



SYIT DATA Structure Journal 2021

Data Structures (University of Mumbai)



Scan to open on Studocu

SYIT DATA STRUCTURE JOURNAL 2021-2022

NAME: SHOBHA CHOUHAN

Roll NO. 37

SUBJECT: DATA STRUCTURE

INDEX

SR NO.	PRACTICAL	DATE	SIGN
1A	Write a program to store the elements in 1-D array and perform the operations like searching, sorting and reversing the elements. [Menu Driven]		
1B	Read the two arrays from the user and merge them and display the elements in sorted order.[Menu Driven]		
1C	Write a program to perform the Matrix addition, Multiplication and Transpose Operation. [Menu Driven]		
2A	Write a program to create a single linked list and display the node elements in reverse order.		
2B	Write a program to search the elements in the linked list and display the same		
2C	Write a program to create double linked list and sort the elements in the linked list.		
3A	Write a program to implement the concept of Stack with Push, Pop, Display and Exit operations.		
3B	Write a program to convert an infix expression to postfix and prefix conversion.		
3C	Write a program to implement Tower of Hanoi problem.		
4A	Write a program to implement the concept of Queue with Insert, Delete, Display and Exit operations.		
4B	Write a program to implement the concept of Circular Queue		
4C	Write a program to implement the concept of Deque.		
5A	Write a program to implement bubble sort.		
5B	Write a program to implement selection sort.		
5C	Write a program to implement insertion sort.		
6A	Write a program to implement merge sort.		
6B	Write a program to search the element using sequential search		
6C	Write a program to search the element using binary search.		
7A	Write a program to create the tree and display the elements.		
7B	Write a program to construct the binary tree.		

7C	Write a program for inorder, postorder and preorder traversal of tree		
8A	Write a program to insert the element into maximum heap.		
8B	Write a program to insert the element into minimum heap.		
9A	Write a program to implement the collision technique.		
9B	Write a program to implement the concept of linear probing.		
10A	Write a program to generate the adjacency matrix.		
10B	Write a program for shortest path diagram.		

Implement the following:

PRACTICAL 1A

Write a program to store the elements in 1-D array and perform the operations like searching, sorting and reversing the elements. [Menu Driven]

INPUT:

```
#include<stdio.h>

#include<conio.h>

int main()

{

int a[100],n, i ,l,j,x,ele,ch,flag=0;

clrscr();

printf("\n Enter the size of the array \t");

scanf("%d",&n);

printf("\n Enter the elements of the array: \n");

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

{

scanf("%d",&a[i]);

}

do

{

clrscr();

printf("\n 1-Searching \n 2-Sorting the array \n 3-Reverse the elements \n 4-Exit");

printf("\n Enter your choice\t");

scanf("%d",&ch);

switch(ch)

{

case 1:

printf("\n Enter the element to be searched \t");

scanf("%d",&ele);

for(i=0;i<n;i++)
```

```
{
if(a[i]==ele)
{ x=
l;
flag=1;
break;
}
}
if(flag==0)
{
printf("\n Element not found!");
}
else
{
printf("\n Element found at position %d",x+1);
flag=0;
}
getch();
break;
case 2:
for(i=0;i<n;i++)
{
for(j=0;j<n-1;j++)
{
if(a[j]>a[j+1])
{
x=a[j];
a[j]=a[j+1];
a[j+1]=x;
}
}
}
}
```

```
printf("\n After sorting\n");
for(i=0;i<n;i++)
{
printf("%d \t ",a[i]);
}

getch();

break;

case 3:

x=n/2;

j=n-1;

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

{

ele=a[i];

a[i]=a[j];

a[j]=ele;

j--;

}

printf("\n Reversed array is \n");

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

{

printf("%d \t ",a[i]);

}

getch();

break;

case 4:

exit(0);

break;

default:

printf("\n INVALID INPUT!");

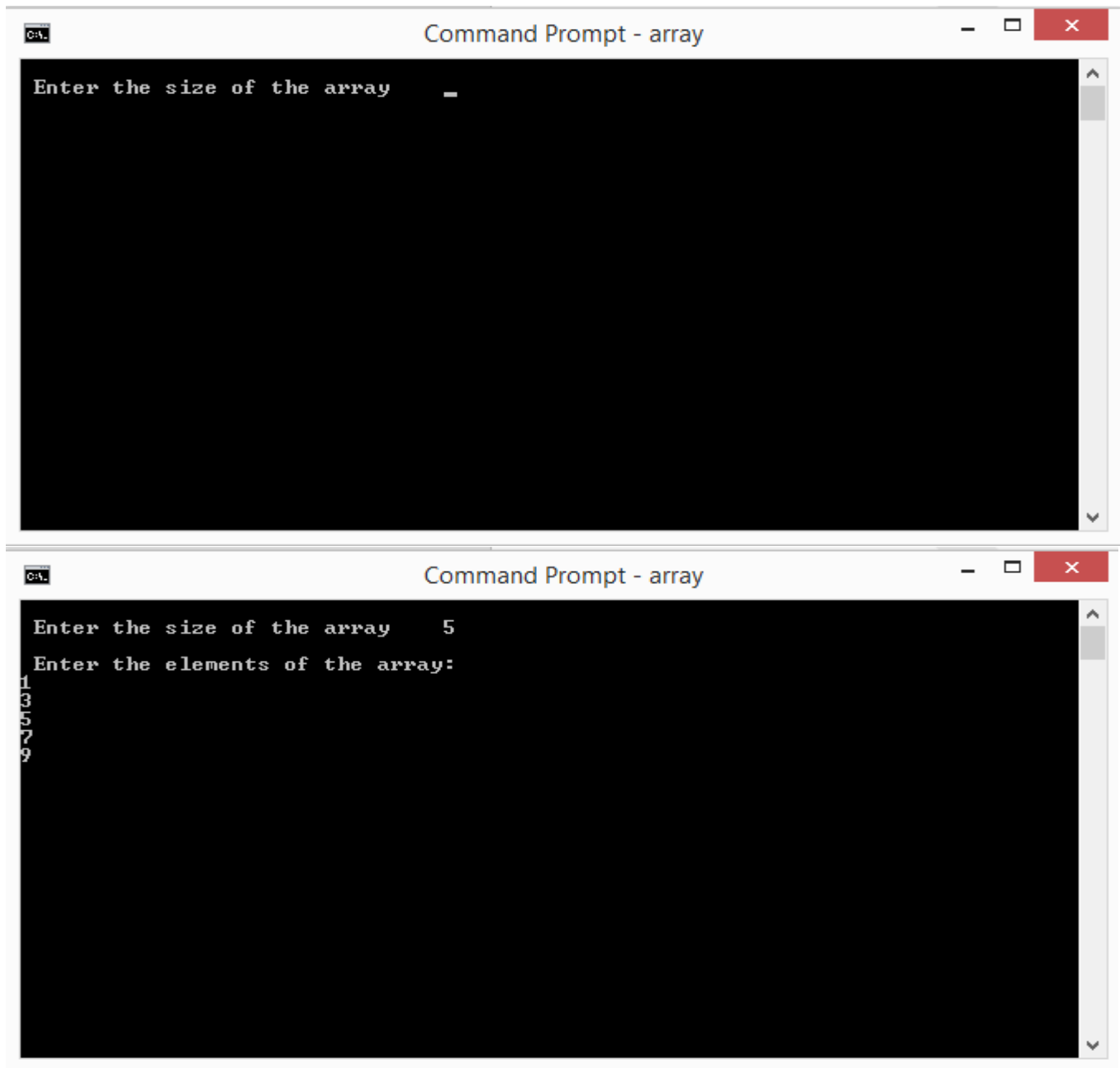
}

}while(ch!=4);getch();

return 0;
```

}

OUTPUT:



```
Command Prompt - array
Enter the size of the array _

Command Prompt - array
Enter the size of the array 5
Enter the elements of the array:
1
3
5
7
9
```



```
CA: Command Prompt - array
1-Searching
2-Sorting the array
3-Reverse the elements
4-Exit
Enter your choice      1
Enter the element to be searched      3
Element found at position 1997081957
```

```
CA: Command Prompt - array
1-Searching
2-Sorting the array
3-Reverse the elements
4-Exit
Enter your choice      2
After sorting
1      3      5      7      9
```

```
CA: Command Prompt - array
1-Searching
2-Sorting the array
3-Reverse the elements
4-Exit
Enter your choice      3
Reversed array is
9      7      5      3      1
```

PRACTICAL 1B

Read the two arrays from the user and merge them and display the elements in sorted order.[Menu Driven]

INPUT:

```
#include<stdio.h>
#include<conio.h>

intmain()
{
int arr1[20],arr2[20],arr3[40];
inti,j,k,size1,size2,temp;

printf("Enter the array size of array 1 :");
scanf("%d",&size1);
printf("Enter the element in arra 1 :\n");
for(i=0;i<size1;i++)
{
scanf("%d",&arr1[i]);
}

printf("Enter the size of array 2 : ");
scanf("%d",&size2);
printf("Enter the element in array2 : \n");
for(j=0;j<size2;j++)
{
scanf("%d",&arr2[j]);
}

for(i=0;i<size1;i++)
{
for(j=0;j<size1-i-1;j++)
{
if(arr1[j]>arr1[j+1])
{
temp=arr1[j];
```

