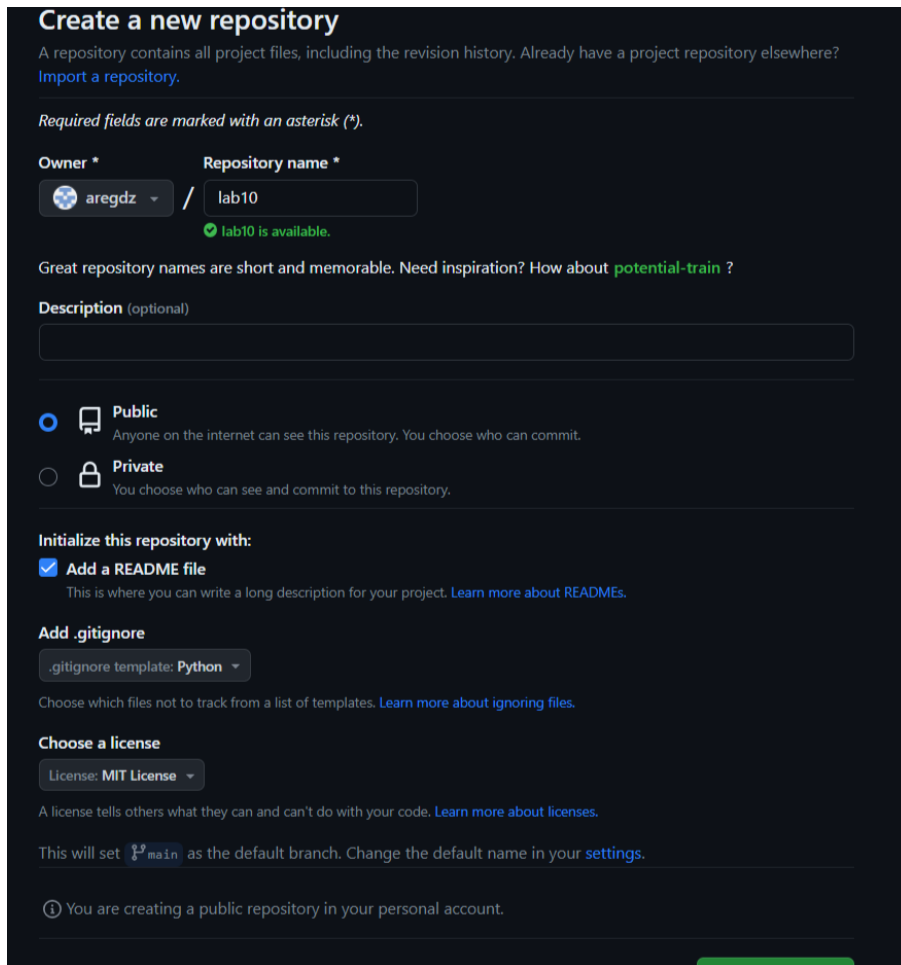
Ставрополь 2022

Тема: Лабораторная работа 2.7 Работа с множествами в языке Python.

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

Выполнение работы:

1. Изучил теоретический материал работы.
2. Создал репозиторий на git.hub.



The screenshot shows the 'Create a new repository' page on GitHub. The form includes fields for 'Owner' (set to 'aregdz') and 'Repository name' (set to 'lab10'). A green checkmark indicates 'lab10 is available'. Below this is a 'Description' field. The 'Public' option is selected for repository visibility. Under 'Initialize this repository with:', the 'Add a README file' checkbox is checked. The '.gitignore' section shows the 'Python' template selected. The 'Choose a license' section shows the 'MIT License' selected. At the bottom, a note states 'This will set main as the default branch. Change the default name in your settings.'

Рисунок 1 – создание репозитория

3. Клонировал репозиторий.

```

aregd@DESKTOP-5KV9QA9 MINGW64 ~
$ cd "C:\Users\aregd\OneDrive\Рабочий стол\git10"

aregd@DESKTOP-5KV9QA9 MINGW64 ~/OneDrive/Рабочий стол/git10
$ git clone https://github.com/aregdz/lab10.git
Cloning into 'lab10'...
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
Receiving objects: 100% (5/5), done.

aregd@DESKTOP-5KV9QA9 MINGW64 ~/OneDrive/Рабочий стол/git10
$ |

```

Рисунок 2 – клонирование репозитория 4.

