Task-5 implement various searching and sorting operations
Date-03/9/25 in Python Programming

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

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 represented 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 employee list.
4. Check for Matching ID
within the loop, check if the id field of the current dictionary matches the target_id.
5. Return Matching Record.
& if match is found, return the current dictionary
6. Handle No Match.
if the loop completes without finding a match, return none.

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': 'name': Alice', 'department': 'HR'},
    {'id': 2, 'name': 'Bob', 'department': Engineering'},
    {'id': 3, 'name': 'charlie', 'department': 'sales'}
]
Printf(find_employee_by_id(employees,2)) # output: {'id': 2, 'name': 'Bob'
'department': Engineering'}
```

output :-

```
{'id' :2, 'name': 'bob', 'department': 'engineering'}
```

52. you are developing a grade management system for a School. The System maintain a list of student records where each record is represented as dictionary Containing a student's name and score. The School needs to generate a report that displays student's Scores in ascending order. You task is to implement a feature that Sorts the student records by their Scores using bubble sort algorithm.

Algorithm:-

1. Initialization

· Get the length of the Student list and Store it in n.

2. outer loop.

• Iterate from i=0 to n-1. This loop represented the number of passes through the list.

3. Track swaps.

• Intialize a boolean variable swapped to false. This variable will track if any swap are made in the current pass.

4. Inner loop.

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

5. Compare and swap

for each pair of adjacent elements

· Compare their Score values.

• if Student[j]['Score'] > Students[j+1]['Score'], Swap the two elements

· Set swapped to True to indicate that a swap was made.

6. Early Termination.

• After each pass of inner loop, check if swapped is false. If no Swaps were made during the pass, the list is already sorted, and You can break out of the outer loop early.

7. Completion.

· The function modifies the student list in place, sorting it by Score.

output:-

```
Befor sorting:
{'name':'alice','score':88}
{'name':'bob','score':95}
{'name':'charilé,'score': 75}
{'name': 'diana','score':85}
After sorting
[{'name':'alice','Score':88}
 {'name':'bob','Score':95}
 {'name':'Charile','Score':75}
 {'name':'diana','Score':85}]
```

```
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]['score'] > student [~~i~~j +1]['score'];
#Swap if the score of the current student is greater that the
next
 students [j], students [j+1] = students [j+1] ~~[student~~ [student [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("\n After Sorting:")
 for student in students:
   Print(student)
```

VEL TECH	5
EX NO.	[illegible]
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	
RECORD (5)	
L (20)	15
WITH DATE	

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