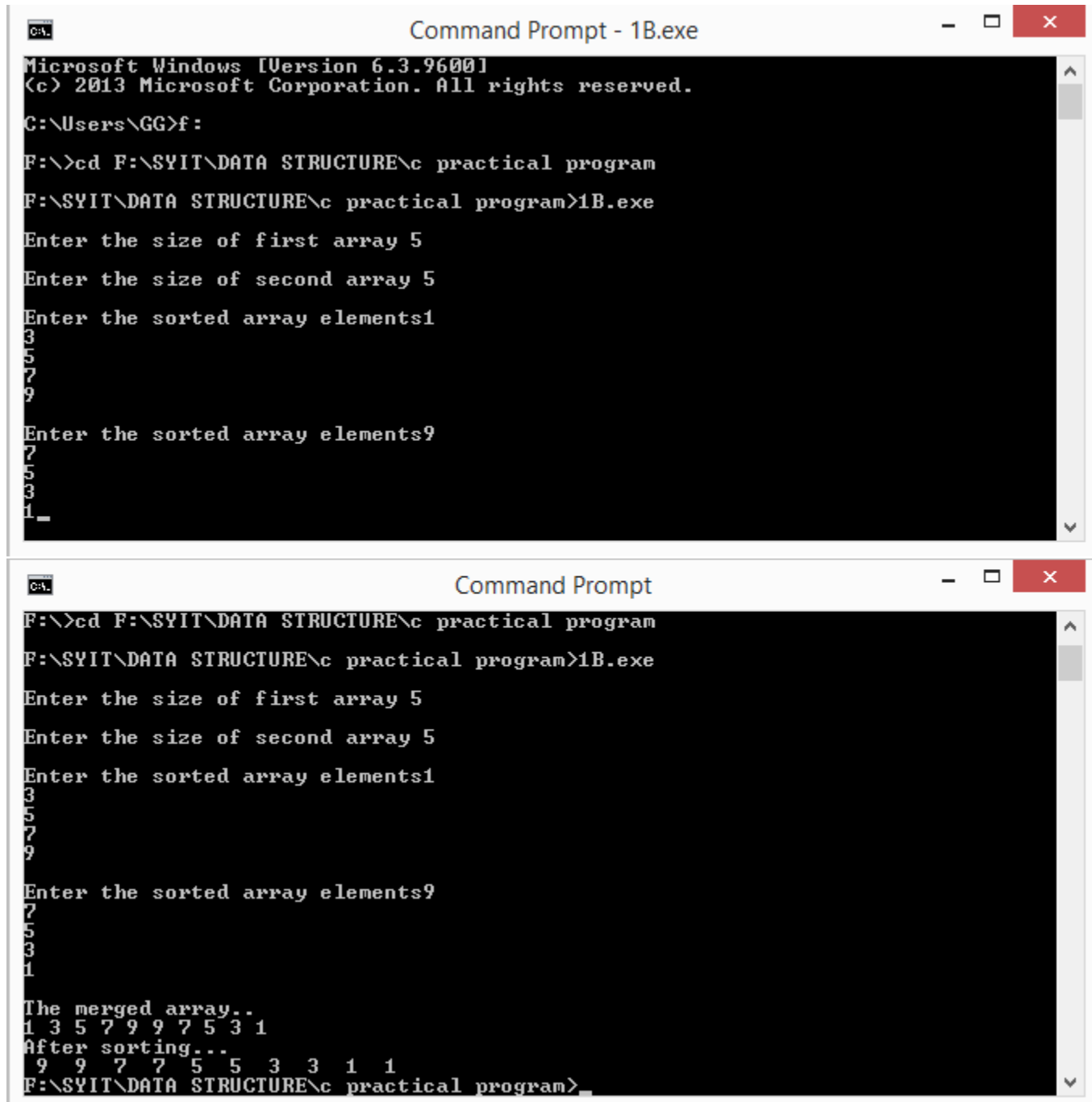
```

arr1[j]=arr1[j+1];
arr1[j+1]=temp;
}
}
}
for(i=0;i<size2;i++)
{
for(j=0;j<size2-i-1;j++)
{
if(arr2[j]>arr2[j+1])
{
temp=arr2[j];
arr2[j]=arr2[j+1];
arr2[j+1]=temp;
}
}
}
/* printf("Sorting array 1\n");
for(i=0;i<size1;i++)
{
printf("%d \n",arr1[i]);
}
printf("Sorting array 2\n");
for(i=0;i<size2;i++)
{
printf("%d \n",arr2[i]);
}*/
i=0;
j=0;
k=0;
while(i<size1 && j<size2)
{

```

```
if(arr1[i]<arr2[j])
{
arr3[k]=arr1[i];
i++;
k++;
}
else
{
arr3[k]=arr2[j];
j++;
k++;
}
}
while(i<size1)
{
arr3[k]=arr1[i];
i++;
k++;
}
while(j<size2)
{
arr3[k]=arr2[j];
j++;
k++;
}
printf("merged array \n");
for(k=0;k<size1+size2;k++)
printf("%d \n",arr3[k]);
getch();
return 0;
}
```

OUTPUT:



```
Command Prompt - 1B.exe
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\GG>f:

F:\>cd F:\SYIT\DATA STRUCTURE\c practical program
F:\SYIT\DATA STRUCTURE\c practical program>1B.exe
Enter the size of first array 5
Enter the size of second array 5
Enter the sorted array elements1
3
5
7
9
Enter the sorted array elements9
7
5
3
1_

Command Prompt
F:\>cd F:\SYIT\DATA STRUCTURE\c practical program
F:\SYIT\DATA STRUCTURE\c practical program>1B.exe
Enter the size of first array 5
Enter the size of second array 5
Enter the sorted array elements1
3
5
7
9
Enter the sorted array elements9
7
5
3
1
The merged array..
1 3 5 7 9 9 7 5 3 1
After sorting...
9 9 7 7 5 5 3 3 1 1
F:\SYIT\DATA STRUCTURE\c practical program>_
```

PRACTICAL 1C

Write a program to perform the Matrix addition, Multiplication and Transpose Operation. [Menu Driven]

INPUT:

```
#include<stdio.h>

#include<conio.h>

int main()

{

int a[3][3],b[3][3],c[3][3],i,j,k,ch;

clrscr();

do

{ printf("\n1-Read matrix A\n2-Read Matrix B\n3-addition \n4-multiplication\n5transpose \n6-exit \n");
printf("\nEnter the choice \t");

scanf("%d",&ch);

switch(ch)

{

case 1:

{   printf("\nEnter the element of matrix A\n");

for(i=0;i<3;i++)

{   for(j=0;j<3;j++)

{   scanf("%d",&a[i][j]);   }

}

break;

}

case 2:

{   printf("\nEnter the element of matrix B\n");

for(i=0;i<3;i++)

{   for(j=0;j<3;j++)

{   scanf("%d",&b[i][j]);   }

}

break;   }
```

```

case 3:
{   printf("\nMatrix addition");
    for(i=0;i<3;i++)
{   for(j=0;j<3;j++)
{   c[i][j]=a[i][j]+b[i][j];   }
    }
printf("\nResultant Addition matrix is\n");
for(i=0;i<3;i++)
{   for(j=0;j<3;j++)
{   printf("%d\t",c[i][j]);
    }
    printf("\n");
}
getch();
break;
}

case 4:
{   for(i=0;i<3;i++)
{   for(j=0;j<3;j++)
{   c[i][j]=0;   }
    }
for(i=0;i<3;i++)
{   for(j=0;j<3;j++)
{   for(k=0;k<3;k++)
{   c[i][j]=c[i][j]+(a[i][k]*b[k][j]);   }
    }
}
for(i=0;i<3;i++)
{   for(j=0;j<3;j++)
{   printf("%d\t",c[i][j]);
    }   printf("\n");
}
}

```

```
getch();
    break;
} case 5:
{    printf("\nTranspose matrix\n");
    for(i=0;i<3;i++)
    {    for(j=0;j<3;j++)
    {    printf("%d\t",a[j][i]);
    }    printf("\n");
    }
getch();
    break;
}
case 6:
{
    exit(0);
    break;
}
default:
{
    printf("\nInvalid input");
}
}
} while(ch!=6);
    getch();
}
```

OUTPUT:


```
CA: Command Prompt - 1C.exe
1-Read matrix A
2-Read Matrix B
3-addition
4-multiplication
5transpose
6-exit
Enter the choice      1
Enter the element of matrix A
123
321
213
231
213
123
132
321
312
1-Read matrix A
2-Read Matrix B
3-addition
4-multiplication
```

```
CA: Command Prompt - 1C.exe
4-multiplication
5transpose
6-exit
Enter the choice      2
Enter the element of matrix B
2
3
4
5
6
7
8
5
4
1-Read matrix A
2-Read Matrix B
3-addition
4-multiplication
5transpose
6-exit
Enter the choice
```

```
Command Prompt - 1C.exe

Enter the element of matrix B
2
3
4
5
6
7
8
5
4

1-Read matrix A
2-Read Matrix B
3-addition
4-multiplication
5transpose
6-exit

Enter the choice      3

Matrix addition/nResultant Addition matrix is
125      324      217
236      219      130
140      326      316
```

```
Command Prompt - 1C.exe

2-Read Matrix B
3-addition
4-multiplication
5transpose
6-exit

Enter the choice      3

Matrix addition/nResultant Addition matrix is
125      324      217
236      219      130
140      326      316

1-Read matrix A
2-Read Matrix B
3-addition
4-multiplication
5transpose
6-exit

Enter the choice      4
3555      3360      3591
2511      2586      2907
4365      3882      4023
```

```
Command Prompt - 1C.exe

2-Read Matrix B
3-addition
4-multiplication
5transpose
6-exit

Enter the choice      4
3555      3360      3591
2511      2586      2907
4365      3882      4023

1-Read matrix A
2-Read Matrix B
3-addition
4-multiplication
5transpose
6-exit

Enter the choice      5

Transpose matrix
123      231      132
321      213      321
213      123      312
```

Implement the following for Linked List:

PRACTICAL 2A

Write a program to create a single linked list and display the node elements in reverse order.

INPUT:

```
#include<stdio.h>
#include<conio.h>
#include<malloc.h>
#include<stdlib.h>

struct node
{
    int info;
    struct node *next;
};

struct node *start=NULL;
struct node *create(struct node *start);
struct node *dispaly(struct node *start);
void reverse(struct node *start);

int main()
{
    clrscr();
    start=create(start);
    start=dispaly(start);
    printf("\n");
    printf("Reverse \t");
    reverse(start);
    getch ();
    return 0;
}

struct node *create(struct node *start)
```

```

{

    struct node *new_node=NULL,*temp=NULL;

    int val;

    printf("Enter -1 value to exit list.\n");

    printf("Enter the value : \n");

    scanf("%d",&val);

    while(val!=-1)

    {

        new_node=(struct node*)malloc(sizeof(struct node));

        new_node->info=val;

        if(start==NULL)

        {

            start=new_node;

            new_node->next=NULL;

        }

        else

        {

            temp=start;

            while(temp->next!=NULL)

            {

                temp=temp->next;

            }

            temp->next=new_node;

            new_node->next=NULL;

        }

        printf("Enter the value : \n");

        scanf("%d",&val);

    }

    printf("List is successfully created.\n");

    return start;

}

struct node *dispaly(struct node *start)

```

```

{
    struct node *temp=NULL;
    temp=start;
    printf("List is :\n");
    while(temp!=NULL)
    {
        printf("%d \t",temp->info);
        temp=temp->next;
    }
    return start;
}

void reverse(struct node *start)
{
    struct node *prev=NULL;
    struct node *current=start;
    struct node *next_node;
    while(current!=NULL)
    {
        next_node=current->next;
        current->next=prev;
        prev=current;
        current=next_node;
    }
    start=prev;
    start=display(start);
}

```

OUTPUT:

```
Command Prompt - 2A.exe

Enter -1 value to exit list.
Enter the value :
12345
Enter the value :
54321
Enter the value :
12345
Enter the value :
54321
Enter the value :
12345
Enter the value :
54321
Enter the value :
12345
Enter the value :
54321
Enter the value :
12345
Enter the value :
54321
Enter the value :
12345
Enter the value :
54321
Enter the value :
12345
Enter the value :
54321
```

```
Command Prompt - 2A.exe

Enter the value :
12345
Enter the value :
54321
Enter the value :
12345
Enter the value :
54321
Enter the value :
12345
Enter the value :
54321
Enter the value :
12345
Enter the value :
54321
Enter the value :
12345
Enter the value :
54321
Enter the value :
-1
List is successfully created.
List is :
12345 54321 12345 54321 12345 54321 12345 54321 12345 54321
12345 54321
Reverse List is :
54321 12345 54321 12345 54321 12345 54321 12345 54321 12345
54321 12345
```

PRACTICAL 2B

Write a program to search the elements in the linked list and display the same.

