Department of IT and Computer Science
Pak-Austria Fachhochschule: Institute of Applied Sciences and Technology, Haripur, Pakistan

COMP-261L Computer Organization and Assembly Language

Lab Report 05

Class: Computer Science

Name: Yaseen Ejaz Ahmed

Registration No.: B20F0283CS014

Semester: Third

Submission Date:

Submitted to: Engr. Rafi Ullah

_____

Instructor Signature

Lab No. 5

Sorting Algorithms: Quick, Merge & insertions Sort

Objectives:

To understand & implement the working of sorting algorithms using arrays in C++

Tools/Software Required:

C++ Compiler

Introduction:

SORTING:

It is often necessary to arrange the elements in an array in numerical order from highest to lowest values i.e.; from ascending to descending and vice versa. If an array contains string values or alphabetical order, then arrays need to be sorted. The process of sorting an array requires the exchanging of values. While this seems to be a simple process, a computer must be careful that no values are lost during this exchange.

Lab Tasks:

Lab Task 01: Writing a function to sort array elements using insertion sort**.**

[1,8,4,6,0,3,5,2,7,9]

Code:

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

InsertionSort(int a[],int n)
{
	int key;
	int i,j;

	for(i=1;i<n;i++)
	{
		key =a[i];

		j=i-1;

		while(j>=0 && a[j]>key)
		{
			a[j+1] = a[j];
			j--;
		}
		a[j+1]=key;
   }

   cout<<"\n\nSorted Array :\n";
   for(int i=0;i<n;i++)
   {
	cout<<a[i]<<"\t";
	}
}

int main()
{
	int n=10;
	int a[n]={1,8,4,6,0,3,5,2,7,9};

	cout<<"Original Array :\n";
```

```
for(int i=0;i<n;i++)
   {
        cout<<a[i]<<"\t";
        }

        InsertionSort(a,n);
}
```

Output:

```
Original Array :
1       8       4       6       0       3       5       2       7       9

Sorted Array :
0       1       2       3       4       5       6       7       8       9
```

Lab Task 02: Write a code to sort the following arrays using Quick sort method.
[10, 34, 45, 33, 23, 47, 31, 23, 45, 69, 2, 56,7,67, 88, 42]

Code:

```
#include <iostream>          //yaseen ejaz ahmed   B20F0283CS014
using namespace std;

void swapping(int* a,int* b)
{
        int temp=*a;
        *a=*b;
        *b=temp;
}
int partition(int a[],int l,int h)
{
        int pivot = a[h];               // pivot
        int i=(l-1);
        for (int j=l; j<=h-1;j++)
        {
                if (a[j]<pivot)
                {
                        i++;
                        swapping(&a[i],&a[j]);
                }
        }
```

```
        swapping(&a[i+1],&a[h]);
        return (i+1);
}

void quickSort(int a[],int low,int high)
{
        if (low<high)
        {
                int pi=partition(a,low,high);

                quickSort(a,low,pi-1);
                quickSort(a,pi+1,high);
        }
}


void Show(int a[],int size)
{
        int i;
        for (i=0;i<size;i++)
        cout<<a[i]<<"\t";
}

int main()
{
        int a[] = {10, 34, 45, 33, 23, 47, 31, 23, 45, 69, 2, 56,7,67, 88, 42};
        int size = sizeof(a)/sizeof(a[0]);

        cout << "\nThe array is\n";
        Show(a,size);

        quickSort(a,0,size-1);

        cout<<"\n\nSorted array:\n";
        Show(a,size);
}
```

Output:

```
The array is
10      34      45      33      23      47      31      23      45      69      2       56      7       67      88      42

