MULTIMEDIA UNIVERSITY OF KENYA

FACULTY OF COMPUTING AND INFORMATICS (FOCIT)

DEPARTMENT OF COMPUTER SCIENCE (CS)

DESIGN AND ANALYSIS OF ALGORITHMS

LAB MANUAL

Year : 2022

Course Code : CSC

Prepared by

Dr. Bonface Ngari Ireri

(Senior Lecturer/Director CODEL)

Learning Outcomes

By the end of the practical sessions, the learner should be able to:

1. Design and implement efficient algorithms for a specified application
2. Identify and use appropriate tools of analyzing algorithms
3. Make appropriate decision in selecting a suitable algorithm for a specific real world problem.

LABS/EXPERIMENTS

LAB-1 QUICK SORT

1. Using a quicksort algorithm, Sort a given set of elements and determine the time required to sort the elements.
2. Repeat the experiment for different values of n, the number of elements in the 1st to be sorted and
3. Plot a graph of the time taken versus n.

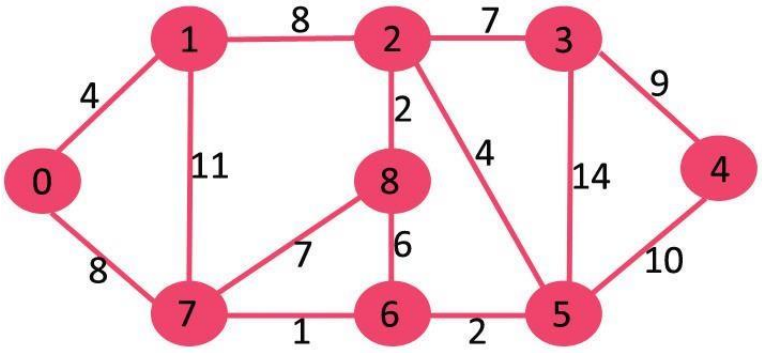
   NB:The elements can be read from a file or can be generated using the random number generator.

LAB-2 MERGE SORT

Repeat the experiment in LAB1 sing a Merge Sort Algorithm.

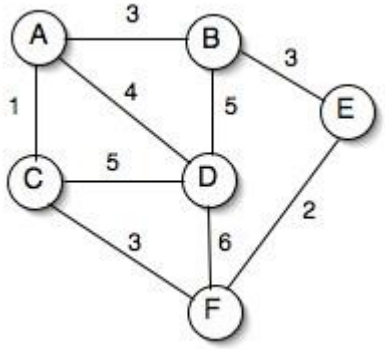
LAB-3 SHORTEST PATHS ALGORITHM

From a given vertex in a weighted connected graph, find shortest paths to other vertices using Dijkstra's algorithm.



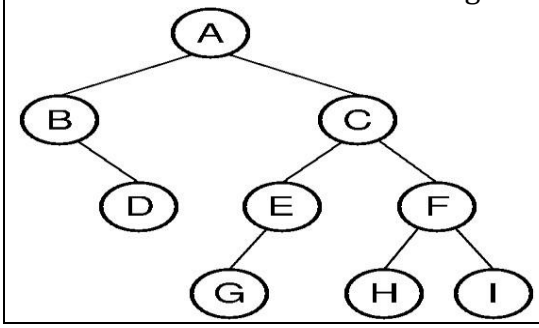
LAB-4 MINIMUM COST SPANNING TREE

Find Minimum Cost Spanning Tree of a given undirected graph using Kruskal's algorithm.



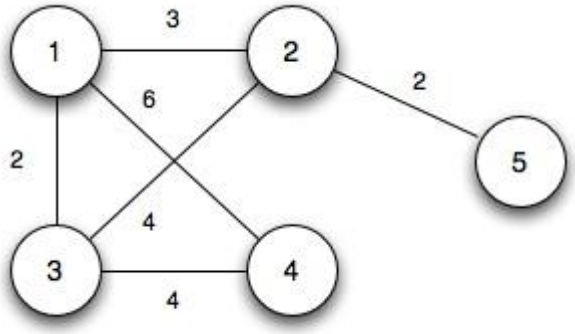
LAB-5	TREE TRAVESRSALS

Perform various tree traversal algorithms for a given tree.



LAB-6	MINIMUM COST SPANNING TREE

Find Minimum Cost Spanning Tree of a given undirected graph using Prim's algorithm.



References

1. Levitin A, "Introduction to the Design And Analysis of Algorithms", Pearson Education, 2008.
2. Goodrich M.T.,R Tomassia, "Algorithm Design foundations Analysis and Internet Examples", John Wileyn and Sons, 2006.

