**Experiment Number – 7**

**Title – Write a program for Result Processing using dictionary**

**Theory –**

An effective data structure for storing data in Python is dictionaries, in which can simulate the real-life data arrangement where some specific value exists for some particular key.

* Python Dictionary is used to store the data in a key-value pair format.
* It is the mutable data-structure.
* The elements Keys and values is employed to create the dictionary.
* Keys must consist of just one element.
* Value can be any type such as list, tuple, integer, etc.

In other words, we can say that a dictionary is the collection of key-value pairs where the value can be of any Python object. In contrast, the keys are the immutable Python object, i.e., Numbers, string, or tuple. Dictionary entries are ordered as of Python version 3.7.

**Creating the Dictionary**

The simplest approach to create a Python dictionary is by using curly brackets {}, but there are other methods as well. The dictionary can be created by using multiple key-value pairs enclosed with the curly brackets {}, and each key is separated from its value by the colon (:). The syntax to define the dictionary is given below.

|  |
| --- |
| Dict = {}  Dict = {1: 'One', 2: 'Two', 3: 'Three'}  print(Dict) |

**Adding elements to a Dictionary**

Addition of elements can be done in multiple ways. One value at a time can be added to a Dictionary by defining value along with the key e.g. Dict[Key] = ‘Value’. Updating an existing value in a Dictionary can be done by using the built-in **update()** method. Nested key values can also be added to an existing Dictionary. For example,

Dict[4]=”Four”

print(Dict)

**Accessing elements of a Dictionary**

In order to access the items of a dictionary refer to its key name. Key can be used inside square brackets.

print(Dict[1])

**Change Value of Dictionary**

We can also use [] to change the value associated with a particular key. For example,

Dict[3]=”Three”

print(Dict)

**Deleting Elements using del Keyword**

The items of the dictionary can be deleted by using the del keyword as given below.

del Dict[2]

print(Dict)

**Python Dictionary Methods**

Methods that are available with a dictionary are tabulated below. Some of them have already been used in the above examples.

|  |  |
| --- | --- |
| **Method** | **Description** |
| dic.clear() | Remove all the elements from the dictionary |
| dict.copy() | Returns a copy of the dictionary |
| dict.get(key, default = “None”) | Returns the value of specified key |
| dict.items() | Returns a list containing a tuple for each key value pair |
| dict.keys() | Returns a list containing dictionary’s keys |
| dict.update(dict2) | Updates dictionary with specified key-value pairs |
| dict.values() | Returns a list of all the values of dictionary |
| pop() | Remove the element with specified key |
| popItem() | Removes the last inserted key-value pair |
| dict.setdefault(key,default= “None”) | Set the key to the default value if the key is not specified in the dictionary |
| dict.get(key, default = “None”) | used to get the value specified for the passed key. |

**Nested Dictionary**

In Python, a nested dictionary is a dictionary inside a dictionary. It's a collection of dictionaries into one single dictionary.

nested\_dict = { 'dictA': {'key\_1': 'value\_1'},

'dictB': {'key\_2': 'value\_2'}}

Here, the *nested\_dict* is a nested dictionary with the dictionary dictA and dictB. They are two dictionary each having own key and value.

## Create a Nested Dictionary

We're going to create dictionary of people within a dictionary.

people = {1: {'name': 'Arun', 'age': '27', 'gender': 'Male'},

2: {'name': 'Anup', 'age': '22', 'gender': 'Female'}}

print(people)

print(people[1]['name']) #to access individual dictionary element

Exercise –

1. Write a Python script to generate and print a dictionary that contains a number (between 1 and n) in the form (x, x\*x).
2. Write a Python program to drop empty items from a given dictionary.
3. Write a Python program to find the sum of all items in a dictionary.
4. Write a Python program to extract unique values from dictionary values.
5. Write a Python program to count frequency of letters in a string using dictionary.
6. Write a Python program to remove all duplicates words from a given sentence