INPUT:

```
#include<stdio.h>

#include<conio.h>

#include<malloc.h>

#include<stdlib.h>

struct node
{
    int info;
    struct node *next;
};

struct node *start=NULL;
struct node *create(struct node *start);
struct node *dispaly(struct node *start);
struct node *search(struct node *start);
int main()
{
    start=create(start);
    start=dispaly(start);
    printf("\n");
    start=search(start);
    getch ();
    return 0;
}

struct node *create(struct node *start)
{
    struct node *new_node=NULL,*temp=NULL;

    int val;

    printf("Enter -1 value to exit list.\n");

    printf("Enter the value : \n");
```

```

scanf("%d",&val);
while(val!=-1)
{
new_node=(struct node*)malloc(sizeof(struct node));
new_node->info=val;
if(start==NULL)
{
start=new_node;
new_node->next=NULL;
}
else
{
temp=start;
while(temp->next!=NULL)
{
temp=temp->next;
}
temp->next=new_node;
new_node->next=NULL;
}
printf("Enter the value : \n");
scanf("%d",&val);
}
printf("List is successfully created.\n");
return start;
}
struct node *dispaly(struct node *start)
{
struct node *temp=NULL;
temp=start;
printf("List is :\n");
while(temp!=NULL)

```



```

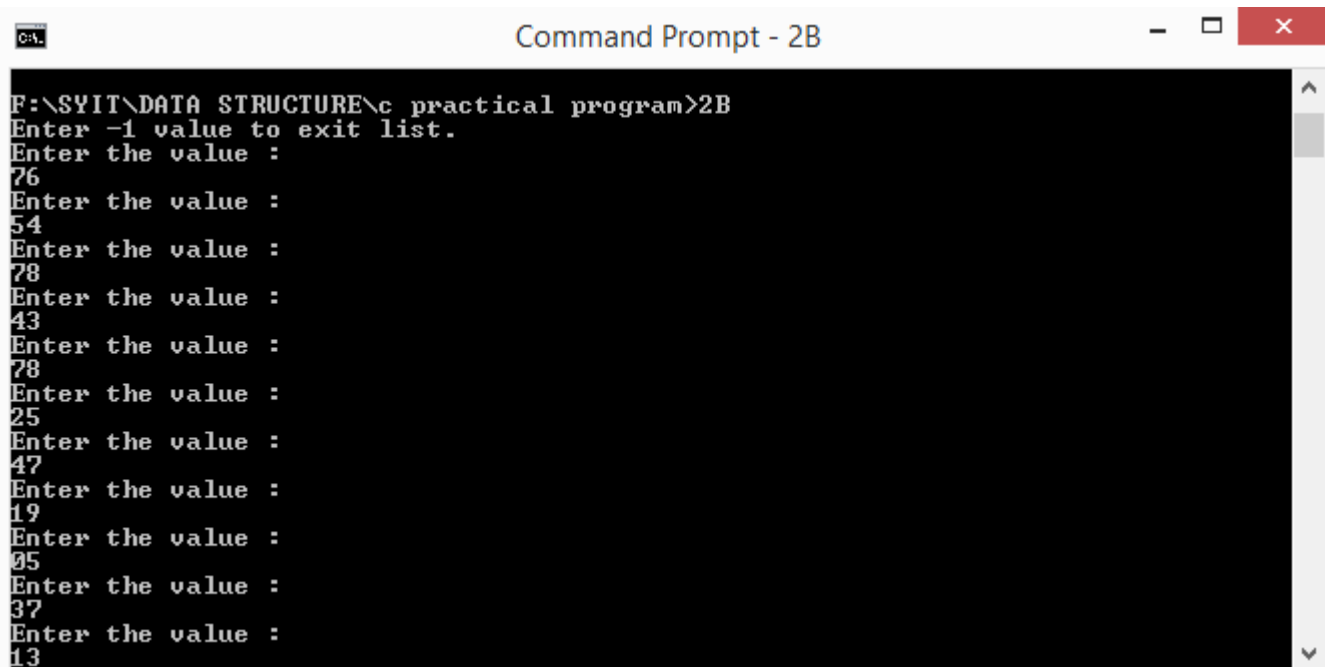
        {
            printf("%d \t",temp->info);
            temp=temp->next;
        }
        return start;
    }
    struct node *search(struct node *start)
    {
        int val,count;
        struct node *temp;
        printf("\nwhich value are you looking for?\n");
        scanf("%d",&val);
        count=1;
        temp=start;
        while(temp->info!=val && temp->next!=NULL)
        {
            temp=temp->next;
            count++;
        }
        //temp=temp->next;
        if(temp->next==NULL && temp->info!=val)
        {
            printf("value not found");
        }
        else if(temp->next==NULL && temp->info==val)
        {
            printf("value found at %d node",count);
        }
        else
        {
            printf("value found at %d node",count);
        }
    }

```

```
return start;
```

```
}
```

OUTPUT:



```
Command Prompt - 2B
F:\SYIT\DATA STRUCTURE\c practical program>2B
Enter -1 value to exit list.
Enter the value :
76
Enter the value :
54
Enter the value :
78
Enter the value :
43
Enter the value :
78
Enter the value :
25
Enter the value :
47
Enter the value :
19
Enter the value :
05
Enter the value :
37
Enter the value :
13
```



```
Command Prompt - 2B
19
Enter the value :
05
Enter the value :
37
Enter the value :
13
Enter the value :
31
Enter the value :
-1
List is successfully created.
List is :
76      54      78      43      78      25      47      19      5      37
13      31
which value are you looking for?
```

PRACTICAL 2C

Write a program to create double linked list and sort the elements in the linked list.

INPUT:

```
#include<stdio.h>

#include<conio.h>

#include<malloc.h>

#include<stdlib.h>

struct node

{int data;

    struct node *next;

    struct node *prev;

};

struct node *start=NULL;

struct node *create(struct node *start);

struct node *display(struct node *start);

struct node *sort(struct node *start);

int main()

{

    start=create(start);

    start=display(start);

    printf("\n");

    printf("sort \t");

    start=sort(start);

}

struct node *create(struct node *start)

{

    struct node *new_node=NULL,*temp=NULL,prev;

    int val;

    printf("Enter the data or enter -1 to exit:");

    scanf("%d",&val);
```

```

while(val!=-1)
{
    new_node=(struct node*)malloc(sizeof(struct node));
    new_node->data=val;
    if(start==NULL)
    {
        start=new_node;
        new_node->next=NULL;
        new_node->prev=NULL;
    }
    else
    {
        temp=start;
        while(temp->next!=NULL)
        {
            temp=temp->next;
        }
        temp->next=new_node;
        new_node->prev=NULL;
        new_node->next=NULL;
    }
    printf("enter the data or enter -1 to exit:");
    scanf("%d",&val);
}

printf("Linked list successfully created.\n");
return start;
}

struct node *display(struct node *start)
{
    struct node *temp=NULL;
    temp=start;
    printf("\nThe Linked list is:");

```

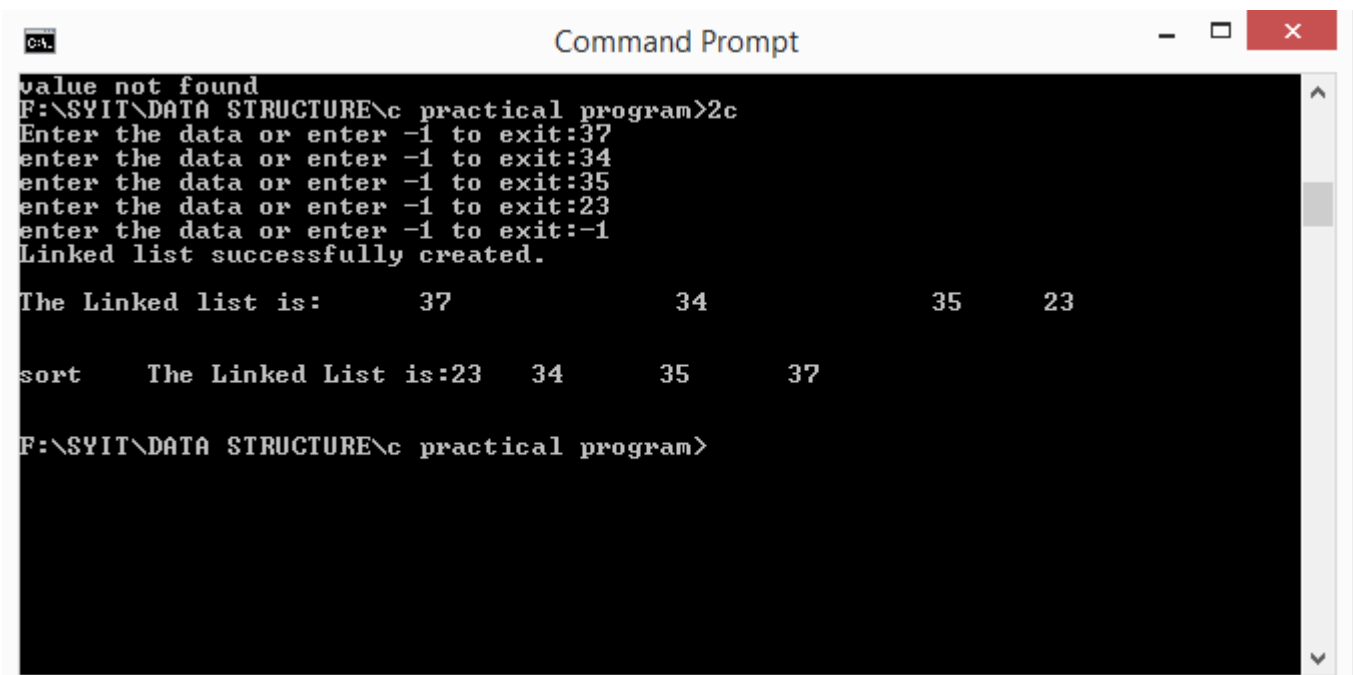
```

while(temp->next!=NULL)
{
    printf("\t %d \t",temp->data);
    temp=temp->next;
}
if(temp->next==NULL)
printf("%d \n",temp->data);
printf("\n");
return start;
}
struct node *sort(struct node*start)
{
struct node *temp1=start;
struct node *temp2,*temp;
int x;
while (temp1->next!=NULL)
{
    temp2=start;
    while(temp2->next!=NULL)
    {
        temp=temp2->next;
        if(temp2->data>temp->data)
        {
            x=temp->data;
            temp->data=temp2->data;
            temp2->data=x;
        }
        temp2=temp2->next;
    }
    temp1=temp1->next;
}
temp=start;

