SUPERIOR UNIVERSITY

Data Structure and Algorithm (Lab)

Assignment – 4

Name:

Ali Maqsood.

Roll no:

SU92-BSAIM-F23-050.

Department:

Software Engineering Department.

Program:

Artificial Intelligence.

Section:

BSAI-3A

Selection Sort

Question # 1:

1. **Basic Selection Sort Implementation:**
   Implement a function to perform selection sort on a given list of integers.
   Example Input: [29, 10, 14, 37, 14]
   Expected Output: [10, 14, 14, 29, 37]

Code:

```
def task1(arr1):
    print(f"Input Array: {arr1}.")
    for i in range(len(arr1)-1):
        small=i
        for j in range(i+1,len(arr1)):
            if arr1[j]<arr1[small]:
                small=j
        arr1[i],arr1[small]=arr1[small],arr1[i]
    print(f"Output Array: {arr1}.")
task1([29, 10, 14, 37, 14])
```

Output:

```
E:\Uni\3rd Semester\4) Data Structures & Algorithms (Lab)\Assignments\Assignment 4>python task.py
Input Array: [29, 10, 14, 37, 14].
Output Array: [10, 14, 14, 29, 37].
```

Question # 2:

Sorting Strings Using Selection Sort:
Write a program to sort a list of strings alphabetically using selection sort.
Example Input: ["apple", "orange", "banana", "kiwi"]
Expected Output: ["apple", "banana", "kiwi", "orange"]

Code:

```
def task2(arr2):
    print(f"Input Array: {arr2}.")
    for i in range(len(arr2)-1):
        small=i
        for j in range(i+1,len(arr2)):
            if arr2[j]<arr2[small]:
                small=j
        arr2[i],arr2[small]=arr2[small],arr2[i]
    print(f"Output Array: {arr2}.")
task2(["apple","orange","banana","kiwi"])
```

Output:

```
E:\Uni\3rd Semester\4) Data Structures & Algorithms (Lab)\Assignments\Assignment 4>python task.py
Input Array: ['apple', 'orange', 'banana', 'kiwi'].
Output Array: ['apple', 'banana', 'kiwi', 'orange'].
```

Question # 3:

1. **Descending Order Selection Sort:**
   Modify the selection sort algorithm to sort the list of integers in descending order.
   Example Input: [12, 4, 45, 23, 18]
   Expected Output: [45, 23, 18, 12, 4]

Code:

```
def task3(arr3):
    print(f"Input Array: {arr3}.")
    for i in range(len(arr3)-1):
        small=i
        for j in range(i+1,len(arr3)):
            if arr3[j]<arr3[small]:
                small=j
        arr3[i],arr3[small]=arr3[small],arr3[i]
    arr3.reverse()
    print(f"Output Array: {arr3}.")
task3([12, 4, 45, 23, 18])
```

Output:

```
E:\Uni\3rd Semester\4) Data Structures & Algorithms (Lab)\Assignments\Assignment 4>python task.py
Input Array: [12, 4, 45, 23, 18].
Output Array: [45, 23, 18, 12, 4].
```

Question # 4:

1. **Selection Sort with Custom Comparators:**
   Implement a selection sort that can handle custom comparator functions to sort a list.
   Example: Sort based on the second character of each string.
   Example Input: ["cat", "bat", "apple", "car"]
   Expected Output: ["bat", "cat", "car", "apple"]

Code:

```
def task4(arr4):
    print(f"Input Array: {arr4}.")
    for i in range(len(arr4)-1):
        small=i
        for j in range(i+1,len(arr4)):
            if arr4[j][1]<arr4[small][1]:
                small=j
        arr4[i],arr4[small]=arr4[small],arr4[i]
    print(f"Output Array: {arr4}.")
task4(["cat", "bat", "apple", "car"])
```

Output:

```
E:\Uni\3rd Semester\4) Data Structures & Algorithms (Lab)\Assignments\Assignment 4>python task.py
Input Array: ['cat', 'bat', 'apple', 'car'].
Output Array: ['cat', 'bat', 'car', 'apple'].
```

Question # 5:

1. **Count the Number of Swaps:**
   Modify the selection sort algorithm to count the total number of swaps performed during sorting. Return the sorted list along with the count of swaps.

Code:

```
def task5(arr5):
    print(f"Input Array: {arr5}.")
    swaps=0
    for i in range(len(arr5)-1):
        small=i
        for j in range(i+1,len(arr5)):
            if arr5[j]<arr5[small]:
                small=j
        arr5[i],arr5[small]=arr5[small],arr5[i]
        swaps+=1
    print(f"Output Array: {arr5}.")
    print(f"Total Swaps Taken: {swaps}.")
task5([29, 10, 14, 37, 14])
```

Output:

```
E:\Uni\3rd Semester\4) Data Structures & Algorithms (Lab)\Assignments\Assignment 4>python task.py
Input Array: [29, 10, 14, 37, 14].
Output Array: [10, 14, 14, 29, 37].
Total Swaps Taken: 4.
```

Insertion Sort

Question # 6:

1. **Basic Insertion Sort Implementation:**
   Implement insertion sort to sort a list of integers.
   Example Input: [12, 11, 13, 5, 6]
   Expected Output: [5, 6, 11, 12, 13]

Code:

```
def task6(arr):
    print(f"Input Array: {arr}.")
    for i in range(1,len(arr)):
        key=arr[i]
        j=i-1
        while j >= 0 and key<arr[j]:
            arr[j+1]=arr[j]
            j-=1
        arr[j + 1]=key
    print(f"Output Array: {arr}.")
task6([12, 11, 13, 5, 6])
```

Output:

```
E:\Uni\3rd Semester\4) Data Structures & Algorithms (Lab)\Assignments\Assignment 4>python task.py
Input Array: [12, 11, 13, 5, 6].
Output Array: [5, 6, 11, 12, 13].
```

Question # 7:

1. **Insertion Sort for Linked Lists:**
   Implement insertion sort to sort the elements of a singly linked list in ascending order.

