1) Medplus" is a medical store which needs to maintain medicine detail like medicine ID. Each medicine is identified by the medicine ID. Design a program in C to read the medicine provided in the prescription and search the medicine based on its ID by using linear search method and display the relevant message whether the medicine is available or not in the store. Determine the time required to search for medicine. Repeat the experiment for different values of n and plot a graph of the time taken versus n.(n=no of elements).

→

```
#include<stdio.h>
#include<time.h>
#include<stdlib.h>
int main()
 {
       int i,a[10],key;
        int n;
       double clk;
      clock_t starttime, endtime;
                printf("Enter the number of medicine types\n");
                 scanf("%d",&n);
            //try with calling function linear(a,n)
                  for(i=0;i<n;i++)
                     a[i]=rand()%100;     //to get 2 digit numbers
                printf("\nThe medicine IDs are: \n");
                for(i=0;i<n;i++)
                   printf("%d\t",a[i]);
               printf("\nEnter the medicine ID to search\n");
                  scanf("%d",&key);
                 starttime=clock();
                  for (i=0;i<n;i++)
                  {
                        if (a[i]==key)     /* if required element found */
                        {
                          printf("Medicine ID %d is present at location %d\n", key, i+1);
                           endtime=clock();
                           clk=(double)(endtime-starttime)/CLOCKS_PER_SEC;
                          printf("Time taken to search is %f sec\n",clk);
                          return 0;
                         }
                    }
                 printf("not found\n");
            }
```

2)" Cineplix" a multiplex theatre needs to maintain the details of the seat no in ascending order and display the same to the user. Design and develop a program in C to sort the seat numbers by using bubble sort algorithm, Determine the time required to sort the seat

numbers. Repeat the experiment for different values of n and plot a graph of the time taken versus n.(n=no of elements).

→

```
//bubble sort
#include<stdio.h>
#include<time.h>
 main()
{
    int i,n,temp,j,a[10],key;
   double clk;
   clock_t starttime,endtime;
   printf("Enter the seat capacity of theatre\n");
   scanf("%d",&n);
   for(i=0;i<n;i++)
        a[i]=rand()%100;
    printf("The seat # are \n):
   for(i=0;i<n;i++)
      printf("%d\n", a[i]);
       starttime=clock();
  for(i=0;i<n-1;i++)
    {
        for(j=0;j<n-i-1;j++)
         {
           if(a[j]>a[j+1])
           {
                 temp=a[j];
                 a[j]=a[j+1];
                 a[j+1]=temp;
           }
       }
  }


   endtime=clock();
  clk=(double)(endtime-starttime)/CLOCKS_PER_SEC;
  printf("\n In ascending order of seat after bubble sort: \n");
   for(i=0;i<n;i++)
   printf("\t%d",a[i]);
  printf("%f\n",clk);
  }
```

3) Cox and kings" a tourism company needs a better way to travel, in particular it should be easy to plan an optimal route through multiple destinations. Design and develop a program in C for traveling salesman problem and repeat the experiment for different values of n and plot a graph of the time taken versus n (n=no of cities).

→

```
//TSP
```

PROGRAM

```
#include<stdio.h>
#include<time.h>
int a[10][10], visited[10],n,cost=0;
mincost(int city)
{
   int i,ncity;
   visited[city]=1;
   printf("%d -->",city+1);
   ncity=least(city);
   if(ncity==999)
   {
      ncity=0;
      printf("%d",ncity+1);
      cost+=a[city][ncity];
      return;
   }
   mincost(ncity);
}

int least(int c)
{
   int i,nc=999;
   int min=999;
   for(i=0;i<n;i++)
   {
      if((a[c][i]!=0)&&(visited[i]==0))
         if(a[c][i] < min)
         {
            min=a[c][i];
            nc=i;
         }
   }
   if(min!=999)
      cost+=min;
   return nc;
}

main()
{
      int i,j;
      double clk;
      clock_t starttime,endtime;
   printf("Enter No. of Cities: ");
   scanf("%d",&n);
   printf("\n Enter Cost Matrix\n");
```

```
    for(i=0;i < n;i++)
    {
       printf("\nEnter Elements of Row # : %d\n",i+1);
       for( j=0;j<n;j++)
          scanf("%d",&a[i][j]);
       visited[i]=0;
    }

    printf("\n\nThe cost list is:\n\n");
    for( i=0;i < n;i++)
    {
       printf("\n\n");
       for(j=0;j < n;j++)
          printf("\t%d",a[i][j]);
    }
       printf("\n\nThe Path is:\n\n");
          starttime=clock();
          mincost(0);
          endtime=clock();
          clk=(double)(endtime-starttime)/CLOCKS_PER_SEC;
    printf("\n\nMinimum cost:%d\n",cost);
       printf("\nThe run time is %f\n",clk);
}
```

