Name=Kuldeep Gheghate

PRN=123B1B118

Batch =B3

ASSIGNMENT NO. 1(a)

Title: Create a dictionary to store student names and their corresponding scores in a test. Write functions to add new students, update scores, delete students, and find the student with the highest score.

Aim: Create a dictionary to store student names and their corresponding scores in a test. Write functions to add new students, update scores, delete students, and find the student with the highest score.

Topic Theory:

Dictionary: In Python, a dictionary is a built-in data structure that stores key-value pairs. It's especially useful for mapping unique keys to their corresponding values and allows for quick retrieval of values using their keys.

Key Characteristics of Dictionaries:

a) Dictionaries do not preserve any specific order for the key-value pairs.

b) Elements in a dictionary can be added, removed, or modified.

c) Each entry in a dictionary consists of a key and its associated value.

Functions in Dictionary:

a) Direct Assignment: You can add new key-value pairs to a dictionary by directly assigning a value to a new key. If the key already exists, this will update the existing value. Syntax: my_dict[key] = value

b) Update: This function updates the dictionary with key-value pairs from another dictionary or an iterable of key-value pairs. Syntax: dict.update(other_dict)

c) Delete: The del statement removes a key-value pair from the dictionary. If the key does not exist, it raises a KeyError. Syntax: del dict[key]

d) Direct Access: You can access a value directly using its key. Syntax: value = my_dict[key]

1. **Initialize the Dictionary**: Start by creating an empty dictionary to store student names along with their respective scores.
2. **Function Definitions**:

- **Add Student**:
    1. Verify if the student is already in the dictionary.
    2. If the student is not present, add the student and their score to the dictionary.
    3. If the student is already present, indicate that the student already exists.
- **Update Score**:
    1. Verify if the student is in the dictionary.
    2. If the student is present, update their score.
    3. If the student is not present, indicate that the student does not exist.
- **Delete Student**:
    1. Verify if the student is in the dictionary.
    2. If the student is present, remove them from the dictionary.
    3. If the student is not present, indicate that the student does not exist.
- **Find Highest Score**:
    1. Check if the dictionary is empty.
    2. If it is not empty, find the student with the highest score using the max() function.
    3. Return the name of the student with the highest score and the score itself.
    4. If the dictionary is empty, indicate that there are no students

Algorithm:

Step 1: Initialize an Empty Dictionary:

- Create an empty dictionary named student_scores to store student names and their corresponding scores.

Step 2: Define Functions:

- add_student(name, score): Adds a new student and their score to the dictionary. If the student already exists, display a message.
- update_score(name, score): Updates the score for an existing student. If the student does not exist, display a message.
- delete_student(name): Removes a student from the dictionary. If the student does not exist, display a message.
- view_data(): Displays all the student data stored in the dictionary.

- max_score(): Finds and displays the student with the highest score. If the dictionary is empty, display a message.

Step 3: Main Loop:

- Continuously prompt the user for a choice until they choose to exit.
- Based on the user's choice:
    - Option 1: Add a new student.
    - Option 2: Update an existing student's score.
    - Option 3: Delete a student from the dictionary.
    - Option 4: Find and display the student with the highest score.
    - Option 5: Display all student data.
    - Option 6: Exit the program.
- Display an error message if an invalid choice is entered.

Step 4: Exit Program:

- End the program based on the user's choice

CODE=

```
students_scores = {}

def addstudent(name, score):

in students_scores:

{name} already exists")

if name

 print(f"Student

else:

students_scores[name] = score

print(f"Student {name} added with score {score}.")

def updatescore(name,

score):

 if name in

students_scores:

students_scores[name] = score

print(f"Score for {name} updated to {score}.")
```

```
else:
print(f"Student {name} does not exist")
def deletestudent(name):
students_scores:
del
students_scores[name]
{name} has been deleted.")
 if name in
 print(f"Student
else:
print(f"Student {name} does not exist.")
def
find_highest_score():
if not students_scores:
print("No students available.")
return None
highest_score_student=max(students_scores,key=students_scores.
 get)
 highest_score = students_scores[highest_score_student]
return highest_score_student, highest_score
addstudent("Dinesh", 95)
addstudent("Neha", 85)
updatescore("sachin", 90)
deletestudent("natasha")
print(find_highest_score())
```

OUTPUT=

```
Student dinesh added with score 91.
Student pankaj added with score 81.
Score for sachin updated to 90.
Student natasha has been deleted.
('sachin', 90)
```