3/9/25

Task5:- Implement various Searching and sorting Operations in python programming.

Aim: To implement various searching and sorting Operations in python programming.

5.1 A company stores employee records in a list of dictionaries, where each dictionary contains id, name. and department. write a function find_employee by_id that takes this List and a target employee ID as arguments and return the dictionary of the employee with the matching ID or None if no such employee is found.

Algorithm:

1) Input Definition:
2) Define the function find_employee by_id that takes two parameters:
   a) A List of dictionaries (employees), where each dictionary represents an employee record with keys id, name, and department
   b) An integer (target_id) representing the employee ID to be searched
3) Iterate Through the list:-
   Use a for Loop to iterate through each dictionary in the employees list.
4) Check for Matching ID:
   within the loop, check if the id fieled of the current dictionary matches the target_id
5) return matching Record:
   If a match is found, return the current dictionary
6) Handle No Match:
   If the loop completes without finding a match return None.

Output:

```
{'id' : 2, 'name' = 'Bob', 'department' : 'Engineering'}
```

Program:

```
def find-employee_by_id(employees, target_id):
  for employee in employees:
    if employee[id] = = target_id:
        return employee
        return None

# Test the function
    employees = [
    {'id': 1, 'name': 'Alice', 'department': 'HR'},
   {'id': 2, 'name': 'Bob', 'department': 'Engineering'},
   {'id': 3, 'name': 'Charlie', 'department': 'sales'}.
   ]

Print(find-employee_by_id(employees, 2))
```

5.2 you are develobping a grade management system for a school. the system maintains a list of student records, where each record is represented as a dictionary containing a student's name and scores. The school needs to generated report that displays students' scores using the Bubble sort algorithm:

Algorithm:-

1) Initialization:

=) Get the length of the students list and store it in n:

2) outer loop:

=) Iterate from i = 0 to n-1 (inclusive). This loop represents the number of passes through the list

3) Track swaps:

=) Initialize a boolean variable swapped to False. This variable will track if any swaps are made in the current

4) Inner Loop:

* Iterate from j=0 to n-i-2 (inclusive). This loop compares adjacent elements in the list and performs swaps if necessary

5) Compare and swap:

* for each pair of adjacent element (i.e student[j] and students[j+1])
* compare their score values
* If student[i][score] > students[j+1]['score'], swap the two elements
* set swapped to true to idicate that a swap was made.

6) Early termination:

* After each pass of the inner Loop check if swapped is false. If no swaps were made during the pass List is already sotred and you can break out of the outer Loop early

7) completion:

* The function modifies the students list in place, sorting it by score.

<u>Program 5.2</u>

```
def bubble sort_scores (students):
        n = Len (students)
    for i in range (n):
    # Track if any swap is made in this pass
      swapped = false
        for j in range (0, n-i-1):
if students [j][source'] > students [j+1] ['score'];
#swap if the score of the currents student is greater
than the next student [j], student [j+1] , student[j]
        swapped = True
  # If no two elements were swapped, the list
is alredy sorted
   if not swapped:
     break
```

Output:

```
Before sorting:

{'name': 'Alice', 'score': 88}
{'name': 'Bob', 'score': 95}
{'name': 'charlie', 'score': 75}
{'name': 'Diana', 'score': 85}

After sorting:

{'name': 'Charlie', 'score': 75}
{'name': 'Diana', 'score': 85}
{'name': 'Alice', 'score': 88}
{'name': 'Bob', 'score': 95}
```

```
#Example usage
 Student = [
  {'name': 'Alice', 'score': 88}
  {'name': 'Bob', 'score': 95}
  {'name': 'Charlie', 'score': 75},
  {'name': 'Diana', 'score', 85}
 ]
 print("Before sorting")
   for student in students:
       print(student)
 bubble_sort_score(students)
 print("\n After sorting").
 for student in students:
 print (student)
```

VEL TECH	
EX NO.	5
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	4
RECORD (5)	5
TOTAL (20)	
SIGN WITH DATE	20

Result:
Thus, the program for various searching and sorting Opertions is executed and verified successfully.