Expected output:

```
Enter No. of Cities: 4
Enter Cost Matrix
0 2 5 7
2 0 8 3
5 8 0 1
7 3 1 0
The cost list is:
0 2 5 7
2 0 8 3
5 8 0 1
7 3 1 0
The Path is: 1 2 4 3 1
Minimum cost: 11
The run time is 0.989011
```

---

4) Design and develop a program in C to print all the nodes reachable from a given starting node in a digraph by using BFS method. Repeat the experiment for different values of n and plot a graph of the time taken versus n(n=no of nodes).

→

//BFS

PROGRAM

```
#include<stdio.h>
#include<time.h>
void bfs (int a[10][10],int n, int source)
{
   int visited[10],q[10],i,j,f=0,r=-1;

   for(i=0;i<n;i++)
      visited[i]=0;


   q[++r]=source;
   visited[source]=1;
   while(f<=r){
      j=q[f++];
      for(i=0;i<n;i++)
         if(a[j][i]==1&&visited[i]==0)
         {
         q[++r]=i;
         printf("The BFS Traversal is:%d-->%d\n",j,i);
         printf("%d is reachable from source %d \n",i,source);
         visited[i]=1;
         }
   }
}


void main()
{
   int a[10][10],n,i,j,s;
   double clk;
   clock_t starttime,endtime;
   printf("Enter the number of vertices: ");
    scanf("%d",&n);
   printf("\nEnter the matrix represention:\n");
    for(i=0;i<n;i++)
    for(j=0;j<n;j++)
     scanf("%d",&a[i][j]);
    printf("\nEnter the source vertix: [ give 0 to n-1 ]");
    scanf("%d",&s);
   starttime=clock();
    bfs(a,n,s);
   endtime=clock();
   clk=(double)(endtime-starttime)/CLOCKS_PER_SEC;
```

```
    printf("\nThe run time is %f\n",clk);
}
```

Expected output: //try with different inputs

Enter the number of vertices: 5

Enter the matrix represention:
0 1 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 1 0 0
1 0 0 1 0

Enter the source vertix: 4
The BFS Traversal is:4-->0
0 is reachable from source 4
The BFS Traversal is:4-->3
3 is reachable from source 4
The BFS Traversal is:0-->1
1 is reachable from source 4
The BFS Traversal is:3-->2
2 is reachable from source 4

The run time is 0.000076

---

5) Design and develop a program in C to print all the nodes reachable from a given starting node in a digraph by using DFS method. Repeat the experiment for different values of n and plot a graph of the time taken versus n(n=no of nodes).

→

```
//5 DFS
#include<stdio.h>
#include<time.h>
void DFS(int);
int G[10][10],visited[10],n;    //n is no of vertices and graph is sorted in array G[10][10]
void main()
{
```

```
   int i,j, source;
   clock_t starttime,endtime;
   printf("Enter number of vertices:");
         scanf("%d",&n);
   //read the adjecency matrix
         printf("\nEnter adjecency matrix of the graph:");
                  for(i=0;i<n;i++)
                     for(j=0;j<n;j++)
                              scanf("%d",&G[i][j]);
   //visited is initialized to zero
  for(i=0;i<n;i++)
      visited[i]=0;
   printf("Enter source vertex: between 0 to n-1\n");
   scanf("%d",&source);
   starttime=clock();
   DFS(source);
   endtime=clock();
   printf("\nTime for execution= %f seconds\n", (float)(endtime-
starttime)/CLOCKS_PER_SEC);
   printf("reachable vertices from source %d are :\n", source);
   for(i=0;i<n;i++)
         if(i!=source&& visited[i])
         printf("%d\n",i);
}

void DFS(int i)
{
       int j;
         //printf("\n %d",i); //reachable
         visited[i]=1;
         for(j=0;j<n;j++)
      if(!visited[j]&&G[i][j]==1)
            {
```

```
                printf("%d--->%d\n",i,j);  //dfs traversal
        DFS(j);
          }
 }
```

