DSA LAB – 9

Name: Etcherla Sai Manoj **Mis. No:** 112015044 **Branch:** CSE

Question 1:

Code:

```
#include<iostream>
using namespace std;

int Linear_search(int array[], int n, int x){
   cout << "Implementing Linear Search...\n";
   for(int i = 0; i < n; i++){
      if(array[i] == x){
         cout << "The element " << x << " is found at position " << i << " in the array\n";
         return 0;
      }
   }
   cout << "The element is not present in the array\n";
   return 0;
}

int Binary_search(int array[], int left, int right, int x){
   cout << "Implementing Binary Search...\n";
   while(left <= right){
      int middle = left + (right - left) / 2;
      if(array[middle] == x){
         cout << "The element " << x << " is found at position " << middle << " in the array\n";
         return 0;
      }
      if(array[middle] > x){
         right = middle - 1;
      }
      else{
         left = middle + 1;
      }
   }
   cout << "The element is not present in the array\n";
   return 0;
}

void Bubble_sort(int array[], int n){
   cout << "Implementing Bubble Sort...\n";
   for(int i = 0; i < n; i++){
      for(int j = 0; j < n-i; j++){
         if(array[j] > array[j+1]){
            swap(array[j], array[j+1]);
         }
      }
   }
}

void Insertion_sort(int array[], int n){
   cout << "Implementing Insertion Sort...\n";
   for(int i = 1; i < n; i++){
      int temp = array[i];
      int j = i -1;
      while(temp <= array[j] && j >= 0){
         array[j+1] = array[j];
         j = j - 1;
      }
      array[j+1] = temp;
   }
}

void Selection_sort(int array[], int n){
   cout << "Implementing Selection Sort...\n";
   for (int i = 0; i < n-1; i++){
      int min_idx = i;
      for (int j = i+1; j < n; j++)
      if (array[j] < array[min_idx]){
```

```
            min_idx = j;
        }
        swap(array[min_idx], array[i]);
    }
}

void Merge(int array[], int start, int middle, int end){
    int index = 0, temp[end - start + 1];
    int i = start, j = middle + 1;
    while(i <= middle && j <= end){
        if(array[i] <= array[j]){
            temp[index] = array[i];
            i = i + 1;
        }
        else{
            temp[index] = array[j];
            j = j + 1;
        }
        index = index + 1;
    }

        while(i <= middle){
            temp[index] = array[i];
            index = index + 1;
            i = i + 1;
        }

        while(j <= end){
            temp[index] = array[j];
            index= index + 1;
            j = j + 1;
        }

    for(i = start; i <= end; i++){
        array[i] = temp[i - start];
    }
}

void Merge_sort(int array[], int start, int end){
    if(start < end){
        int middle = (start + end) / 2;
        Merge_sort(array, start, middle);
        Merge_sort(array, middle + 1, end);
        Merge(array, start, middle, end);
    }
}

void heapify(int array[], int n, int i){
    int largest = i;
    int left = 2 * i + 1;
    int right = 2 * i + 2;
    if(left < n && array[left] > array[largest]){
        largest = left;
    }
    if(right < n && array[right] > array[largest]){
        largest = right;
    }
    if(largest != i){
        swap(array[i], array[largest]);
        heapify(array, n, largest);
    }
}

void Heap_sort(int array[], int n){
    cout << "Implementing Heap Sort...\n";
    for(int i = n / 2; i >= 0; i--){
        heapify(array, n, i);
    }
    for(int i = n - 1; i > 0; i--){
        swap(array[0], array[i]);
        heapify(array, i, 0);
    }
```

```
}

int partition(int array[], int start, int end){
   int left  = start, right = end, loc = start, flag = 0;
   while(flag == 0){
      while(array[loc] <= array[right] && loc != right){
         right = right - 1;
      }
      if(loc == right){
         flag = 1;
      }
      else if(array[loc] > array[right]){
         swap(array[loc], array[right]);
         loc = right;
      }
      if(flag == 0){
         while(array[loc] >= array[left] && loc != left){
            left = left + 1;
         }
         if(loc == left){
            flag = 1;
         }
         else if(array[loc] < array[left]){
            swap(array[loc], array[left]);
            loc = left;
         }
      }
   }
   return loc;
}

void Quick_sort(int array[], int start, int end){
   int loc;
   if(start < end){
      loc = partition(array, start, end);
      Quick_sort(array, start, loc - 1);
      Quick_sort(array, loc + 1, end);
   }
}
void display(int array[], int n){
   cout << "Sorted array : ";
   for(int i = 0; i < n; i++){
      cout << array[i] << " ";
   }
   cout << "\n";
}

int main(){
   int n, choice;
   cout << "Enter the size of array : ";
   cin >> n;
   int array[n];
   cout << "Enter the elements of array  : ";
   for(int i = 0; i < n; i++){
      cin >> array[i];
   }

   cout << "==================Menu==================\n";
   cout << "1. Linear Search\n";
   cout << "2. Binary Search\n";
   cout << "3. Bubble Sort\n";
   cout << "4. Insertion Sort\n";
   cout << "5. Selection Sort\n";
   cout << "6. Merge Sort\n";
   cout << "7. Heap Sort\n";
   cout << "8. Quick Sort\n";
   cout << "9. EXIT\n";
   cout << "========================================\n";

   while(1){
      cout << "\nEnter your choice to perform : ";
      cin >> choice;
```

```
switch(choice)
{
case 1:
   int a;
   cout << "Enter the value you want to search : ";
   cin >> a;
   Linear_search(array, n, a);
   break;
case 2:
   int b;
   cout << "Enter the value you want to search : ";
   cin >> b;
   Binary_search(array, 0, n-1, b);
   break;
case 3:
   Bubble_sort(array, n);
   display(array, n);
   break;
case 4:
   Insertion_sort(array, n);
   display(array, n);
   break;
case 5:
   Selection_sort(array, n);
      display(array, n);
      break;
   case 6:
      cout << "Implementing Merge Sort...\n";
      Merge_sort(array, 0, n -1);
      display(array, n);
      break;
   case 7:
      Heap_sort(array, n);
      display(array, n);
      break;
   case 8:
      cout << "Implementing Quick Sort...\n";
      Quick_sort(array, 0, n-1);
      display(array, n);
      break;
   case 9:
      return 0;
   default:
      cout << "Enter valid choice...!!!\n";
      break;
   }
  }
  return 0;
}
```