```

```
printf("The Linked List is:");  
while(temp->next!=NULL)  
{  
    printf("%d \t",temp->data);  
    temp=temp->next;  
}  
if(temp->next==NULL)  
printf("%d \n",temp->data);  
printf("\n");  
return start;  
}
```

OUTPUT:



```
C:\>  
F:\SYIT\DATA STRUCTURE\c practical program>2c  
Enter the data or enter -1 to exit:37  
enter the data or enter -1 to exit:34  
enter the data or enter -1 to exit:35  
enter the data or enter -1 to exit:23  
enter the data or enter -1 to exit:-1  
Linked list successfully created.  
  
The Linked list is:      37          34          35      23  
  
sort    The Linked List is:23   34      35      37  
  
F:\SYIT\DATA STRUCTURE\c practical program>
```

Implement the following for Stack:

PRACTICAL 3A

Write a program to implement the concept of Stack with Push, Pop, Display and Exit operations.

INPUT:

```
#include<stdio.h>
#include<conio.h>
#define MAX 30
int stack[MAX];
int top = -1; //Stack is empty.
void push();
int pop();
int peek();
void display();
int main()
{
    int choice;
    do
    {
        printf("\n **** Main Menu **** \n");
        printf("1.Push\n");
        printf("2.Pop \n");
        printf("3.Peek \n");
        printf("4.Display \n");
        printf("Enter your choice :");
        scanf("%d",&choice);
        printf("\n");
        switch(choice)
        {
            case 1: push();
            break;
```

```
case 2 : pop();  
break;  
case 3 : peek();  
break;  
case 4 : display();  
break;  
case 5 : break;  
}  
}  
while(choice!=5);  
return 0;  
}  
void push()  
{  
int val;  
if(top == MAX -1)  
{  
printf("Stack is full.");  
} else  
{  
printf("Enter the value to be pushed : ");  
scanf("%d",&val);  
stack[++top]=val;  
printf("Successfully pushed.\n");  
}  
}  
int pop()  
{  
if(top == -1)  
{  
printf("Stack is already empty.");  
}
```



```

else
{
int val = stack[top];
top--;
printf("The value is popped : %d",val);
}
}

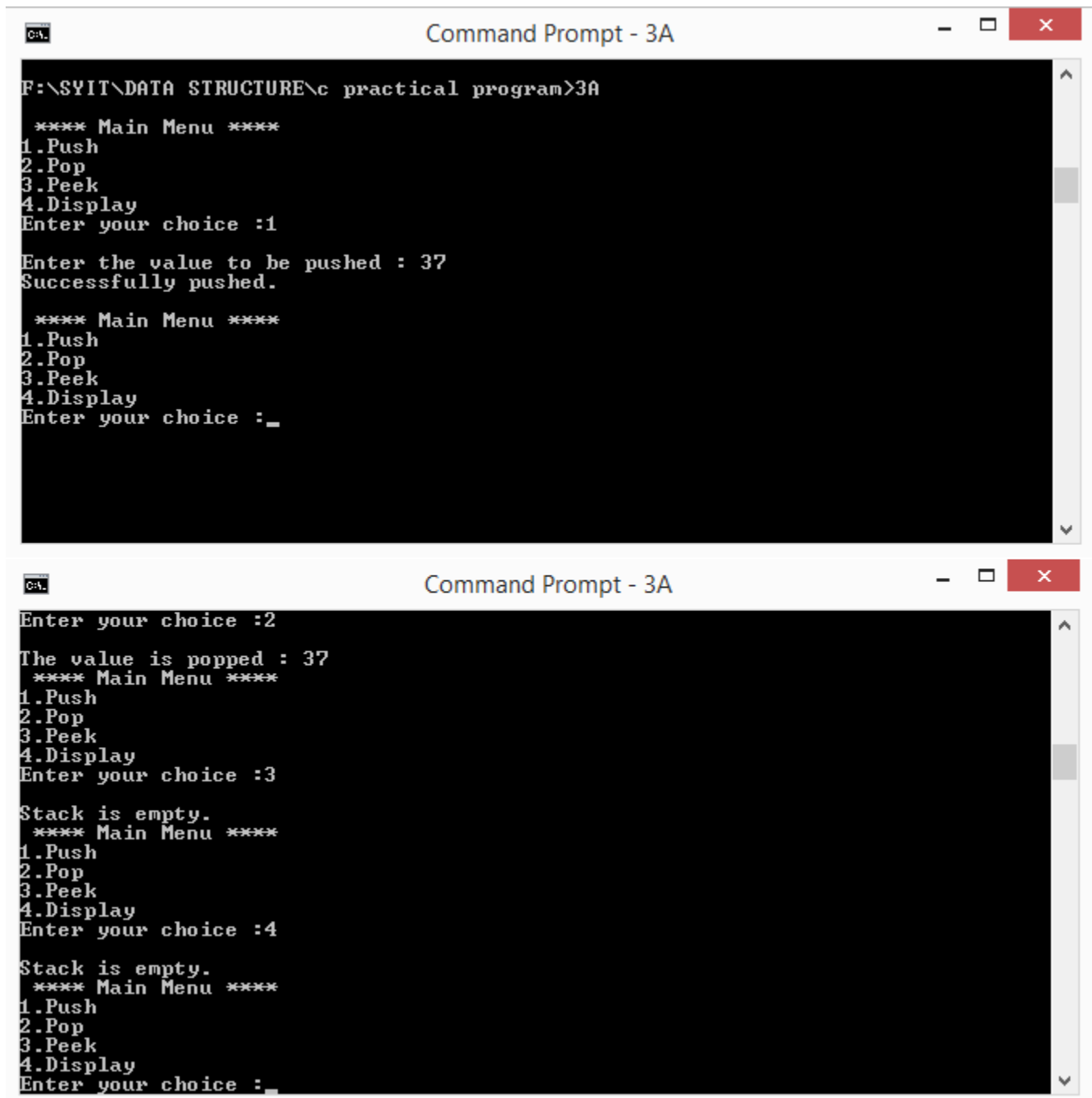
int peek()
{
if(top == -1)
{
printf("Stack is empty.");
}
else
{
int topmost = stack[top];
printf("The topmost element of stack : %d ",topmost);
}
}

void display()
{
if(top == -1)
{
printf("Stack is empty.");
}
else
{
int i;
printf("Stack is : ");
for(i=top;i>=0;i--)
{
printf("\t%d",stack[i]);

```

```
}  
}  
}
```

OUTPUT:



```
Command Prompt - 3A  
F:\SYIT\DATA STRUCTURE\c practical program>3A  
  
**** Main Menu ****  
1.Push  
2.Pop  
3.Peek  
4.Display  
Enter your choice :1  
  
Enter the value to be pushed : 37  
Successfully pushed.  
  
**** Main Menu ****  
1.Push  
2.Pop  
3.Peek  
4.Display  
Enter your choice :_  
  
Command Prompt - 3A  
Enter your choice :2  
The value is popped : 37  
**** Main Menu ****  
1.Push  
2.Pop  
3.Peek  
4.Display  
Enter your choice :3  
Stack is empty.  
**** Main Menu ****  
1.Push  
2.Pop  
3.Peek  
4.Display  
Enter your choice :4  
Stack is empty.  
**** Main Menu ****  
1.Push  
2.Pop  
3.Peek  
4.Display  
Enter your choice :_
```

PRACTICAL 3B

Write a program to convert an infix expression to postfix and prefix conversion.

INPUT:

```
#include<stdio.h>

#include<conio.h>

int move(int n,char source,char temp,char destination);

int main()
{
    int n;

    printf("enter number of Disk");

    scanf("%d",&n);

    move(n,'A','B','C');

    getch();

    return 0;
}

int move(int n,char source,char temp,char destination)
{
    if(n == 1)
    {
        printf("\n Move from %c to %c",source,destination);
    }
    else
    {
        move(n-1,source,destination,temp);
        move(1,source,temp,destination);
        move(n-1,temp,source,destination);
    }
}
```

Output:

```
Command Prompt - 3B
F:\SYIT\DATA STRUCTURE\c practical program>3B
enter number of Disk5

Move from A to C
Move from A to B
Move from C to B
Move from A to C
Move from B to A
Move from B to C
Move from A to C
Move from A to B
Move from C to B
Move from C to A
Move from B to A
Move from C to B
Move from A to C
Move from A to B
Move from C to B
Move from A to C
Move from B to A
Move from B to C
Move from A to C
Move from B to A
Move from C to B
Move from C to A
Move from B to A
Move from B to C
Move from A to C
Move from A to B
Move from C to B
Move from A to C
Move from B to A
Move from B to C
Move from A to C
```

PRACTICAL 3C

Write a program to implement Tower of Hanoi problem.