6) **"Digishop" a online shopping website needs to keep track of the product availability based on the product ID. Design a program in C to read the product ID provided by the customer and search for it's availability by using Binary search method and display the relevant message whether the product is in stock or not. Determine the time required to search for the product. Repeat the experiment for different values of n and plot a graph of the time taken versus n. (n=no of elements).**

→

```
#include<stdio.h>
#include<time.h>
#include<stdlib.h>


int bottom,top,mid,n;
void bubble_sort(int a[]);
int binary_search(int a[], int *key);

void main()
{
    int i,a[50],key;
    int result;
  double clk;
    clock_t starttime,endtime;
     printf("Enter the number of products\n");
    scanf("%d",&n);
    for(i=0;i<n;i++)
          a[i]=rand()%1000;

  printf("\nThe Product IDs are: \n");
  for(i=0;i<n;i++)
        printf("%d\t",a[i]);

  bubble_sort(a);

  printf("\n\nSorted Product IDs  are: \n");
  for(i=0;i<n;i++)
      printf("%d\t",a[i]);


   printf("\n\nEnter the ISBN to be searched: \n");
    scanf("%d",&key);
```

```
        starttime=clock();
        result= binary_search(a,&key);
        endtime=clock();

        if (result == 1)
          {
            printf("\nProduct found!!\n");
               printf("\n\n Product ID %d found in position: %d\n", key, mid + 1);
        }
         else
            printf("\n  Out of stock\nproduct ID %d not found\n", key);

         clk=(double)(endtime-starttime)/CLOCKS_PER_SEC;
        printf("\nTime taken for searching %f seconds\n",clk);
}


int binary_search(int a[], int *key)
{
   bottom = 0;
   top = n-1;
    do
    {
        mid = (bottom + top) / 2;
         if (*key < a[mid])
            top = mid - 1;
        else if (*key > a[mid])
            bottom = mid + 1;


       }while (*key != a[mid] && bottom <= top);

    if(*key == a[mid])
        return 1;
    else
      return 0;
}


void bubble_sort(int a[])
{
   int i,j,temp;
   for(i=0;i<n;i++)
   {
         for(j=0;j<n-i-1;j++)
           {
             if(a[j]>a[j+1])
```

```
                {
                    temp=a[j];
                    a[j]=a[j+1];
                    a[j+1]=temp;
                }
            }
        }
    }
```

7）“Aircel” a mobile network company need to maintain the telephone numbers of its customer, in order to call and inform them about the new year offer. They have to sort the contact numbers in ascending order to keep track of the customers whom they called. , design and develop a program in C to sort the phone numbers by using insertion sort algorithm, Input should be generated randomly. Determine the time required to sort the elements. Repeat the experiment for different values of n and plot a graph of the time taken versus n. (n=no of elements).

→

```
/* Insertion sort ascending order */


#include<stdio.h>
#include<time.h>
#include<stdlib.h>


void insertionsort(int a[], int);


int main()
{
        int n, num[50], i, t;
        clock_t start,end;
        double time;
        printf("Enter number of customers\n");
        scanf("%d", &n);
        for(i = 0; i < n; i++)
                a[i]=rand()%100;
        printf("Customers' contact numbers are:\n");
        for(i = 0; i < n; i++)
                printf("%d\t", num[i]);
```

```
        start=clock();
        insertionsort(num,n);
        end=clock();
        printf("\nSorted contact list in ascending order:\n");
         for(i = 0; i < n; i++)
            printf("%d\t", num[i]);
         time=(double)(end-start)/CLOCKS_PER_SEC;
        printf("\nTime taken for execution=%lf\n",time);
        return 0;
}

 void insertionsort(int num[], int n)
{
        int i,t,j,flag=0;
        for(i=0;i<n;i++)
        {
                t=a[i];
                for(j=i-1;j>=0;j--)
                {
                        if(num[j]>t)
                        {
                                num[j+1]=num[j];
                                flag=1;
                 }
                 else
               break;
                }
                if(flag)
                num[j+1]=t;
        }
 }
```

8) “Deloit”, a software company needs to maintain its employee details like employee id, name, address in a record, design and develop a program in C to sort the employee records

based on their employee ID by using merge sort algorithm, employee ID should be generated randomly. Determine the time required to sort the elements. Repeat the experiment for different values of n and plot a graph of the time taken versus n. (n=no of elements).

