**Ayesha Ashraf**

**Deep learning (Task 5)**

**What is introducing sets?**

A Set in Python programming is an unordered collection data type that is iterable, mutable and has no duplicate elements.

**Set are represented by { } (values enclosed in curly braces)**

The major **Advantage** of using a set, as opposed to a list, is that it has a highly optimized method for checking whether a specific element is contained in the set. This is based on a data structure known as a hash table. Since sets are unordered, we cannot access items using indexes as we do in lists.

Example

# a set cannot have duplicate values

myset = {"Geeks", "for", "Geeks"}

print(myset)

# values of a set cannot be changed

myset[1] = "Hello"

print(myset)

**Error Generate:**

The first code explains that the set cannot have a duplicate value. Every item in it is a unique value.

The second code generates an error because we cannot assign or change a value once the set is created. We can only add or delete items in the set.

{'Geeks', 'for'}

TypeError: 'set' object does not support item assignment

In some cases it will not generate error but not duplicate data exist represent just single element.

### Heterogeneous Element with Python Set

Python sets can store heterogeneous elements in it, i.e., a set can store a mixture of string, integer, boolean, etc datatypes.

## Python Frozen Sets

**Frozen sets** in Python are immutable objects that only support methods and operators that produce a result without affecting the frozen set or sets to which they are applied. It can be done with [frozenset() method](https://www.geeksforgeeks.org/frozenset-in-python/) in Python.

While elements of a set can be modified at any time, elements of the frozen set remain the same after creation.

If no parameters are passed, it returns an empty frozenset.

**Different sets functions**

1. Add() : used “add” for add element
2. Union() : used symbol “|” or union for union 2 sets
3. Intersection():used symbol “&” or intersection for intersection 2 sets
4. Difference():used symbol “-” for difference 2 sets
5. Clear() : to clear the sets

**Union**

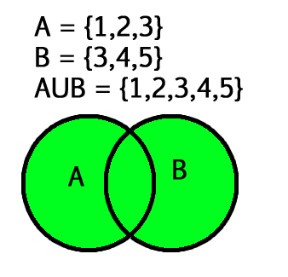
**Python set Union() Method** returns a new set which contains all the items from the original set.

**Union** of two given sets is the set which contains all the elements of both the sets. The union of two given sets A and B is a set which consists of all the elements of A and all the elements of B such that no element is repeated.

*The symbol for denoting union of sets is****‘U’***

**Python set Union() Method Syntax:**

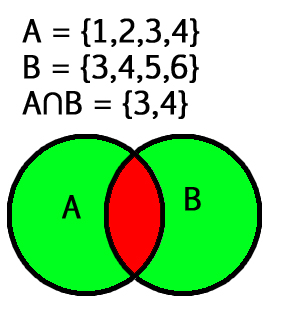
***Syntax:****set1.union(set2, set3, set4….)*



**Intersection**

**Python set intersection() method returns** a new set with an element that is common to all set

The intersection of two given sets is the largest set, which contains all the elements that are **common** to both sets. The intersection of two given sets A and B is a set which consists of all the elements which are common to both A and B.



## ****Python Set intersection() Method Syntax:****

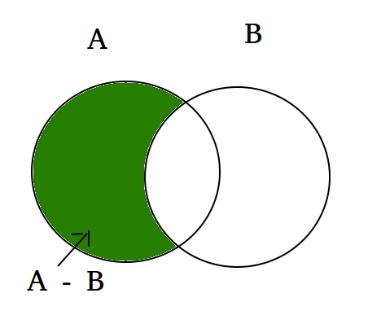
***Syntax:****set1.intersection(set2, set3, set4….)****Parameters:***

* *any number of sets can  be passed*

***Return:****Returns a set which has the intersection of all sets(set1, set2, set3…) with set1. It returns a copy of set1 only if no parameter is passed.*

**Difference**

The difference between the two sets in Python is equal to the difference between the number of elements in two sets. The function difference() returns a set that is the difference between two sets. Let’s try to find out what will be the difference between two sets A and B. Then (set A – set B) will be the elements present in set A but not in B and (set B – set A) will be the elements present in set B but not in set A.



