Assignment-03

Name : **Imtiaz Ahammad Mollah**

ID No – 2018100010023

Section-411

**Program:**

#include <ctime>

#include <iostream>

#include <cstdlib>

#include<stdio.h>

using namespace std;

//Heap Sort Start

void heapify(int array[], int len, int i)

{

int largest = i;

int left = 2 \* i + 1;

int right = 2 \* i + 2;

if (left < len && array[left] > array[largest])

largest = left;

if (right < len && array[right] > array[largest])

largest = right;

if (largest != i)

{

swap(array[i], array[largest]);

heapify(array, len, largest);

}

}

void heapSort(int array[], int len)

{

for (int i = len / 2 - 1; i >= 0; i--)

heapify(array, len, i);

for (int i = len - 1; i >= 0; i--)

{

swap(array[0], array[i]);

heapify(array, i, 0);

}

}

//Heap Sort End

//Quicksort Start

int partition(int \*a,int start,int end)

{

int pivot=a[end];

//P-index indicates the pivot value index

int P\_index=start;

int i,t;

for(i=start; i<end; i++)

{

if(a[i]<=pivot)

{

t=a[i];

a[i]=a[P\_index];

a[P\_index]=t;

P\_index++;

}

}

t=a[end];

a[end]=a[P\_index];

a[P\_index]=t;

return P\_index;

}

void Quicksort(int \*a,int start,int end)

{

if(start<end)

{

int P\_index=partition(a,start,end);

Quicksort(a,start,P\_index-1);

Quicksort(a,P\_index+1,end);

}

}

//QuickSort Stop

//Marge Sort Function Start

void merge(int arr[], int l, int m, int r)

{

int i, j, k;

int n1 = m - l + 1;

int n2 = r - m;

int L[n1], R[n2];

for (i = 0; i < n1; i++)

L[i] = arr[l + i];

for (j = 0; j < n2; j++)

R[j] = arr[m + 1+ j];

/\* Merge the temp arrays back into arr[l..r]\*/

i = 0;

j = 0;

k = l;

while (i < n1 && j < n2)

{

if (L[i] <= R[j])

{

arr[k] = L[i];

i++;

}

else

{

arr[k] = R[j];

j++;

}

k++;

}

while (i < n1)

{

arr[k] = L[i];

i++;

k++;

}

while (j < n2)

{

arr[k] = R[j];

j++;

k++;

}

}

void mergeSort(int arr[], int l, int r)

{

if (l < r)

{

int m = l+(r-l)/2;

mergeSort(arr, l, m);

mergeSort(arr, m+1, r);

merge(arr, l, m, r);

}

}

//Marge Sort Function End

int main()

{

int arr [10000];

int i=0;

while(i++ < 10001)

{

int r = (rand() % 1000) + 1;

arr[i]=r;

}

int start\_s=clock();

mergeSort(arr, 0, 10000);

int stop\_s=clock();

int start\_s1=clock();

Quicksort(arr,0,10000);

int stop\_s1=clock();

int start\_s2=clock();

heapSort(arr, 10000);

int stop\_s2=clock();

cout << "time Marge Sort: " << (stop\_s-start\_s) <<" mi sc"<< endl;

cout << "time Quick Sort: " << (stop\_s1-start\_s1) <<" mi sc" << endl;

cout << "time Heap Sort: " << (stop\_s2-start\_s2)<<" mi sc"<< endl;

return 0;

}