Rajshahi University of Engineering & Technology

CSE 2202: Sessional Based on CSE 2201

Lab Report 03

Date: November 20, 2018

Submitted to

Biprodip Pal

Instructor, CSE 2101 & CSE 2202

Assistant Professor, Dept. of CSE

Submitted by

Fuad Al Abir

Roll: 1603021

Section: A

Dept. of CSE

Sessional – Cycle 3 – Problem A

Divide and conquer sorting approach to quick sort.

Code:

```
/*-----------------------------
    I N T R O D U C T I O N
-----------------------------
Author:     Fuad Al Abir
Date:       November 18, 2018
Name:       quickSort.cpp
Objective:  1. This program defines a function that places the first element of its
argument array in the kth smallest position where it belongs on the sorted array.
            2. It uses the concept to sort an array in divide and conquer approach
by calling the function recursively.
*/
/*-----------------------------------
    H E A D E R   F U N C T I O N
-----------------------------------
Header: iostream
Reason: Input/Output stream
Header: cstdlib
Reason: For functions rand and srand
Header: time.h
Reason: For function time, and for data type time_t
*/
#include <iostream>
#include <cstdlib>
#include <time.h>
using namespace std;

/*-----------------------------------------------
    U S E R   D E F I N E D   F U N C T I O N
-----------------------------------------------
Function:   int placeFirst(int data[], int start, int end);
Reason:     This function places the first element to its right position
Function:   void quickSort(int data[], int start, int end);
Reason:     This function sorts the entire array by placeFirst() method using
Divide and conquer approach
*/
int placeFirst(int data[], int start, int end)
{
    while(start != end)
    {
        while(data[start] < data[end]) end--;

        int temp = data[start];
        data[start] = data[end];
        data[end] = temp;

        while(data[start] < data[end]) start++;

        temp = data[start];
        data[start] = data[end];
        data[end] = temp;
    }
    return start;
}
void quickSort(int data[], int start, int end)
{
    if(start < end)
    {
        int position = placeFirst(data, start, end);
        quickSort(data, start, position);
        quickSort(data, position + 1, end);
    }
```

```
}
/*-------------------------------
    M A I N   F U N C T I O N
-------------------------------*/
int main()
{
    time_t random_seed;         // a variable of type time_t is declared, which
holds seconds on clock
    time(&random_seed);         // get variable from system clock and store it in t
    srand(random_seed);         // pass random_seed as seed of rand()

    int size;
    cout << "Enter the size of the array: ";
    cin >> size;

    // Initialize array with random numbers
    int data[size];
    for(int i = 0; i < size; i++)
    {
        data[i] = rand();
    }

    // Printing the array initialized with random numbers
    cout << "Initialized array with random value:\t";
    for(int i = 0; i < size; i++)
    {
        cout << data[i] << " ";
    }
    cout << endl;

    // To place the first element to its right position only
    //placeFirst(data, 0, size - 1);

    // Sorting function quickSort() is called
    quickSort(data, 0, size - 1);

    // Sorted array is printed
    cout << "After sorting the entire array:\t\t";
    for(int i = 0; i < size; i++)
    {
        cout << data[i] << " ";
    }
    cout << endl;

    return 0;
}
```

Input/Output:

Enter the size of the array: 10

Initialized array with random value: **24774** 19130 24129 16243 22714 26045 17882 6915 23101 17116

After placing the first value in its place: 17116 19130 24129 16243 22714 23101 17882 6915 **24774** 26045

Enter the size of the array: 10

Initialized array with random value: 25140 10536 26068 24358 31067 7087 6028 6740 4624 16661

After sorting the entire array: 4624 6028 6740 7087 10536 16661 24358 25140 26068 31067