Example:

set A = {10, 20, 30, 40, 80}

set B = {100, 30, 80, 40, 60}

set A - set B = {10, 20}

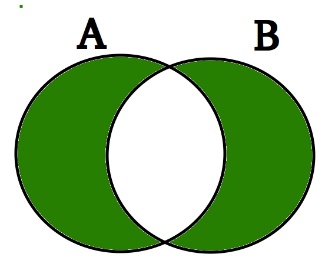
set B - set A = {100, 60}

Explanation: A - B is equal to the elements present in A but not in B

B - A is equal to the elements present in B but not in A

**Symmetric Difference**

**Python Set symmetric\_difference() Method**is used to get the **elements present in either of the two sets, but not common to both the sets**. Let’s look at the Venn diagram of the symmetric\_difference between two sets.



Symmetric Difference is marked in Green If there are a set\_A and set\_B, then the symmetric difference between them will be equal to the union of set\_A and set\_B without the intersection between the two.

It is the opposite of intersection.

## Python set symmetric\_difference() Method Syntax

***Syntax:****set\_A.symmetric\_difference(set\_B)*

***Parameter:****Takes a single parameter that has to be a set*

*Finding symmetric with this symbol ”^”*

**Making data unique with sets**

The set is the unordered collection of unique elements

Using [set()](https://www.geeksforgeeks.org/sets-in-python/) property of Python, we can easily check for the unique values. Insert the values of the list in a set. Set only stores a value once even if it is inserted more than once. After inserting all the values in the set by list\_set=set(list), convert this set to a list to print it.

**Enumerate function ()**

When dealing with iterators, we also get need to keep a count of iterations. Python eases the programmers’ task by providing a built-in function enumerate() for this task. Enumerate() method adds a counter to an iterable and returns it in a form of enumerating object. This enumerated object can then be used directly for loops or converted into a list of tuples using the list() function.

**Syntax:**

enumerate(iterable, start=0)