→ **//MERGE sort**

PROGRAM

```
#include<stdio.h>
#include<time.h>
#include<stdlib.h>


int a[15];
void merge(int,int,int);
void merge_sort(int low,int high)
{
    int mid;
    if(low<high)
    {
      mid=(low+high)/2;
      merge_sort(low,mid);
      merge_sort(mid+1,high);
      merge(low,mid,high);
    }
}
void merge(int low,int mid,int high)
{
    int h,i,j,b[15],k;
    h=low;
    i=low;
    j=mid+1;
    while((h<=mid)&&(j<=high))
   {
        if(a[h]<=a[j])
      {
            b[i]=a[h];
            h++;
      }
      else
      {
         b[i]=a[j];
          j++;
       }
       i++;
    }
    if(h>mid)
    {
```

```
        for(k=j;k<=high;k++)
        {
            b[i]=a[k];
            i++;
        }
    }
     else
     {
        for(k=h;k<=mid;k++)
        {
            b[i]=a[k];
            i++;
        }
     }
    for(k=low;k<=high;k++) a[k]=b[k];
}


int main()
{
    int n,i;
    double clk;
    clock_t starttime,endtime;
    printf("MERGE SORT\n");
    printf("Enter the number of employee records:\n ");
    scanf("%d",&n);


     for(i=0;i<n;i++)
            a[i]=rand()%100; //to get 2 digit number


   printf("The Employee IDs are:\n");
   for(i=0;i<n;i++)
       printf("%d\t",a[i]);


    starttime=clock();
    merge_sort(0,n-1);
    endtime=clock();


    clk=(double)(endtime-starttime)/CLOCKS_PER_SEC;
    printf("\nEmployee IDs in sorted order:\n");
    for(i=0;i<n;i++)
     {
         printf("%d\t",a[i]);
    }
```

```
    printf("\nThe run time is %f seconds. \n",clk);
}
```

Expected output:
Enter the number of employee records: 5

The Employee IDs are:
62 34 50 28 16

Employee IDs in sorted order:
16 28 34 50 62

The time taken is 0.989011

9) Assume that NMIT college needs to maintain the student details like USN, name, and contact details in a record. USN should be generated randomly. Design and develop a program in C to sort the records based on USN by using quick sort algorithm, Determine the time required to sort the roll numbers. Repeat the experiment for different values of n and plot a graph of the time taken versus n. (n=no of elements).

→

```
//PART A 9th pgm QUICK SORT
//try for n=5,15,25..

#include<stdio.h>
#include<time.h>
#include<stdlib.h>

int partition(int a[],int low,int high)
{
    int i,j,temp,pivot;
    pivot=a[low];
    i=low+1;
    j=high;
    while(1)
    {
        while(i<high&&a[i]<=pivot)
```

```
                        i++;
                while(a[j]>pivot)
                        j--;
                if(i<j)
                        a[j]=a[i]+a[j]-(a[i]=a[j]);
                                //swap a[i] and a[j]
                else
                {
                        a[j]=a[low]+a[j]-(a[low]=a[j]);
                                        //swap a[low] and a[j]
                        return j;
                }
        }
}

void quick_sort(int a[],int low,int high)
{
        int j;
        if(low<high)
        {
                j=partition(a,low,high);
                quick_sort(a,low,j-1);
                quick_sort(a,j+1,high);
        }
}

void main()
{
        int i, n,a[30],ticks;
    double clk;
    clock_t starttime,endtime;
        printf("Enter the number of students: \n");
        scanf("%d",&n);
```

```
    for(i=0;i<n;i++)
       a[i]=rand()%1000; //3 digit #

        printf("The students roll # are:\n");
        for(i=0;i<n;i++)
       printf("%d\t",a[i]);

    starttime=clock();
        quick_sort(a,0,n-1);
    endtime=clock();

        ticks=abs(starttime-endtime);
    clk=(double)(ticks)/CLOCKS_PER_SEC;

        printf("\nSorted roll numbers are: \n");
        for(i=0;i<n;i++)
                printf("%d\t",a[i]);

        printf("\nValue of CLOCKS_PER_SEC is %ld\n",CLOCKS_PER_SEC);
    printf("\nTotal elapsed clock tickes=%d, The run time is %f seconds\n",ticks, clk);
}
```