Sorted array:
2       7       10      23      23      31      33      34      42      45      45      47      56      67      69      88
```

Lab Task 03: Write a code to sort the following arrays using Merge sort method**.**

[10, 34, 33, 22, 77, 98, 2, 56, 7, 55, 56, 67, 88, 42]

Code:

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

void Merge(int array[],int const left,int const mid,int const right)
{
   int const subArrayOne=mid -left+1;
   int const subArrayTwo=right-mid;


   int *leftArray = new int[subArrayOne];
   int *rightArray = new int[subArrayTwo];


   for (int i=0; i<subArrayOne;i++)
   leftArray[i]=array[left+i];
   for (int j=0; j<subArrayTwo;j++)
   rightArray[j]=array[mid+1+j];

   int indexOfSubArrayOne=0;
   int indexOfSubArrayTwo=0;
   int indexOfMergedArray=left;


   while (indexOfSubArrayOne<subArrayOne && indexOfSubArrayTwo<subArrayTwo) {
      if (leftArray[indexOfSubArrayOne]<=rightArray[indexOfSubArrayTwo]) {
         array[indexOfMergedArray]=leftArray[indexOfSubArrayOne];
         indexOfSubArrayOne++;
      }
      else {
         array[indexOfMergedArray]=rightArray[indexOfSubArrayTwo];
```

```
            indexOfSubArrayTwo++;
        }
        indexOfMergedArray++;
    }

    while (indexOfSubArrayOne < subArrayOne) {
        array[indexOfMergedArray]=leftArray[indexOfSubArrayOne];
        indexOfSubArrayOne++;
        indexOfMergedArray++;
    }
    while (indexOfSubArrayTwo < subArrayTwo) {
        array[indexOfMergedArray]=rightArray[indexOfSubArrayTwo];
        indexOfSubArrayTwo++;
        indexOfMergedArray++;
    }
}

void Merge(int array[],int const begin,int const end)
{
    if (begin>=end)
    return;

    int mid=begin+(end-begin)/2;
    Merge(array,begin,mid);
    Merge(array,mid+1,end);
    Merge(array,begin,mid,end);
}

void Show(int a[],int size)
{
    for (int i=0; i<size;i++)
    cout<<a[i]<<"\t";
}

int main()
{
            int a[]={10, 34, 33, 22, 77, 98, 2, 56, 7, 55, 56, 67, 88, 42};
            int size = sizeof(a)/sizeof(a[0]);

    cout << "\nThe array is\n";
    Show(a,size);
```

```
    Merge(a,0,size-1);

    cout << "\n\nSorted array is \n";
    Show(a, size);
}
```

Output:

```
The array is
10      34      33      22      77      98      2       56      7       55      56      67      88      42

Sorted array is
2       7       10      22      33      34      42      55      56      56      67      77      88      98
```

Lab Task 04: Write a code to sort the following arrays using Insertion sort method**.**

[12, 31, 41, 37, 49, 45, 69, 2, 56,76,67, 98, 52]

Code:

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

void InsertionSort(int a[],int n)
{
        int key;
        int i,j;

        for(i=1;i<n;i++)
        {
                key =a[i];

                j=i-1;

                while(j>=0 && a[j]>key)
                {
                        a[j+1] = a[j];
                        j--;
                }
                a[j+1]=key;
    }
    cout<<"\n\nSorted Array :\n";
    for(int i=0;i<n;i++)
    {
```

```
        cout<<a[i]<<"\t";
        }
}

int main()
{
        int n=13;

        int a[n]={12, 31, 41, 37, 49, 45, 69, 2, 56,76,67, 98, 52};

        cout<<"Original Array :\n";
        for(int i=0;i<n;i++)
   {
        cout<<a[i]<<"\t";
        }

        InsertionSort(a,n);

}
```

Output:

```
Original Array :
12      31      41      37      49      45      69      2       56      76      67      98      52

Sorted Array :
2       12      31      37      41      45      49      52      56      67      69      76      98
```

Results & Observations:

In this lab we have learnt about different sorting algorithms. We can use different types of these sorting algorithms for different data. Some types of sorting may be better than others depending on the data and time complexity.