INPUT:

```
#include<stdio.h>

#include<conio.h>

int move(int n,char source,char temp,char destination);

int main()
{
    int n;

    printf("enter number of Disk");

    scanf("%d",&n);

    move(n,'A','B','C');

    getch();

    return 0;
}

int move(int n,char source,char temp,char destination)
{
    if(n == 1)
    {
        printf("\n Move from %c to %c",source,destination);
    }
    else
    {
        move(n-1,source,destination,temp);
        move(1,source,temp,destination);
        move(n-1,temp,source,destination);
    }
}
```

OUTPUT:

```
Inc.
Tower.c:
Tower.c:25:1: warning: control reaches end of non-void function [-Wreturn-type]
>
1 warning generated.
Turbo Incremental Link 6.75 Copyright (c) 1997-2016 Embarcadero Technologies, Inc.
c.

F:\SYIT\DATA STRUCTURE\c practical program>Tower
enter number of Disk3

Move from A to C
Move from A to B
Move from C to B
Move from A to C
Move from B to A
Move from B to C
Move from A to C
F:\SYIT\DATA STRUCTURE\c practical program>
```

Implement the following for Queue:

PRACTICAL 4A

Write a program to implement the concept of Queue with Insert, Delete, Display and Exit operations.

INPUT:

```
#include<stdio.h>
#include<conio.h>
#define max 30
int rear=-1;
int front=-1;
void insert();
int deleteq();
void display();
int q[max];
void insert()
{
int val;
printf("Enter value to be inserted :");
scanf("%d",&val);
if(rear==max-1)
{
printf("Queue is full.");
}
else if(front==0)
{
front=rear=0;
q[rear]=val;
printf("Value inserted successfully.");
}
else
{
```

```

q[++rear]=val;
printf("Value inserted successfully.");
}
}

int deleteq()
{
if(front== -1)
{
printf("Queue is already empty.");
return -1;
}
else if(front==rear) //Only one item is present.
{
    int val ;
val=q[front];
front=rear=-1;
printf("Value to be deleted : %d",val);
return val;
}
else
{
    int val ;
val=q[front];
front++;
printf("Value to be deleted : %d",val);
return val;
}
}

void display()
{
if(front== -1)
{
printf("Queue is empty.");

```



```

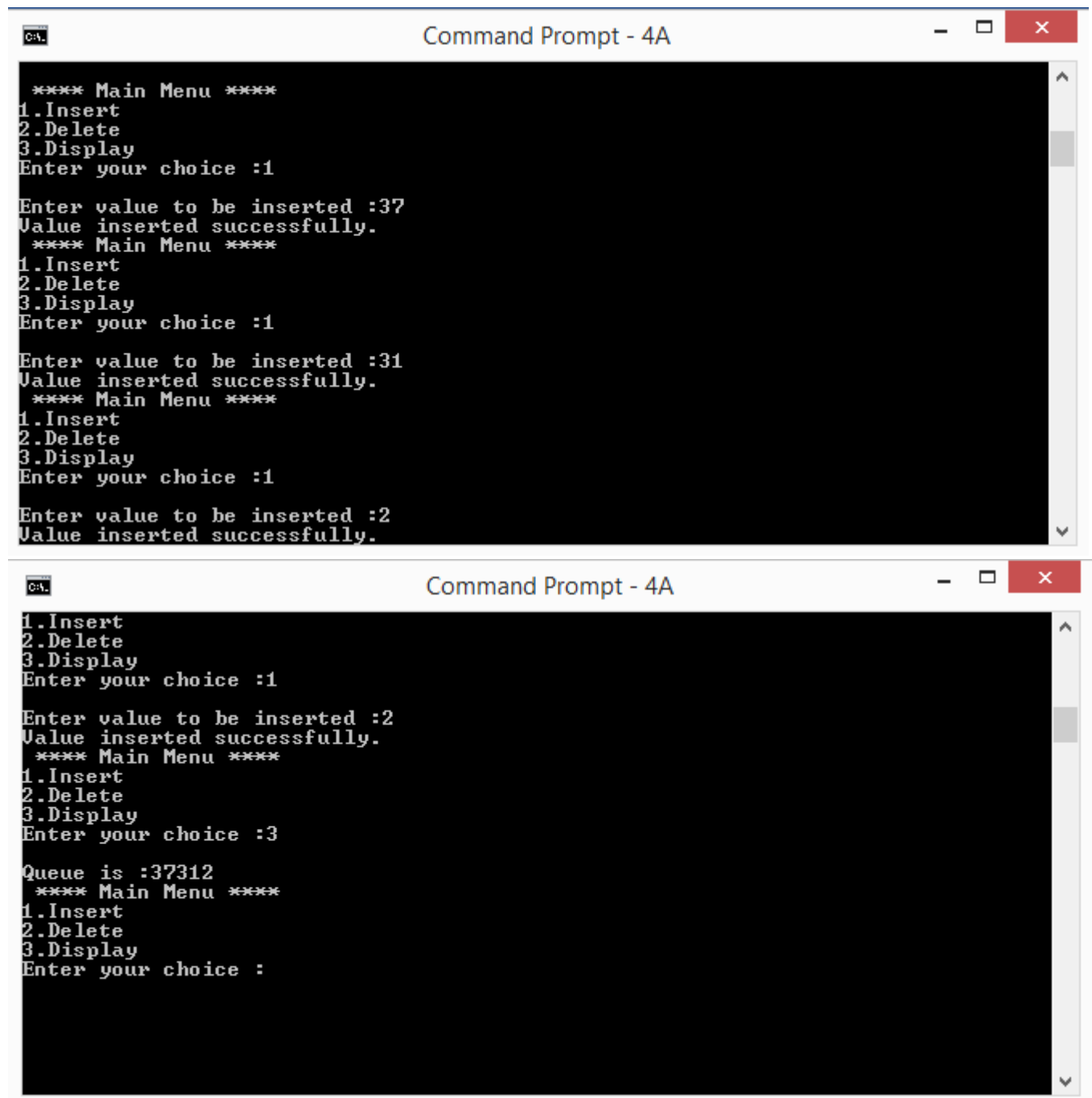
}
else
{
int i;
printf("Queue is :");
for(i=front;i<=rear;i++)
{
printf("%d",q[i]);
}
}
}

int main()
{
int choice;
do
{
printf("\n ***** Main Menu ***** \n");
printf("1.Insert\n");
printf("2.Delete\n");
printf("3.Display \n");
printf("Enter your choice :");
scanf("%d",&choice);
printf("\n");
switch(choice)
{
case 1: insert();
break;
case 2 : deleteq();
break;
case 3 : display();
break;
case 4 : break;

```

```
}  
  
}  
  
while(choice!=4);  
  
return 0;  
  
}
```

OUTPUT:



```
Command Prompt - 4A  
**** Main Menu ****  
1.Insert  
2.Delete  
3.Display  
Enter your choice :1  
Enter value to be inserted :37  
Value inserted successfully.  
**** Main Menu ****  
1.Insert  
2.Delete  
3.Display  
Enter your choice :1  
Enter value to be inserted :31  
Value inserted successfully.  
**** Main Menu ****  
1.Insert  
2.Delete  
3.Display  
Enter your choice :1  
Enter value to be inserted :2  
Value inserted successfully.  
  
Command Prompt - 4A  
1.Insert  
2.Delete  
3.Display  
Enter your choice :1  
Enter value to be inserted :2  
Value inserted successfully.  
**** Main Menu ****  
1.Insert  
2.Delete  
3.Display  
Enter your choice :3  
Queue is :37312  
**** Main Menu ****  
1.Insert  
2.Delete  
3.Display  
Enter your choice :
```

PRACTICAL 4B

Write a program to implement the concept of Circular Queue.

INPUT:

```
/*
 * C++ Program to Implement Circular Queue
 */
#include <iostream>
#define MAX 5
using namespace std;
/*
 * Class Circular Queue
 */
class Circular_Queue
{
private:
    int *cqueue_arr;
    int front, rear;
public:
    Circular_Queue()
    {
        cqueue_arr = new int [MAX];
        rear = front = -1;
    }
    /*
     * Insert into Circular Queue
     */
    void insert(int item)
    {
        if ((front == 0 && rear == MAX-1) || (front == rear+1))
        {
            cout<<"Queue Overflow \n";
```

```

        return;
    }
    if (front == -1)
    {
        front = 0;
        rear = 0;
    }
    else
    {
        if (rear == MAX - 1)
            rear = 0;
        else
            rear = rear + 1;
    }
    cqueue_arr[rear] = item ;
}

/*
 * Delete from Circular Queue
 */
void del()
{
    if (front == -1)
    {
        cout<<"Queue Underflow\n";
        return ;
    }
    cout<<"Element deleted from queue is : "<<cqueue_arr[front]<<endl;
    if (front == rear)
    {
        front = -1;
        rear = -1;
    }
}

```

```

else
{
    if (front == MAX - 1)
        front = 0;
    else
        front = front + 1;
}
}
/*
 * Display Circular Queue
 */
void display()
{
    int front_pos = front, rear_pos = rear;
    if (front == -1)
    {
        cout<<"Queue is empty\n";
        return;
    }
    cout<<"Queue elements :\n";
    if (front_pos <= rear_pos)
    {
        while (front_pos <= rear_pos)
        {
            cout<<cqueue_arr[front_pos]<<" ";
            front_pos++;
        }
    }
    else
    {
        while (front_pos <= MAX - 1)
        {

```

```

        cout<<cqueue_arr[front_pos]<<" ";
        front_pos++;
    }
    front_pos = 0;
    while (front_pos <= rear_pos)
    {
        cout<<cqueue_arr[front_pos]<<" ";
        front_pos++;
    }
}
cout<<endl;
}

};

/*
 * Main
 */
int main()
{
    int choice, item;
    Circular_Queue cq;
    do
    {
        cout<<"1.Insert\n";
        cout<<"2.Delete\n";
        cout<<"3.Display\n";
        cout<<"4.Quit\n";
        cout<<"Enter your choice : ";
        cin>>choice;
        switch(choice)
        {
            case 1:
                cout<<"Input the element for insertion in queue : ";

```

```
    cin>>item;
    cq.insert(item);
        break;
    case 2:
    cq.del();
        break;
    case 3:
    cq.display();
        break;
    case 4:
        break;
    default:
        cout<<"Wrong choice\n";
    }/*End of switch*/
}
while(choice != 4);
return 0;
}
```

OUTPUT:

CA. Command Prompt - 4B

```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\GG>F:

F:\>cd F:\SYIT\DATA STRUCTURE\c practical program

F:\SYIT\DATA STRUCTURE\c practical program>4B
1.Insert
2.Delete
3.Display
4.Quit
Enter your choice : 1
Input the element for insertion in queue : 37
1.Insert
2.Delete
3.Display
4.Quit
Enter your choice : 1
Input the element for insertion in queue : 31
1.Insert
2.Delete
3.Display
4.Quit
Enter your choice :
```

CA. Command Prompt

```
2.Delete
3.Display
4.Quit
Enter your choice : 1
Input the element for insertion in queue : 31
1.Insert
2.Delete
3.Display
4.Quit
Enter your choice : 1
Input the element for insertion in queue : 2
1.Insert
2.Delete
3.Display
4.Quit
Enter your choice : 3
Queue elements :
37 31 2
1.Insert
2.Delete
3.Display
4.Quit
Enter your choice : 4

F:\SYIT\DATA STRUCTURE\c practical program>_
```


PRACTICAL 4C

Write a program to implement the concept of Deque.

