Task 5.1
3/9/25

Implement various searching and sorting operations.

Aim:- To implement various searching and sorting operations in python programing

Algorithm:-

1. Input definition
2. Define the function find_employee_by_id that teles two
3. Iterate through the list use a for loop to iterate through each dictionary in the employees list
4. Check for matching ID.
5. Return for matching Record: If a match is found, return the current dictionary
6. Handle No match.
   If the loop completes without find a match return name

Program:-

```
def Find_employee_by_id Employees, target_id):
    for employee in Employees:
        if employee['id'] == target=id;
            return employee
        return None
    Employee = [
{ 'id' : 1, 'name' : 'Alice', department':
    'HR'}
{'id' : 2, 'name': 'BOB', department':
                Engineering' }:
{'id' : 3 'name': 'charlie', 'depr'sds}
Print find - employee_by_id (Employee. 2))
```

Result:- Thus to implement various searching and sorting operations in Python programing

Output

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

Task 5-2 :-

Aim :- To develop a Python program that starts students records by scores in ascending order using the bubble sort algorithm.

Algorithm :-

1. Intialization
   - find the length of the student list → n.
2. Outer loop (passes)
   - Repeat for i = 0 to n - 1
3. Track swaps
   - Set swapped = false at the start of each pass.
4. Inner loop (comparison)
   for each j = 0 to n - i - 2.
   - Compare students [j] ['store'] and students [j+1] ['score'].
   - If students [j] ['score'] > Student (j+1) ['score']:
     - Swap them
     - set swapped = True
5. Early Termination
   - After the inner loop, if swapped == false, break
6. Completion
   - The list is now sorted in ascending order to scores.

Output

Before sorting

1. {`name': `Alice', `score': 88}
2. {`name': `Bob', `score': 95}
3. {`name': `Charlie', `score': 75}
4. {`name': `Diana', `score': 85}

After sorting

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

Program :-

```
def bubble_sort_scores (students):
    n = len (students)
    For i in range (n);
    Swapped = false
    For j in range [0, n-i, -1];
    if students (j) ('score') > students [j+1]('score')
students [j], students [j+1] = students [j+1],
    swapped = True
    if not swapped
        break
{'name': 'Alice', 'score': 88},
{'name': 'Bob', 'score': 95},
{'name': 'charlie', 'score': 75},
{'name': 'Diane', 'sare': 85}
]
print ("Before sating:")
  for student in student:
      print (student)
bubble_sort_scores (student)
print("\n After sorting:")
for student in student:
    print (student)
```

VEL TECH - CSE	
VIVA	5
PERFORMANCE (5)	5
RESULT AND ANALYSIS (3)	3
VIVA VOCE (3)	3
RECORD (4)	4
TOTAL (15)	15
SIGN WITH DATE	

Result:- Thus the program for various Searching and Sorting Operations is executed and verified Successfully.