4. Дополнить файл gitignore необходимыми правилами.

```

.gitignore – Блокнот
Файл Правка Формат Вид Справка
# Created by .ignore support plugin (hsz.mobi)
### Python template
# Byte-compiled / optimized / DLL files
__pycache__/
*.py[co]
*$py.class

# C extensions
*.so

# Distribution / packaging
.Python
env/
build/
develop-eggs/
dist/
downloads/

```

Рисунок 3 – .gitignore для IDE PyCharm

5. Организовать свой репозиторий в соответствии с моделью ветвления git-flow.

```

aregd@DESKTOP-5KV9QA9 MINGW64 ~/OneDrive/Рабочий стол/git10/lab10 (main)
$ git checkout -b develop
Switched to a new branch 'develop'

aregd@DESKTOP-5KV9QA9 MINGW64 ~/OneDrive/Рабочий стол/git10/lab10 (develop)
$

```

Рисунок 4 – создание ветки develop

6. Проработал примеры из методички.

```
1primer.py x
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3  if __name__ == "__main__":
4      # Определим универсальное множество
5      u = set("abcdefghijklmnopqrstuvwxyz")
6      a = {"b", "c", "h", "o"}
7      b = {"d", "f", "g", "o", "v", "y"}
8      c = {"d", "e", "j", "k"}
9      d = {"a", "b", "f", "g"}
10     x = (a.intersection(b)).union(c)
11     print(f"x = {x}")
12     # Найдем дополнения множеств
13     bn = u.difference(b)
14     cn = u.difference(c)
15     y = (a.difference(d)).union(cn.difference(bn))
16     print(f"y = {y}")
```

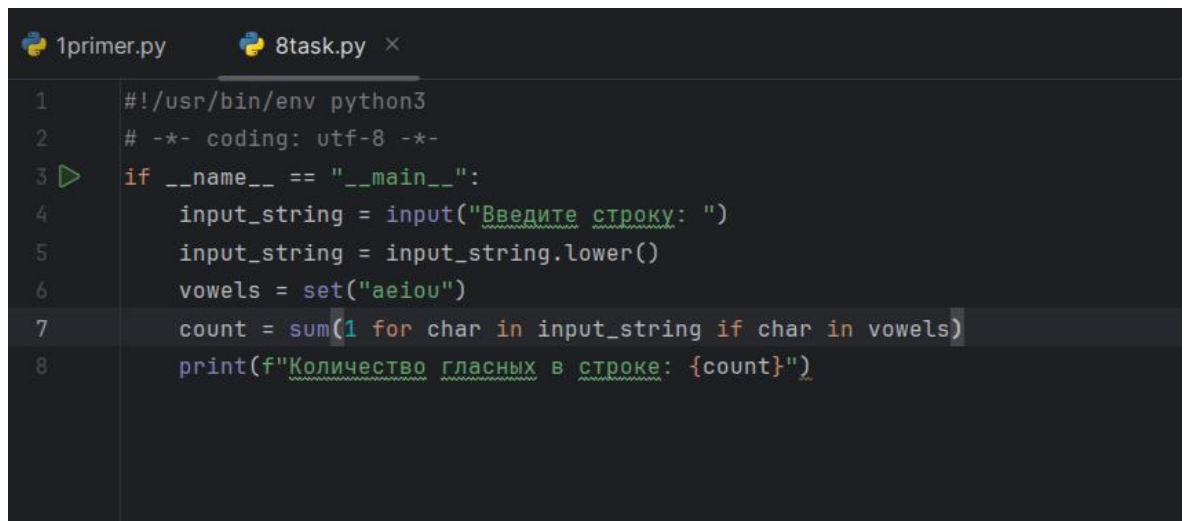
Рисунок 5 – пример 1

```
C:\Users\aregd\AppData\Local\Programs\Python\Python3
x = {'d', 'k', 'e', 'o', 'j'}
y = {'f', 'v', 'c', 'y', 'h', 'o', 'g'}

Process finished with exit code 0
```