INPUT:

```
#include<stdio.h>

#include<conio.h>

#define max 5

int front = -1;

int rear = -1;

int insert_rear();

int insert_front();

void display();

int deleteq_rear();

int deleteq_front();

int q[max];

int main()

{
    int choice;

    clrscr();

    do
    {

        printf("\n **** Main Menu **** \n");

        printf("1.Insert From Rear\n");

        printf("2.Insert From Front\n");

        printf("3.Delete From Front \n");

        printf("4.Delete From Rear \n");

        printf("5.Display\n");

        printf("Enter your choice :");

        scanf("%d",&choice);

        printf("\n");

        switch(choice)

        {

            case 1: insert_rear();
```

```

        break;

        case 2 : insert_front();

        break;

        case 3 : deleteq_front();

        break;

        case 4 : deleteq_rear();

        break;

        case 5 : display();

        break;

        case 6 : break;

    }

}

while(choice!=6);

return 0;

}

int insert_rear()
{
    int val;

    printf("Enter value :");

    scanf("%d",&val);

    if((rear+1)%max==front)
    {

        printf("Queue is full.");

        return 0;

    }

    else if(rear==max-1)
    {

        rear=front=0;

        q[rear]=val;

        printf("Inserted successfully.");

        return val;

    }

    else

```

```

    {
        rear=(rear + 1)%max;
        q[rear]=val;
        printf("Inserted successfully.");
        return val;
    }
}

int insert_front()
{
    int val;
    printf("Enter value :");
    scanf("%d",&val);
    if((rear+1)%max==front)
    {
        printf("Queue is full.");
        return 0;
    }
    else if(front== -1)
    {
        rear=front=0;
        q[front]=val;
        printf("Inserted successfully.");
        return val;
    }
    else
    {
        front=(front-1+max)%max;
        q[front]=val;
        printf("Inserted successfully.");
        return val;
    }
}

int deleteq_front()

```

```

{
    int val;
    if(front== -1)
    {
        printf("Queue is empty.");
        // return -1;
    }
    else if(front == rear)
    {
        int val=q[front];
        front=rear= -1;
        printf("Deleted value : %d",val);
        return val;
    }
    else
    {
        val=q[front];
        front=(front+1)%max;
        printf("Deleted value : %d",val);
        return val;
    }
}

int deleteq_rear()
{
    int val;
    if(rear== -1)
    {
        printf("Queue is empty.");
        return -1;
    }
    else if(front == rear)
    {
        int val=q[rear];
        front=rear= -1;
    }
}

```

```

        printf("Deleted value : %d",val);
        return val;
    }
    else
    {
        val=q[rear];
        rear=(rear-1+max)%max;
        printf("Deleted value : %d",val);
        return val;
    }
}

void display()
{
    int i;
    if(front ==-1)
    {
        printf("Queue is empty.");
    }
    else
    {
        printf("Queue is :"); for(i=front;i!
        =rear;i=(i+1)%max)
        {
            printf(" %d ",q[i]);
        }
        printf(" %d ",q[i]);
    }
}
}

```

OUTPUT:

Command Prompt - deque

```
**** Main Menu ****
1.Insert From Rear
2.Insert From Front
3.Delete From Front
4.Delete From Rear
5.Display
Enter your choice :1

Enter value :37
Inserted successfully.
**** Main Menu ****
1.Insert From Rear
2.Insert From Front
3.Delete From Front
4.Delete From Rear
5.Display
Enter your choice :1

Enter value :31
Inserted successfully.
**** Main Menu ****
1.Insert From Rear
2.Insert From Front
3.Delete From Front
```

Command Prompt - deque

```
Enter value :31
Inserted successfully.
**** Main Menu ****
1.Insert From Rear
2.Insert From Front
3.Delete From Front
4.Delete From Rear
5.Display
Enter your choice :1

Enter value :2
Inserted successfully.
**** Main Menu ****
1.Insert From Rear
2.Insert From Front
3.Delete From Front
4.Delete From Rear
5.Display
Enter your choice :
```

```
CA. Command Prompt - deque
1.Insert From Rear
2.Insert From Front
3.Delete From Front
4.Delete From Rear
5.Display
Enter your choice :1

Enter value :2
Inserted successfully.
**** Main Menu ****
1.Insert From Rear
2.Insert From Front
3.Delete From Front
4.Delete From Rear
5.Display
Enter your choice :5

Queue is : 37 31 2
**** Main Menu ****
1.Insert From Rear
2.Insert From Front
3.Delete From Front
4.Delete From Rear
5.Display
Enter your choice :_
```

Implement the following sorting techniques:

PRACTICAL 5A

Write a program to implement bubble sort.

INPUT:

```
#include<stdio.h>
#include<conio.h>
int size,val;
void disp(int size);
int sort(int size);
int arr[20];
int main()
{
    int i,ch;
    printf("Enter the size of array : ");
    scanf("%d",&size);
    for(i=0;i<size;i++)
    {
        scanf("%d",&arr[i]);
    }
    do
    {
        printf("\n*****Main Menu*****\n");
        printf("1.Display\n");
        printf("2.Sorting\n");
        printf("Enter your Choice : ");
        scanf("%d",&ch);
        switch(ch)
        {
            case 1:disp(size);
                break;
            case 2:sort(size);
```



```

        break;

    }

}

while(ch!=2);

getch ();

return 0;

}

void disp(int size)
{
    int i;

    printf("Given Array :\n");

    for(i=0;i<size;i++)
    {
        printf("%d\n",arr[i]);
    }
}

int sort(size)
{
    int i,j;

    for(i=0;i<size;i++)
    {
        for(j=0;j<size-i-1;j++)
        {
            if(arr[j]>arr[j+1])
            {
                int temp;

                temp=arr[j];

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

                arr[j+1]=temp;
            }
        }
    }
}

```

```

printf("Sorted Array : \n");
for(i=0;i<size;i++)
{
    printf("%d \n",arr[i]);
}
}

```

OUTPUT:

```

Command Prompt - bubblesort

F:\>cd F:\SYIT\DATA STRUCTURE\c practical program
F:\SYIT\DATA STRUCTURE\c practical program>bubblesort
Enter the size of array : 4
37
31
2
6

****Main Menu****
1.Display
2.Sorting
Enter your Choice : 1
Given Array :
37
31
2
6

****Main Menu****
1.Display
2.Sorting
Enter your Choice : 2
Sorted Array :
2
6
31
37

```

PRACTICAL 5B

Write a program to implement selection sort.

INPUT:

```
#include<stdio.h>

#include<conio.h>

#include<malloc.h>

int selection_sort(int n);

int A[20];

int selection_sort(int n)
{
    int imin,i,j,temp;
    for(i=0;i<n;i++)
    {
        imin=i;
        for(j=i+1;j<n;j++)
        {
            if(A[imin]>A[j])
            {
                imin=j;
            }
        }
        temp = A[i];
        A[i] = A[imin];
        A[imin] = temp;
    }

    printf("Successfully sorted using Selection sort :");
}

int main()
{
    int n,i;

    printf("Enter the size :");
```

```

scanf("%d",&n);
printf("Enter the element :\n");
for(i=0;i<n;i++)
{
    scanf("%d",&A[i]);
    printf("\n");
}

selection_sort(n);
    for(i=0;i<n;i++)
    {
        printf("\n %d\n",A[i]);
    }

return 0;
}

```

OUTPUT:

```

C:\>
F:\SYIT\DATA STRUCTURE\c practical program>selectionsort
Enter the size :5
Enter the element :
37
31
2
43
75
Successfully sorted using Selection sort :
2
31
37
43
75
F:\SYIT\DATA STRUCTURE\c practical program>

```

PRACTICAL 5C

Write a program to implement insertion sort.

INPUT:

```
#include<stdio.h>

#include<conio.h>

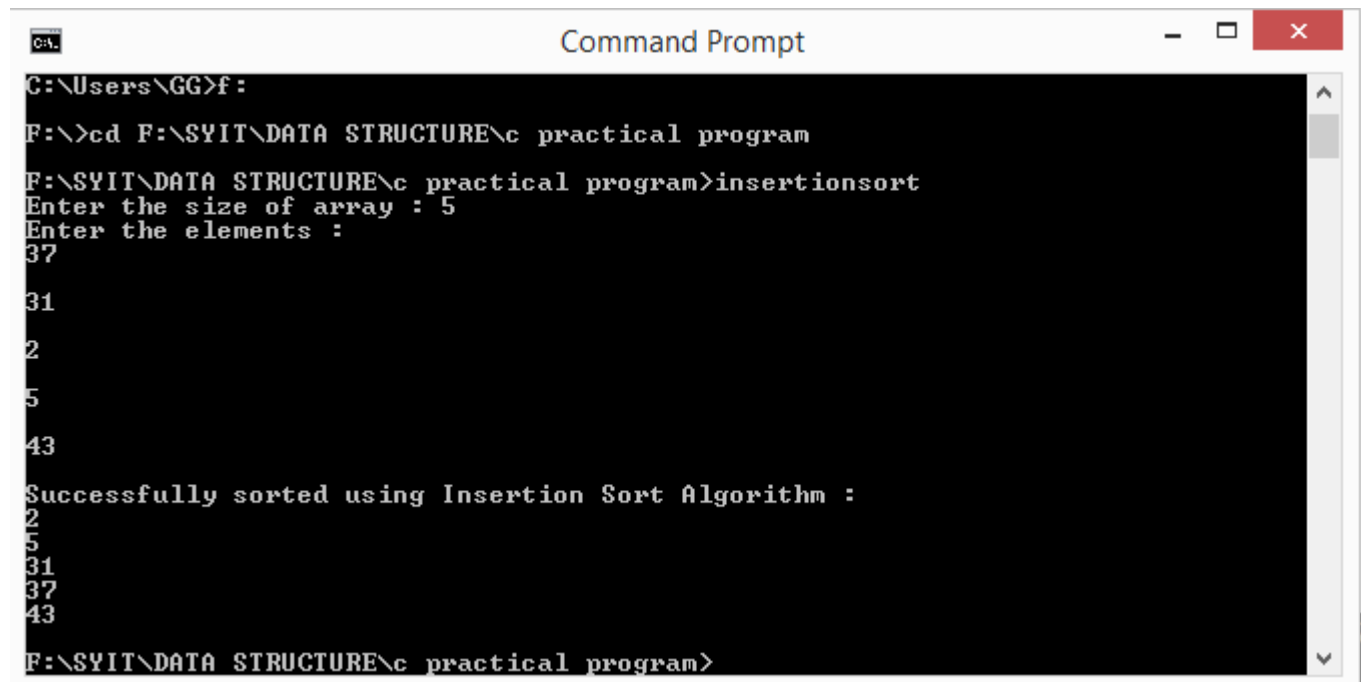
int A[10];

void insertion_sort(int n)
{
    int val,vacant,i;
    for(i=1;i<n;i++)
    {
        val=A[i];
        vacant=i;
        while(A[vacant-1]>val && vacant!=0)
        {
            A[vacant]=A[vacant-1];
            vacant=vacant - 1;
        }
        A[vacant]=val;
    }
    printf("Successfully sorted using Insertion Sort Algorithm : \n");
}

int main()
{
    int n,i;
    printf("Enter the size of array : ");
    scanf("%d",&n);
    printf("Enter the elements :\n");
    for(i=0;i<n;i++)
    {
        scanf("%d",&A[i]);
    }
}
```

```
        printf("\n");  
    }  
    insertion_sort(n);  
    for(i=0;i<n;i++)  
    {  
        printf("%d \n",A[i]);  
    }  
}
```