Input & Output:

Linear Search:

```
PS C:\Users\DELL\OneDrive\Desktop\Labs> cd "c:\Users\DELL\OneDrive\Desktop\Labs\DSA LAB\LAB 9\" ; if ($?) { g++ 1.cpp -o 1 } ; if ($?) { .\1 }
Enter the size of array : 5
Enter the elements of array  : 33 22 11 55 44
==================Menu==================
1. Linear Search
2. Binary Search
3. Bubble Sort
4. Insertion Sort
5. Selection Sort
6. Merge Sort
7. Heap Sort
8. Quick Sort
9. EXIT
========================================

Enter your choice to perform : 1
Enter the value you want to search : 44
Implementing Linear Search...
The element 44 is found at position 4 in the array

Enter your choice to perform : 1
Enter the value you want to search : 22
Implementing Linear Search...
The element 22 is found at position 1 in the array

Enter your choice to perform : 1
Enter the value you want to search : 11
Implementing Linear Search...
The element 11 is found at position 2 in the array

Enter your choice to perform : 1
Enter the value you want to search : 55
Implementing Linear Search...
The element 55 is found at position 3 in the array

Enter your choice to perform : 1
Enter the value you want to search : 33
Implementing Linear Search...
The element 33 is found at position 0 in the array

Enter your choice to perform : 9
PS C:\Users\DELL\OneDrive\Desktop\Labs\DSA LAB\LAB 9>
```

Binary Search:

```
PS C:\Users\DELL\OneDrive\Desktop\Labs> cd "c:\Users\DELL\OneDrive\Desktop\Labs\DSA LAB\LAB 9\" ; if ($?) { g++ 1.cpp -o 1 } ; if ($?) { .\1 }
Enter the size of array : 5
Enter the elements of array  : 11 22 33 44 55
==================Menu==================
1. Linear Search
2. Binary Search
3. Bubble Sort
4. Insertion Sort
5. Selection Sort
6. Merge Sort
7. Heap Sort
8. Quick Sort
9. EXIT
========================================

Enter your choice to perform : 2
Enter the value you want to search : 55
Implementing Binary Search...
The element 55 is found at position 4 in the array

Enter your choice to perform : 2
Enter the value you want to search : 22
Implementing Binary Search...
The element 22 is found at position 1 in the array

Enter your choice to perform : 2
Enter the value you want to search : 44
Implementing Binary Search...
The element 44 is found at position 3 in the array

Enter your choice to perform : 2
Enter the value you want to search : 10
Implementing Binary Search...
The element is not present in the array

Enter your choice to perform : 9
PS C:\Users\DELL\OneDrive\Desktop\Labs\DSA LAB\LAB 9>
```