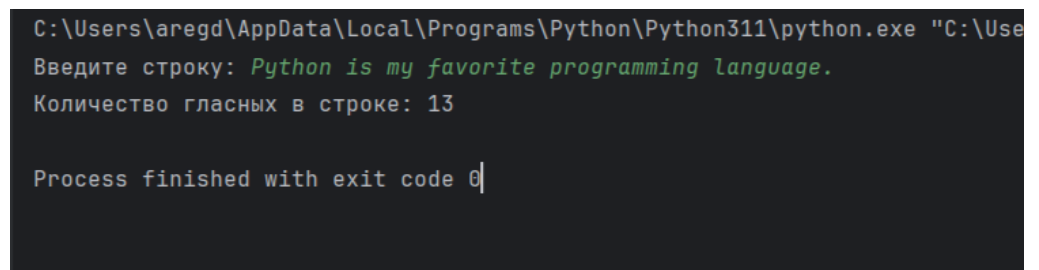
Рисунок 6 – пример выполнения примера 1

7. Подсчитайте количество гласных в строке, введенной с клавиатуры с использованием множеств.



```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3  if __name__ == "__main__":
4      input_string = input("Введите строку: ")
5      input_string = input_string.lower()
6      vowels = set("aeiou")
7      count = sum(1 for char in input_string if char in vowels)
8      print(f"Количество гласных в строке: {count}")
```

Рисунок 7 – задание 9

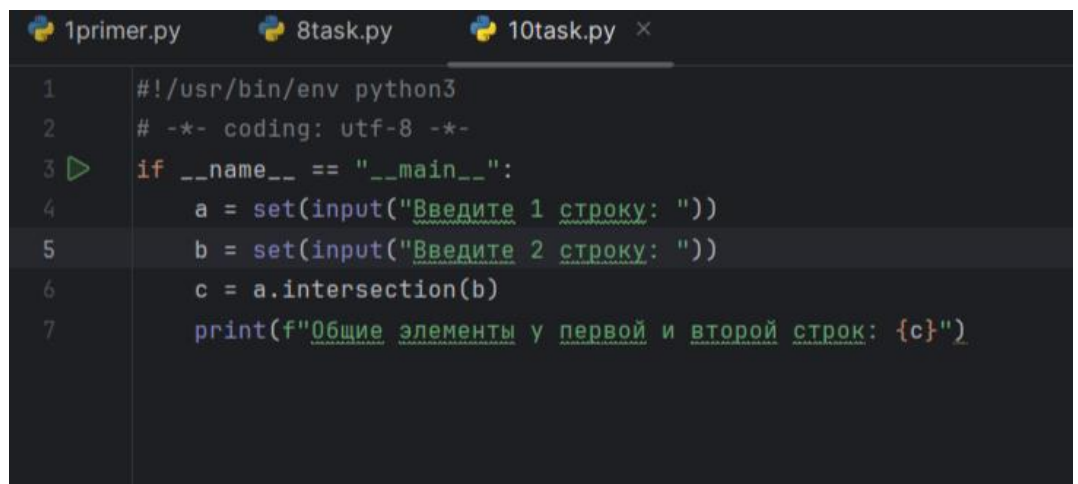


```
C:\Users\aregd\AppData\Local\Programs\Python\Python311\python.exe "C:\Use
Введите строку: Python is my favorite programming language.
Количество гласных в строке: 13

Process finished with exit code 0
```

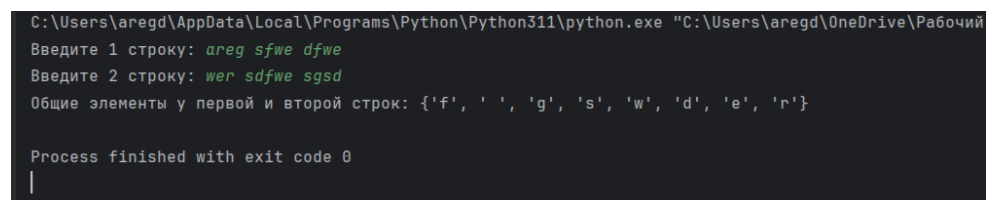
Рисунок 8 – пример выполнения 9 задания

8. Определите общие символы в двух строках, введенных с клавиатуры.



```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3  if __name__ == "__main__":
4      a = set(input("Введите 1 строку: "))
5      b = set(input("Введите 2 строку: "))
6      c = a.intersection(b)
7      print(f"Общие элементы у первой и второй строк: {c}")
```