OUTPUT:



```
C:\Users\GG>f:  
F:\>cd F:\SYIT\DATA STRUCTURE\c practical program  
F:\SYIT\DATA STRUCTURE\c practical program>insertionsort  
Enter the size of array : 5  
Enter the elements :  
37  
31  
2  
5  
43  
Successfully sorted using Insertion Sort Algorithm :  
2  
5  
31  
37  
43  
F:\SYIT\DATA STRUCTURE\c practical program>
```

Implement the following data structure techniques:

PRACTICAL 6A

Write a program to implement merge sort.

INPUT:

```
#include<stdio.h>

#include<conio.h>

#define max 10

int a[11] = { 10, 14, 19, 26, 27, 31, 33, 35, 42, 44, 0 };

int b[10];

void merging(int low, int mid, int high) {

    int l1, l2, i;

    for(l1 = low, l2 = mid + 1, i = low; l1 <= mid && l2 <= high; i++) {

        if(a[l1] <= a[l2])

            b[i] = a[l1++];

        else

            b[i] = a[l2++];

    }

    while(l1 <= mid)

        b[i++] = a[l1++];

    while(l2 <= high)

        b[i++] = a[l2++];

    for(i = low; i <= high; i++)

        a[i] = b[i];

}

void sort(int low, int high) {

    int mid;

    if(low < high) {
```

```
    mid = (low + high) / 2;
    sort(low, mid);
    sort(mid+1, high);
    merging(low, mid, high);
} else {
    return;
}
}
```

```
int main() {
    int i;
    clrscr();
    printf("List before sorting\n");

    for(i = 0; i <= max; i++)
        printf("%d ", a[i]);
    sort(0, max);
    printf("\nList after sorting\n");
    for(i = 0; i <= max; i++)
        printf("%d ", a[i]);
    getch();
}
```

OUTPUT:

Command Prompt - mergesort

```
List before sorting  
10 14 19 26 27 31 33 35 42 44 0  
List after sorting  
0 10 14 19 26 27 31 33 35 42 44
```

PRACTICAL 6B

Write a program to search the element using sequential search.

INPUT:

```
#include<stdio.h>

#include<conio.h>

int size,val;

void disp(int size);

void search(int val,int size);

int arr[20];

int main()
{
    int i,ch;
    printf("Enter the size of array : ");
    scanf("%d",&size);
    for(i=0;i<size;i++)
    {
        scanf("%d",&arr[i]);
    }
    do
    {
        printf("\n****Main Menu****\n");
        printf("1.Display\n");
        printf("2.Search\n");
        printf("Enter your Choice : ");
        scanf("%d",&ch);
        switch(ch)
        {
            case 1:disp(size);
                break;

            case 2:printf("Enter value to be search : ");
                scanf("%d",&val);
```

```

        search(val,size);

        break;
    }
}

while(ch!=2);

getch ();

return 0;

}

void search(int val,int size)
{
    int i;
    for(i=0;i<size;i++)
    {
        if(arr[i]==val)
        {
            printf("Value is found at %d position.",i);

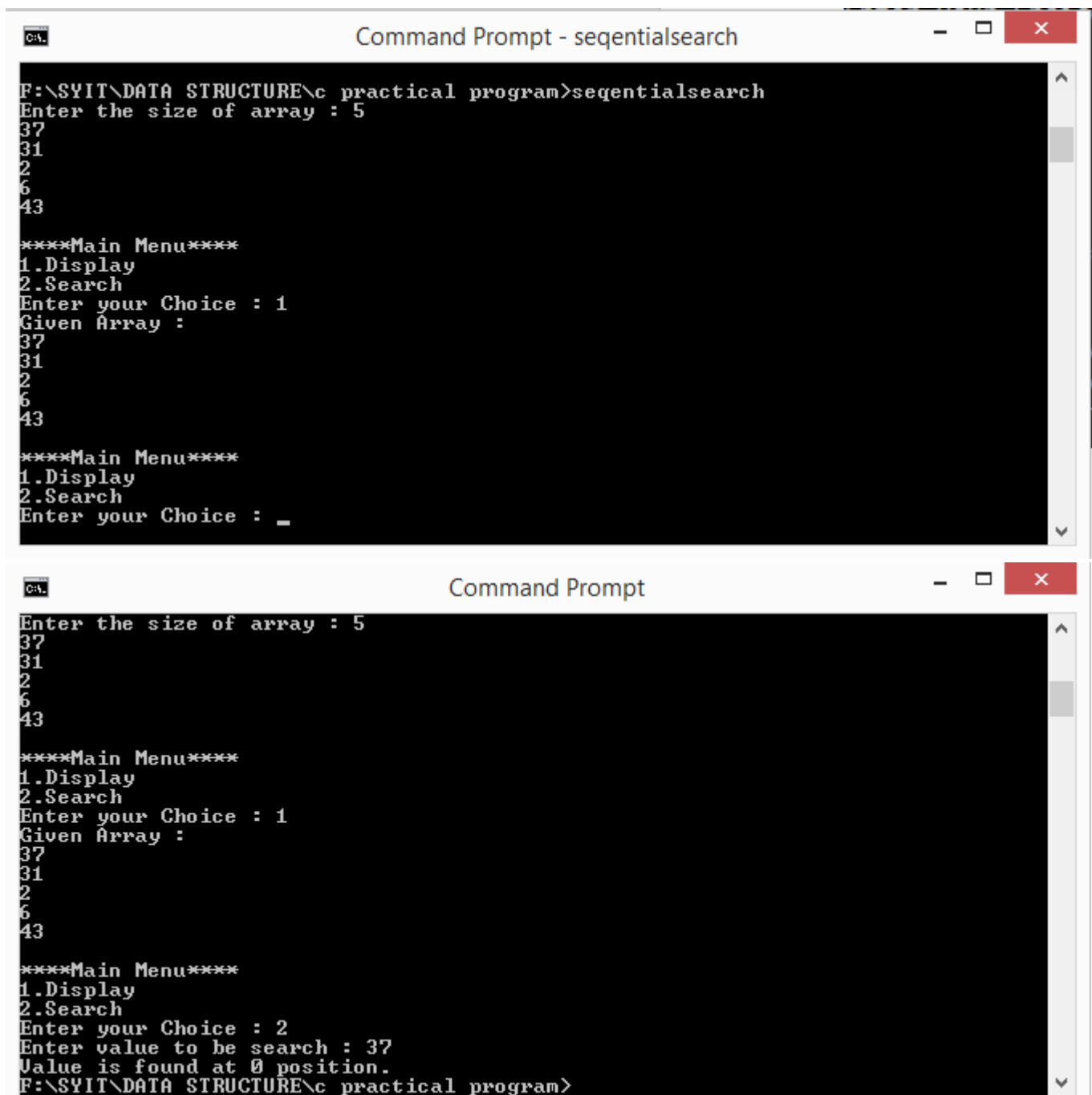
            break;
        }
    }
    if(i==size)
    {
        printf("Value is not found.");
    }
}

void disp(int size)
{
    int i;
    printf("Given Array :\n");
    for(i=0;i<size;i++)
    {
        printf("%d\n",arr[i]);
    }
}

```

```
}
```

OUTPUT:



```
Command Prompt - sequentialsearch

F:\SYIT\DATA STRUCTURE\c practical program>sequentialsearch
Enter the size of array : 5
37
31
2
6
43

****Main Menu****
1.Display
2.Search
Enter your Choice : 1
Given Array :
37
31
2
6
43

****Main Menu****
1.Display
2.Search
Enter your Choice : _

Command Prompt

Enter the size of array : 5
37
31
2
6
43

****Main Menu****
1.Display
2.Search
Enter your Choice : 1
Given Array :
37
31
2
6
43

****Main Menu****
1.Display
2.Search
Enter your Choice : 2
Enter value to be search : 37
Value is found at 0 position.
F:\SYIT\DATA STRUCTURE\c practical program>
```

PRACTICAL 6C

Write a program to search the element using binary search.