Bubble Sort:

```
PS C:\Users\DELL\OneDrive\Desktop\Labs> cd "c:\Users\DELL\OneDrive\Desktop\Labs\DSA LAB\LAB 9\" ; if ($?) { g++ 1.cpp -o 1 } ; if ($?) { .\1 }
Enter the size of array : 5
Enter the elements of array  : 55 11 33 22 44
==================Menu==================
1. Linear Search
2. Binary Search
3. Bubble Sort
4. Insertion Sort
5. Selection Sort
6. Merge Sort
7. Heap Sort
8. Quick Sort
9. EXIT
========================================

Enter your choice to perform : 3
Implementing Bubble Sort...
Sorted array : 11 22 33 44 55

Enter your choice to perform : 9
PS C:\Users\DELL\OneDrive\Desktop\Labs\DSA LAB\LAB 9>
```

Insertion Sort:

```
PS C:\Users\DELL\OneDrive\Desktop\Labs> cd "c:\Users\DELL\OneDrive\Desktop\Labs\DSA LAB\LAB 9\" ; if ($?) { g++ 1.cpp -o 1 } ; if ($?) { .\1 }
Enter the size of array : 5
Enter the elements of array  : 55 11 33 22 44
==================Menu==================
1. Linear Search
2. Binary Search
3. Bubble Sort
4. Insertion Sort
5. Selection Sort
6. Merge Sort
7. Heap Sort
8. Quick Sort
9. EXIT
========================================

Enter your choice to perform : 4
Implementing Insertion Sort...
Sorted array : 11 22 33 44 55

Enter your choice to perform : 9
PS C:\Users\DELL\OneDrive\Desktop\Labs\DSA LAB\LAB 9>
```

Selection Sort:

```
PS C:\Users\DELL\OneDrive\Desktop\Labs> cd "c:\Users\DELL\OneDrive\Desktop\Labs\DSA LAB\LAB 9\" ; if ($?) { g++ 1.cpp -o 1 } ; if ($?) { .\1 }
Enter the size of array : 5
Enter the elements of array  : 55 11 33 22 44
==================Menu==================
1. Linear Search
2. Binary Search
3. Bubble Sort
4. Insertion Sort
5. Selection Sort
6. Merge Sort
7. Heap Sort
8. Quick Sort
9. EXIT
========================================

Enter your choice to perform : 5
Implementing Selection Sort...
Sorted array : 11 22 33 44 55

Enter your choice to perform : 9
PS C:\Users\DELL\OneDrive\Desktop\Labs\DSA LAB\LAB 9>
```

Merge Sort:

```
PS C:\Users\DELL\OneDrive\Desktop\Labs> cd "c:\Users\DELL\OneDrive\Desktop\Labs\DSA LAB\LAB 9\" ; if ($?) { g++ 1.cpp -o 1 } ; if ($?) { .\1 }
Enter the size of array : 5
Enter the elements of array  : 55 11 33 22 44
==================Menu==================
1. Linear Search
2. Binary Search
3. Bubble Sort
4. Insertion Sort
5. Selection Sort
6. Merge Sort
7. Heap Sort
8. Quick Sort
9. EXIT
========================================

Enter your choice to perform : 6
Implementing Merge Sort...
Sorted array : 11 22 33 44 55

Enter your choice to perform : 9
PS C:\Users\DELL\OneDrive\Desktop\Labs\DSA LAB\LAB 9>
```

Heap Sort:

```
PS C:\Users\DELL\OneDrive\Desktop\Labs> cd "c:\Users\DELL\OneDrive\Desktop\Labs\DSA LAB\LAB 9\" ; if ($?) { g++ 1.cpp -o 1 } ; if ($?) { .\1 }
Enter the size of array : 5
Enter the elements of array  : 55 11 33 22 44
==================Menu==================
1. Linear Search
2. Binary Search
3. Bubble Sort
4. Insertion Sort
5. Selection Sort
6. Merge Sort
7. Heap Sort
8. Quick Sort
9. EXIT
========================================

Enter your choice to perform : 7
Implementing Heap Sort...
Sorted array : 11 22 33 44 55

Enter your choice to perform : 9
PS C:\Users\DELL\OneDrive\Desktop\Labs\DSA LAB\LAB 9>
```

Quick Sort:

```
PS C:\Users\DELL\OneDrive\Desktop\Labs> cd "c:\Users\DELL\OneDrive\Desktop\Labs\DSA LAB\LAB 9\" ; if ($?) { g++ 1.cpp -o 1 } ; if ($?) { .\1 }
Enter the size of array : 5
Enter the elements of array  : 55 11 33 22 44
==================Menu==================
1. Linear Search
2. Binary Search
3. Bubble Sort
4. Insertion Sort
5. Selection Sort
6. Merge Sort
7. Heap Sort
8. Quick Sort
9. EXIT
========================================

Enter your choice to perform : 8
Implementing Quick Sort...
Sorted array : 11 22 33 44 55

Enter your choice to perform : 9
PS C:\Users\DELL\OneDrive\Desktop\Labs\DSA LAB\LAB 9>
```