Рисунок 9 – выполнение задания 11

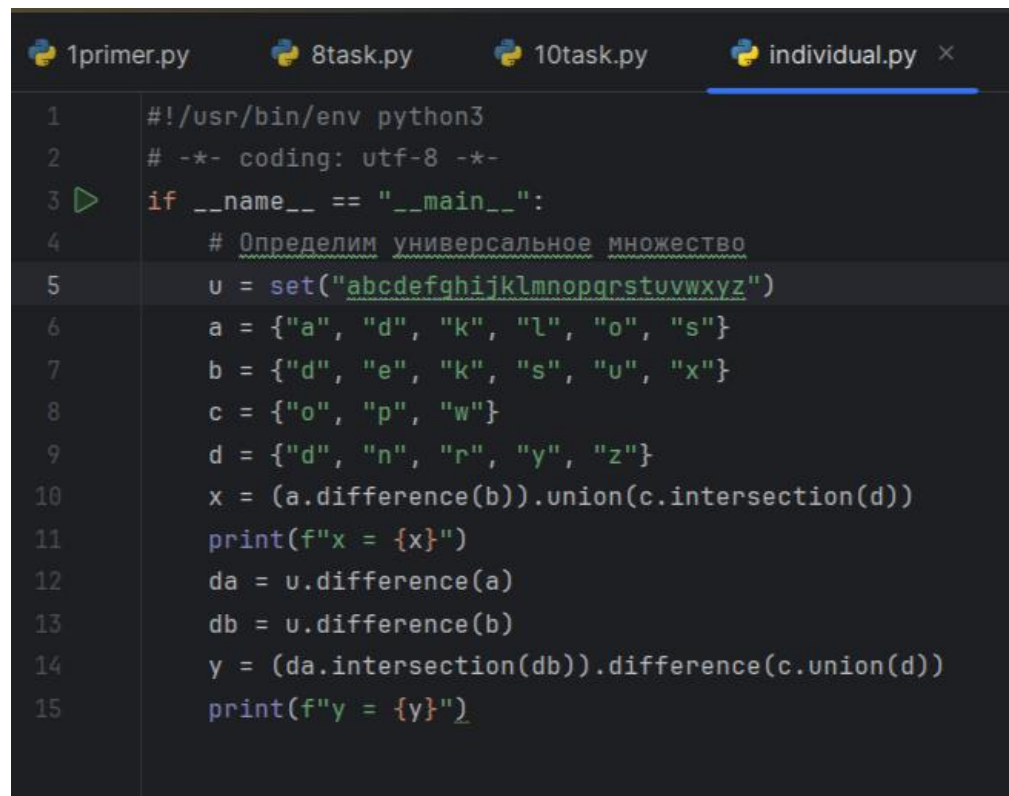


```
C:\Users\aregd\AppData\Local\Programs\Python\Python311\python.exe "C:\Users\aregd\OneDrive\Рабочий
Введите 1 строку: areg sfwe dfwe
Введите 2 строку: wer sdfwe sgsd
Общие элементы у первой и второй строк: {'f', ' ', 'g', 's', 'w', 'd', 'e', 'r'}

Process finished with exit code 0
```

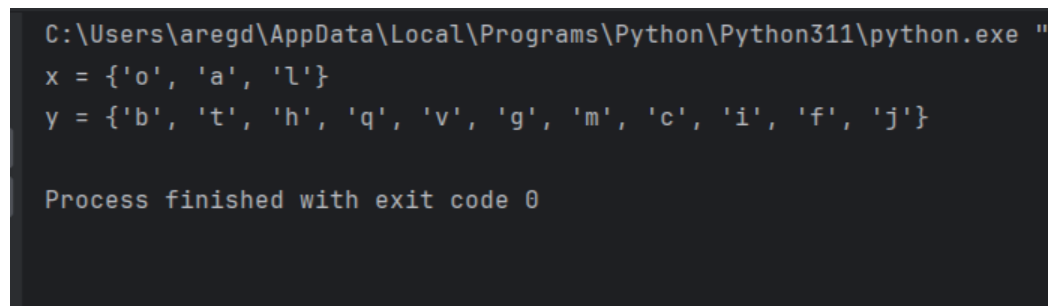
Рисунок 10 – результат выполнения задания 11

