Task 5: Implement various searching and sorting operations in Python Programming.

Aim: To implement various searching and sorting operations in Python Programming

Algorithm:

1. Input Definition:
2. Define the function find_employee_by_id that takes two parameters.
3. Iterate through the list
4. Use a loop to iterate through each dictionary in employee list
5. Return matching found, return the current dictionary in employee
6. Handle No match
   If the loop completes without finding match, return None.

Program 5.1:

```
def find_employee_by_id(employees, target_id):
    for employee in employees:
        if employee[id] == target_id:
            return employee.
    return None.
# Test function
    employee = [
    {'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) # output {'id': 2, 'name':
                                    'Bob', 'department': 'Engineering'}
```

Output:

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

5.2:

Aim: To implement a feature that sorts the student records by their scores using the Bubble sorth Algorithm.

Algorithm:

1. Initialization:
   - Get the length of students and store it in.
2. outer loop:
   - Iterate from i=0 to n-1. this loop represents the number of passes through the list.
3. Track Swaps:
   - Initialize boolean variable swapped to false, This variable will track if any swaps are made in current pass.
4. Inner loop:
   - Iterate from i=0 to n-i-2 (inclusive). This loop compares adjacent elements in list and performs swaps if necessary.
6. Early Termination:
   After both pass of inner loop, Check if swapped if false. If no Swaps were during pass, the list is already sorted and you can break.
6. Completion:
   . The function modifies the students list in place sorting if by score.

Program:

```
def bubble_sort_scores(students):
    n = len(students)
    for in range(n):
# Track if any swap is made in his pass
    swapped = false
    for j in range (0, n-i-1);
    if students[j]["score"] > students[j+1]["score"];
```

Output:

Before sorting:

{'name': 'Alice', 'score': 88}

{'name': 'Bob', 'score': 95}

{'name': 'Bob', 'score': 75}

{'name': 'Charlie', 'score': 85}

Diana

After Sorting:

{'name': 'Charlie', 'score': 75}

{'name': 'Diana', 'score': 85}

{'name': 'Alice', 'score': 88}

{'name': 'Alice', 'score': 95}

```
# Swap if the score of the current student is greater than the next
    students[j], students[j+1] = students[j+1], students[j]
    swapped = True
# If no two elements were swapped, the list is already sorted.
    if not swapped:
        break
# Example usage
students = [
    {'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_scores(students)
Print("\nAfter sorting:")
for student in students:
    Print(student)
```

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

Result: Thus, the Program for various Searching and Sorting Operations is executed and verified Successfully.