Code:

```
def task7(arr):
    print("Not Attempted.")
    pass
task7()
```

Output:

```
E:\Uni\3rd Semester\4) Data Structures & Algorithms (Lab)\Assignments\Assignment 4>python task.py
Not Attempted.
```

Question # 8:

1. **Binary Insertion Sort Optimization:**
   Implement insertion sort using binary search to find the appropriate position for inserting each element, thereby reducing the number of comparisons.

Code:

```
def task8(arr):
    print("Not Attempted.")
    pass
task8("True")
```

Output:

```
E:\Uni\3rd Semester\4) Data Structures & Algorithms (Lab)\Assignments\Assignment 4>python task.py
Not Attempted.
```

Question # 9:

1. **Sort a List of Tuples Using Insertion Sort:**
   Write a program that sorts a list of tuples based on the second element of each tuple using insertion sort.
   Example Input: [(1, 3), (4, 1), (2, 2)]
   Expected Output: [(4, 1), (2, 2), (1, 3)]

Code:

```
def task9(arr):
    print(f"Input Array: {arr}.")
    for i in range(1,len(arr)):
        key=arr[i]
        j=i-1
        while j >= 0 and key<arr[j]:
            arr[j+1]=arr[j]
            j-=1
        arr[j + 1]=key
    print(f"Output Array: {arr}.")
task9([(1,3),(4,1),(2,2)])
```

Output:

```
E:\Uni\3rd Semester\4) Data Structures & Algorithms (Lab)\Assignments\Assignment 4>python task.py
Input Array: [(1, 3), (4, 1), (2, 2)].
Output Array: [(1, 3), (2, 2), (4, 1)].
```

Question # 10:

1. **Insertion Sort with Reverse Sorting:**
   Modify the insertion sort to sort the given list of integers in descending order.
   Example Input: [5, 2, 9, 1, 5, 6]
   Expected Output: [9, 6, 5, 5, 2, 1]

Code:

```
def task10(arr):
    print(f"Input Array: {arr}.")
    for i in range(1,len(arr)):
        key=arr[i]
        j=i-1
        while j >= 0 and key<arr[j]:
            arr[j+1]=arr[j]
            j-=1
        arr[j + 1]=key
    arr.reverse()
    print(f"Output Array: {arr}.")
task10([12, 11, 13, 5, 6])
```

Output:

```
E:\Uni\3rd Semester\4) Data Structures & Algorithms (Lab)\Assignments\Assignment 4>python task.py
Input Array: [12, 11, 13, 5, 6].
Output Array: [13, 12, 11, 6, 5].
```

Question # 11:

1. **Count Shifts During Insertion Sort:**
   Track and print the number of shifts performed while sorting the list using insertion sort.

Code:

```
def task11(arr):
    print(f"Input Array: {arr}.")
    counter=0
    for i in range(1,len(arr)):
        key=arr[i]
        j=i-1
        while j >= 0 and key<arr[j]:
            counter+=1
            arr[j+1]=arr[j]
            j-=1
        arr[j + 1]=key
    print(f"Output Array: {arr}.")
    print(f"Total Swaps Taken: {counter}.")
task11([12, 11, 13, 5, 6])
```

Output:

```
E:\Uni\3rd Semester\4) Data Structures & Algorithms (Lab)\Assignments\Assignment 4>python task.py
Input Array: [12, 11, 13, 5, 6].
Output Array: [5, 6, 11, 12, 13].
Total Swaps Taken: 7.
```

Question # 12:

1. Insertion Sort in Matrix Sorting:
   Sort each row of a 2D matrix using insertion sort.

   Example Input:

   [[5, 1, 4],

   [3, 9, 2],

   [8, 6, 7]]

   Expected Output:

   [[1, 4, 5],

   [2, 3, 9],

   [6, 7, 8]]

Code:

```
def task12(matrix):
    print(f"Input Matrix: {matrix}.")
    for r in matrix:
        for i in range(1, len(r)):
            key=r[i]
            j=i - 1
            while j >= 0 and key < r[j]:
                r[j + 1]=r[j]
                j-=1
            r[j + 1] = key
    print(f"Output Matrix: {matrix}.")
task12([[5, 1, 4],[3, 9, 2],[8, 6, 7]])
```

Output:

```
E:\Uni\3rd Semester\4) Data Structures & Algorithms (Lab)\Assignments\Assignment 4>python task.py
Input Matrix: [[5, 1, 4], [3, 9, 2], [8, 6, 7]].
Output Matrix: [[1, 4, 5], [2, 3, 9], [6, 7, 8]].
```