9. Индивидуальное задание. Определить результат выполнения операций над множествами. Считать элементы множества строками. Проверить результаты вручную.



```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3  if __name__ == "__main__":
4      # Определим универсальное множество
5      u = set("abcdefghijklmnopqrstuvwxyz")
6      a = {"a", "d", "k", "l", "o", "s"}
7      b = {"d", "e", "k", "s", "u", "x"}
8      c = {"o", "p", "w"}
9      d = {"d", "n", "r", "y", "z"}
10     x = (a.difference(b)).union(c.intersection(d))
11     print(f"x = {x}")
12     da = u.difference(a)
13     db = u.difference(b)
14     y = (da.intersection(db)).difference(c.union(d))
15     print(f"y = {y}")
```

Рисунок 11 – выполнение индивидуального задания

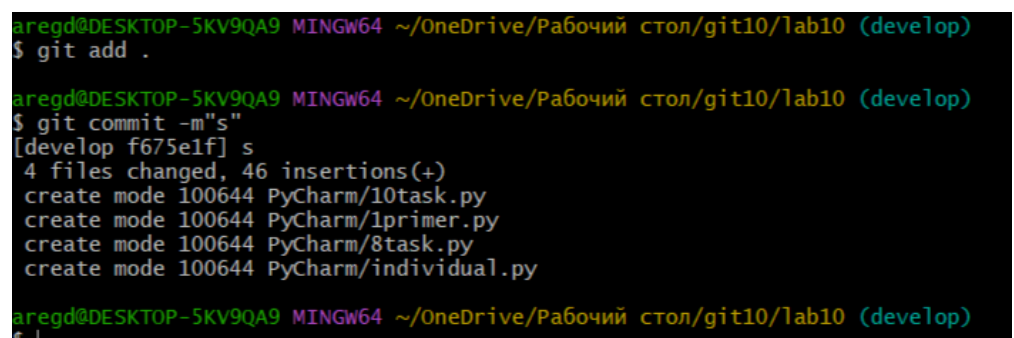


```
C:\Users\aregd\AppData\Local\Programs\Python\Python311\python.exe "
x = {'o', 'a', 'l'}
y = {'b', 't', 'h', 'q', 'v', 'g', 'm', 'c', 'i', 'f', 'j'}

Process finished with exit code 0
```

Рисунок 12 – результат выполнения индивидуального задания

9. Зафиксировал все изменения в github в ветке develop.



```
aregd@DESKTOP-5KV9QA9 MINGW64 ~/OneDrive/Рабочий стол/git10/lab10 (develop)
$ git add .

aregd@DESKTOP-5KV9QA9 MINGW64 ~/OneDrive/Рабочий стол/git10/lab10 (develop)
$ git commit -m"s"
[develop f675e1f] s
4 files changed, 46 insertions(+)
create mode 100644 PyCharm/10task.py
create mode 100644 PyCharm/1primer.py
create mode 100644 PyCharm/8task.py
create mode 100644 PyCharm/individual.py

aregd@DESKTOP-5KV9QA9 MINGW64 ~/OneDrive/Рабочий стол/git10/lab10 (develop)
$ |
```

Рисунок 13 – фиксация изменений в ветку develop

10. Слил ветки.

```
aregd@DESKTOP-5KV9QA9 MINGW64 ~/OneDrive/Рабочий стол/git10/lab10 (develop)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

aregd@DESKTOP-5KV9QA9 MINGW64 ~/OneDrive/Рабочий стол/git10/lab10 (main)
$ git merge develop
Updating 5788e82..f675e1f
Fast-forward
 PyCharm/10task.py | 7 ++++++
 PyCharm/1primer.py | 16 ++++++++
 PyCharm/8task.py | 8 ++++++
 PyCharm/individual.py | 15 ++++++++
 4 files changed, 46 insertions(+)
 create mode 100644 PyCharm/10task.py
 create mode 100644 PyCharm/1primer.py
 create mode 100644 PyCharm/8task.py
 create mode 100644 PyCharm/individual.py
```

Рисунок 14 – сливание ветки develop в ветку main

Вывод: приобрел навыки по работе с множествами при написании программ с помощью языка программирования Python версии 3.x.

Контрольные вопросы:

1. Что такое множества в языке Python?