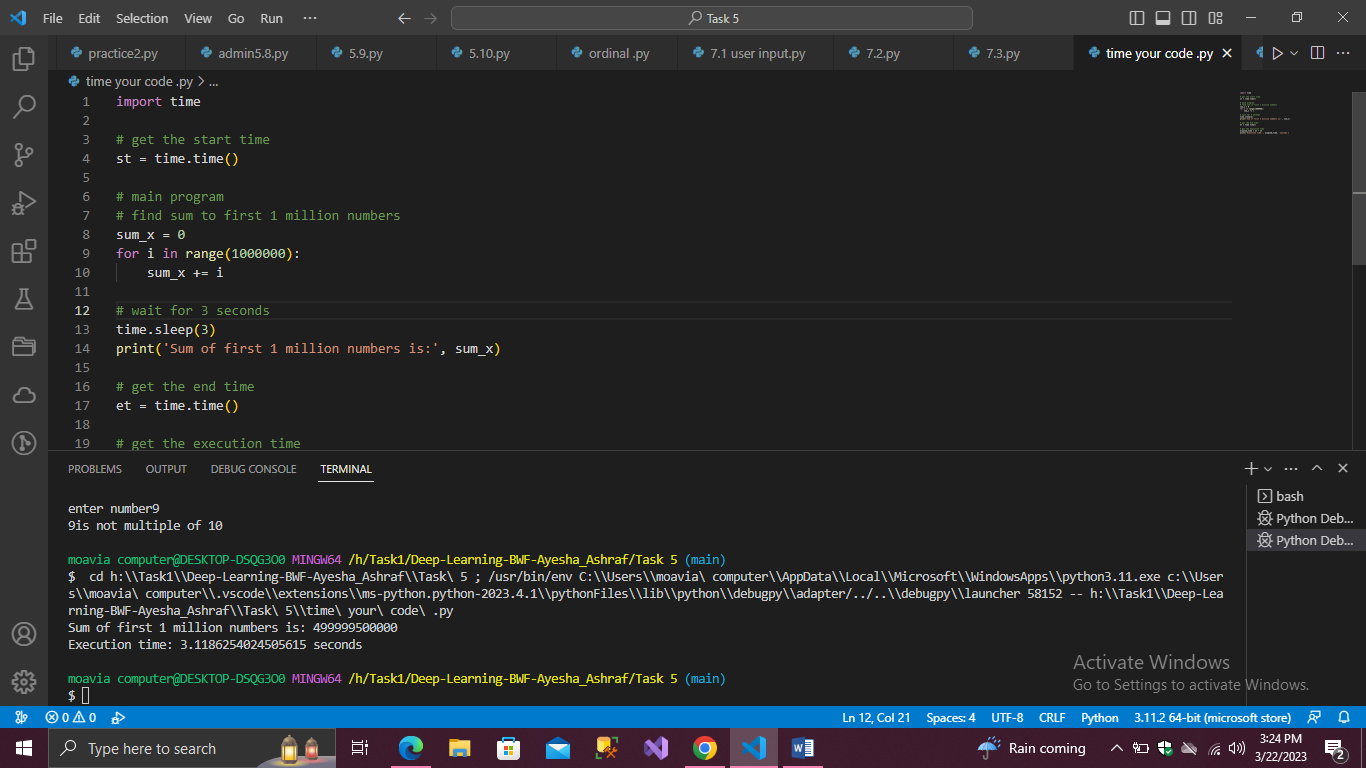
**Parameters:**

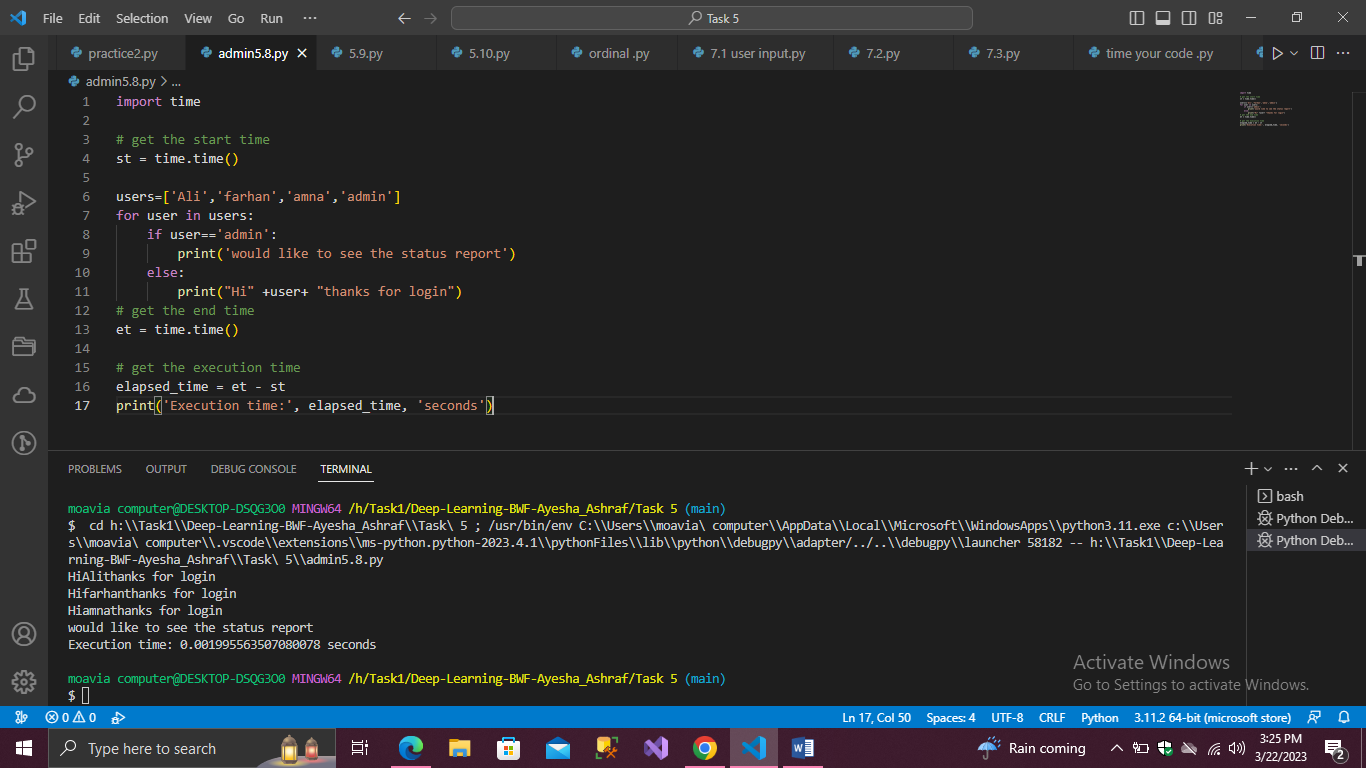
* **Iterable:**any object that supports iteration
* **Start:**the index value from which the counter is to be started, by default it is 0

**Time your code**

**We will use the following four ways** to measure the execution time in Python: –

* **time.time()** function: measure the the total time elapsed to execute the script in seconds.
* **time.process\_time()**: measure the CPU execution time of a code
* **timeit module**: measure the execution time of a small piece of a code including the single line of code as well as multiple lines of code
* **DateTime module**: measure the execution time in the hours-minutes-seconds format.

****

****