WEEK-1

QUICK SORT

1.1 OBJECTIVE:

Sort a given set of elements using the Quick sort method and determine the time required to sort the elements. Repeat the experiment for different values of n, the number of elements in the list to be sorted and plot a graph of the time taken versus n. The elements can be read from a file or can be generated using the random number generator.

1.2 RESOURCES:

Dev C++ or any preferred compiler

1.3 PROGRAM LOGIC:

QuickSort is a Divide and Conquer algorithm. It picks an element as pivot and partitions the given array around the picked pivot.

There are many different versions of QuickSort that pick pivot in different ways.

1. Always pick first element as pivot.
2. Always pick last element as pivot (implemented below)
3. Pick a random element as pivot.
4. Pick median as pivot.

The key process in QuickSort is partition. Target of partitions is, given an array and an element x of array as pivot, put x at its correct position in sorted array and put all smaller elements (smaller than x) before x, and put all greater elements (greater than x) after x.

1.4 PROCEDURE:

1. Write a program, save it and Compile then Execute

1.5 EAMPLE OF A SOURCE CODE:

NB: You free to modify to your preferred language

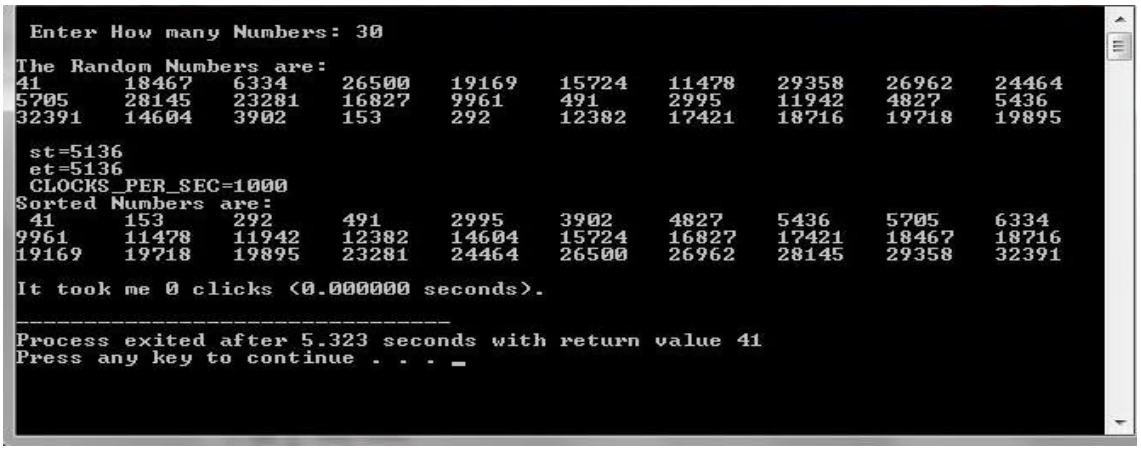
include <stdio.h>

```
include <time.h>

voidExch(int *p, int
*q){ int temp = *p;
*p = *q;
*q = temp;

} voidQuickSort(int a[], int
low, int high){

int i, j, key, k;
if(low>=high)

return;

key=low;

i=low+1; j=high;
while(i<=j){ while ( a[i]
<= a[key] )

i=i+1;

while ( a[j] >
a[key] ) j=j -1;
if(i<j)

Exch(&a[i], &a[j]);

}

Exch(&a[j], &a[key]);

QuickSort(a, low, j-1);

QuickSort(a, j+1, high);

} void main(){ int n,
a[1000],k;

clock_tst,et; double ts; clrscr();
printf("\n Enter How many
Numbers: "); scanf("%d", &n);
```

```
        printf("\nThe Random Numbers are:\n"); for(k=1; k<=n; k++){
        a[k]=rand(); printf("%d\t",a[k]);
        }

        st=clock();

        QuickSort(a, 1, n);

       et=clock();

        ts=(double)(et-st)/CLOCKS
_PER_SEC; printf("\nSorted Numbers are:
\n ");  for(k=1; k<=n; k++)
printf("%d\t",                    a[k]);
        printf("\nThe time taken is
%e",ts);
}
```

1.6 INPUT/ OUTPUT

E.g. Screen like this



1.7 SELF QIUZ

1. What is the average case time complexity of quick sort.
2. Explain if it is divide and conquer.
3. Define in place sorting algorithm.
4. List different ways of selecting pivot element.