Множеством в языке программирования Python называется неупорядоченная совокупность уникальных значений. В качестве элементов этого набора данных могут выступать любые неизменяемые объекты, такие как числа, символы, строки. В отличие от массивов и списков, порядок следования значений не учитывается при обработке его содержимого. Над одним, а также несколькими множествами можно выполнять ряд операций, благодаря функциям стандартной библиотеки языка программирования Python.

2. Как осуществляется создание множеств в Python?

Сделать это можно, просто присвоив переменной последовательность значений, выделив их фигурными скобками. Существует и другой способ создания множеств, который подразумевает использование вызова `set`. Аргументом этой функции может быть набор неких данных или даже строка с текстом.

3. Как проверить присутствие/отсутствие элемента в множестве?

Для этого используется `in`.

4. Как выполнить перебор элементов множества?

```
for a in {0, 1, 2}:
    print(a)
```

5. Что такое `set comprehension`?

Для создания множества можно в Python воспользоваться генератором, позволяющих заполнять списки, а также другие наборы данных с учетом неких условий.

6. Как выполнить добавление элемента во множество?

Чтобы внести новые значения, потребуется вызывать метод `add`. Аргументом в данном случае будет добавляемый элемент последовательности.

7. Как выполнить удаление одного или всех элементов множества?

Для удаления элементов из множества используются следующие функции в Python (кроме очистки, которая будет рассмотрена ниже):

`remove` — удаление элемента с генерацией исключения в случае, если такого элемента нет;

`discard` — удаление элемента без генерации исключения, если элемент отсутствует;

`pop` — удаление первого элемента, генерируется исключение при попытке удаления из пустого множества.

Иногда необходимо полностью убрать все элементы. Чтобы не удалять каждый элемент отдельно, используется метод `clear`, не принимающий аргументов.

8. Как выполняются основные операции над множествами: объединение, пересечение, разность?

Чтобы объединить все элементы двух разных множеств, стоит воспользоваться методом `union` на одном из объектов.

Чтобы добавить все элементы из одного множества к другому, необходимо вызывать метод `update` на первом объекте. Таким образом можно перенести уникальные данные из одного набора чисел в другой.

Чтобы найти общие элементы для двух разных множеств, следует применить функцию `intersection`, принимающую в качестве аргумента один из наборов данных.

Чтобы вычислить разность для двух разных множеств, необходимо воспользоваться методом `difference`. Функция позволяет найти элементы, уникальные для второго набора данных, которых в нем нет.

9. Как определить, что некоторое множество является надмножеством или подмножеством другого множества?

Чтобы выяснить, является ли множество `a` подмножеством `b`, стоит попробовать вывести на экран результат выполнения метода `issubset`, как в следующем примере.

Чтобы узнать, является ли множество `a` надмножеством `b`, необходимо вызвать метод `issuperset` и вывести результат его работы на экран.

10. Каково назначение множеств frozenset ?

frozenset в Python - это неизменяемая (immutable) версия типа данных "множество" (set). Основное назначение frozenset заключается в том, что оно может использоваться в ситуациях, где требуется неизменяемое множество, то есть множество, элементы которого нельзя изменить после его создания. Вот некоторые случаи, когда frozenset может быть полезным:

- Ключи в словаре: Поскольку словари Python могут использовать только неизменяемые объекты в качестве ключей, frozenset может быть использован в качестве ключа для словаря.
- Элементы множества в другом множестве: Вы можете создать множество, содержащее frozenset, чтобы использовать его в качестве элемента другого множества, так как frozenset является неизменяемым и поэтому может быть элементом множества.
- Защита от изменений: Если вам нужно гарантировать, что набор элементов останется неизменным и не будет изменен случайно или намеренно, вы можете использовать frozenset вместо set.

11. Как осуществляется преобразование множеств в строку, список, словарь?

Для преобразования множества в строку используется конкатенация текстовых значений, которую обеспечивает функция join. В этом случае ее аргументом является набор данных в виде нескольких строк. Запятая в кавычках выступает в качестве символа, разделяющего значения.

Чтобы получить из множества словарь, следует передать функции dict набор из нескольких пар значений, в каждом из которых будет находиться ключ.

По аналогии с предыдущими преобразованиями можно получить список неких объектов. На этот раз используется вызов list.