INPUT:

```
#include <stdio.h>

int main()
{
    int c, first, last, middle, n, search, array[100];

    printf("Enter number of elements\n");
    scanf("%d",&n);

    printf("Enter %d integers\n", n);

    for (c = 0; c < n; c++)
        scanf("%d",&array[c]);

    printf("Enter value to find\n");
    scanf("%d", &search);

    first = 0;
    last = n - 1;
    middle = (first+last)/2;

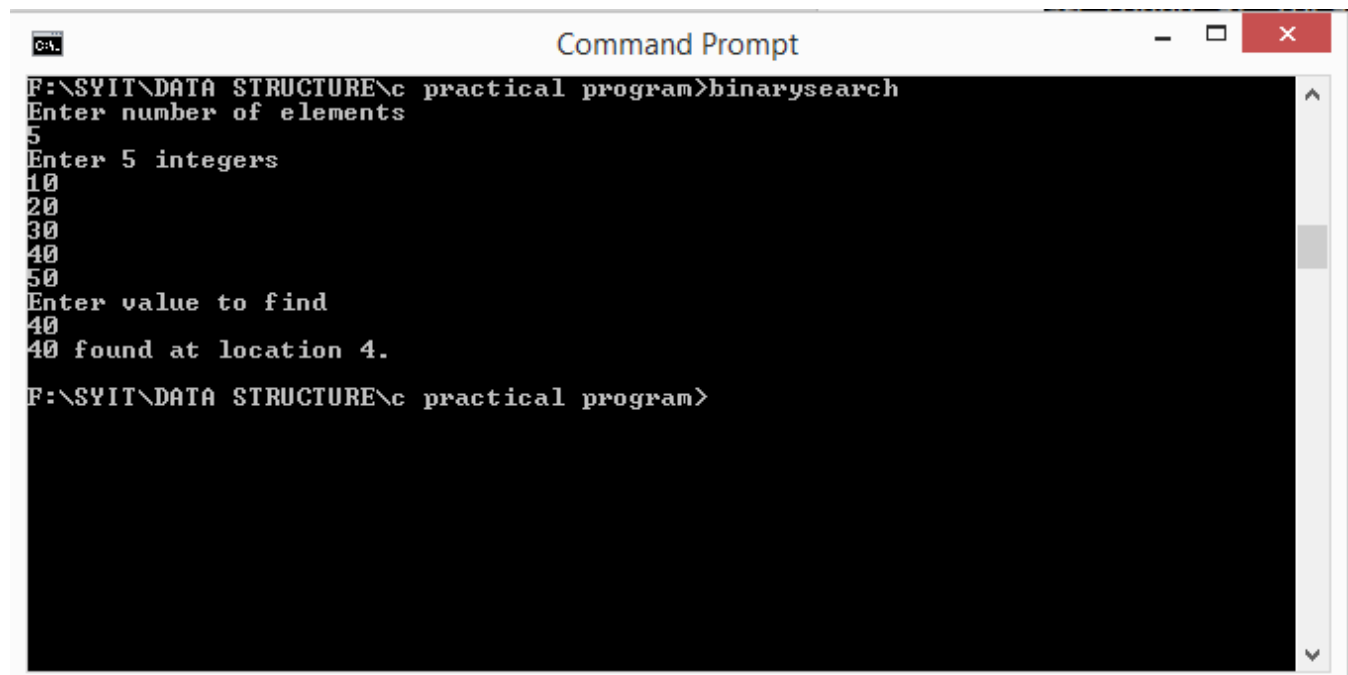
    while (first <= last)
    {
        if (array[middle] < search)
            first = middle + 1;
        else if (array[middle] == search)
        {
            printf("%d found at location %d.\n", search, middle+1);
```

```
        break;
    }
    else if(array[middle]>search)
        last = middle - 1;

    middle = (first + last)/2;
}
if (first > last)
    printf("Not found! %d is not present in the list.\n", search);

return 0;
}
```

OUTPUT:



The screenshot shows a Windows Command Prompt window titled "Command Prompt". The prompt is at the directory `F:\SYIT\DATA STRUCTURE\c practical program`. The user has entered the command `binarysearch`. The program then prompts for the number of elements, which is entered as `5`. It then prompts for 5 integers, which are entered as `10`, `20`, `30`, `40`, and `50`. Next, it prompts for the value to find, which is entered as `40`. The program outputs `40 found at location 4.` and returns to the command prompt.

```
Command Prompt
F:\SYIT\DATA STRUCTURE\c practical program>binarysearch
Enter number of elements
5
Enter 5 integers
10
20
30
40
50
Enter value to find
40
40 found at location 4.
F:\SYIT\DATA STRUCTURE\c practical program>
```

Implement the following data structure techniques:

PRACTICAL 7

A .Write a program to create the tree and display the elements.

B . Write a program to construct the binary tree.

C . Write a program for inorder, postorder and preorder traversal of tree

INPUT:

```
#include<stdio.h>
#include<malloc.h>
struct node
{
int data;
struct node *left;
struct node *right;
};
struct node *root=NULL;
struct node *create(struct node*);
struct node *display(struct node*);
void preorder(struct node *temp);
void postorder(struct node *temp);
void inorder(struct node *temp);
int main()
{
int choice, val, count, min, max;
do
{
printf("***** Main Menu ***\n");
printf("1. create a binary search\n");
printf("2. Display the tree \n");
printf("3. EXIT \n");
printf("Enter your choice:");
scanf("%d",&choice);
```

```

printf("\n\n");
switch(choice)
{
case 1:root=create(root);
break;
case 2:root=display(root);
break;
case 3:break;
}
}while(choice!=3);
return 0;
}

struct node *create(struct node *root)
{
struct node *newnode=NULL,*temp=NULL,*parent=NULL;
int val;
printf("Enter the data or enter -1 to exit:");
scanf("%d",&val);
while(val!=-1)
{
newnode=(struct node*)malloc(sizeof(struct node));
newnode->data=val;
if(root==NULL)
{
root=newnode;
newnode->left=NULL;
newnode->right=NULL;
}
else
{
temp=root;
while(temp!=NULL)

```



```

{
parent=temp;
if(val<temp->data)
{
temp=temp->left;
}
else
{
temp=temp->right;
}
}
if(val<parent->data)
{
parent->left=newnode;
newnode->left=NULL;
newnode->right=NULL;
}
else
{
parent->right=newnode;
newnode->left=NULL;
newnode->right=NULL;
}
}
printf("Enter the data or enter -1 to exit:");
scanf("%d",&val);
}
printf("Succesfully created \n");
return root;
}
struct node *display(struct node *root)
{

```

```

int choice1;

printf("*** Display Menu***\n");
printf("1.pre-order\n");
printf("2.In-order\n");
printf("3.post-order\n");
printf("4. EXIT\n");
printf("Enter your choice :");
scanf("%d",&choice1);
switch(choice1)
{
case 1:printf("\tThe Pre-order Traversal is:");
preorder(root);
break;
case 2:printf("\tThe in order traversal is:");
inorder(root);
break;
case 3:printf("\tThe post-order traversal is:");
postorder(root);
break;
case 4:break;
}
printf("\n");
return root;
}

void preorder(struct node *temp)
{
if(temp!=NULL)
{
printf("%d",temp->data);
preorder(temp->left);
preorder(temp->right);
}
}

```

```
}  
  
void postorder(struct node *temp)  
{  
    if(temp!=NULL)  
    {  
        postorder(temp->left);  
        postorder(temp->right);  
        printf("%d",temp->data);  
  
    }  
}  
  
void inorder(struct node *temp)  
{  
    if(temp!=NULL)  
    {  
        inorder(temp->left);  
        printf("%d",temp->data);  
        inorder(temp->right);  
    }  
}
```

OUTPUT:

```
Command Prompt - Q7
F:\>cd F:\SYIT\DATA STRUCTURE\c practical program
F:\SYIT\DATA STRUCTURE\c practical program>Q7
**** Main Menu ****
1. create a binary search
2. Display the tree
3. EXIT
Enter your choice:1

Enter the data or enter -1 to exit:1
Enter the data or enter -1 to exit:2
Enter the data or enter -1 to exit:3
Enter the data or enter -1 to exit:4
Enter the data or enter -1 to exit:5
Enter the data or enter -1 to exit:-1
Sucesfully created
**** Main Menu ****
1. create a binary search
2. Display the tree
3. EXIT
Enter your choice:
```

```
Command Prompt - Q7
Enter the data or enter -1 to exit:2
Enter the data or enter -1 to exit:3
Enter the data or enter -1 to exit:4
Enter the data or enter -1 to exit:5
Enter the data or enter -1 to exit:-1
Sucesfully created
**** Main Menu ****
1. create a binary search
2. Display the tree
3. EXIT
Enter your choice:2

*** Display Menu***
1.pre-order
2.In-order
3.post-order
4. EXIT
Enter your choice :1
The Pre-order Traversal is:12345
**** Main Menu ****
1. create a binary search
2. Display the tree
3. EXIT
Enter your choice:_
```

```
Command Prompt - Q7

4. EXIT
Enter your choice :1
    The Pre-order Traversal is:12345
**** Main Menu ****
1. create a binary search
2. Display the tree
3. EXIT
Enter your choice:2

**** Display Menu****
1.pre-order
2.In-order
3.post-order
4. EXIT
Enter your choice :3
    The post-order traversal is:54321
**** Main Menu ****
1. create a binary search
2. Display the tree
3. EXIT
Enter your choice:
```

```
Command Prompt

Enter your choice :1
    The Pre-order Traversal is:12345
**** Main Menu ****
1. create a binary search
2. Display the tree
3. EXIT
Enter your choice:2

**** Display Menu****
1.pre-order
2.In-order
3.post-order
4. EXIT
Enter your choice :3
    The post-order traversal is:54321
**** Main Menu ****
1. create a binary search
2. Display the tree
3. EXIT
Enter your choice:3

F:\SYIT\DATA STRUCTURE\c practical program>
```

Implement the following data structure techniques:

PRACTICAL 8

A . Write a program to insert the element into maximum heap.

INPUT:

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#define SIZE 30
int a[SIZE],n;
void maxHeapify(int a[],int i,int n1);
void buildHeap(int a[],int n1);
void heap_sort(int a[]);
void swap(int i,int j);
int length(int a[]);
void main()
{
    int i,j;
    clrscr();
    printf("Enter the number of element:");
    scanf("%d",&n);
    for(i=0;i<=n;i++)
    {
        printf("Enter a Value:");
        scanf("%d",&a[i]);
        buildHeap(a,i);
    }
    for(j=0;j<n;j++)
    {
        printf("%d ",a[j]);
    }
    printf("\n");
    getch();
}

void buildHeap(int a[],int n1)
{
    int i,j;
    for(i=(n1/2)-1;i>=0;i--)
    {
        maxHeapify(a,i,n1);
    }
    for(j=0;j<n1;j++)
    {
        printf("%d ",a[j]);
    }

    printf("\n");
}

void maxHeapify(int a[],int i,int n1)
{

```

```

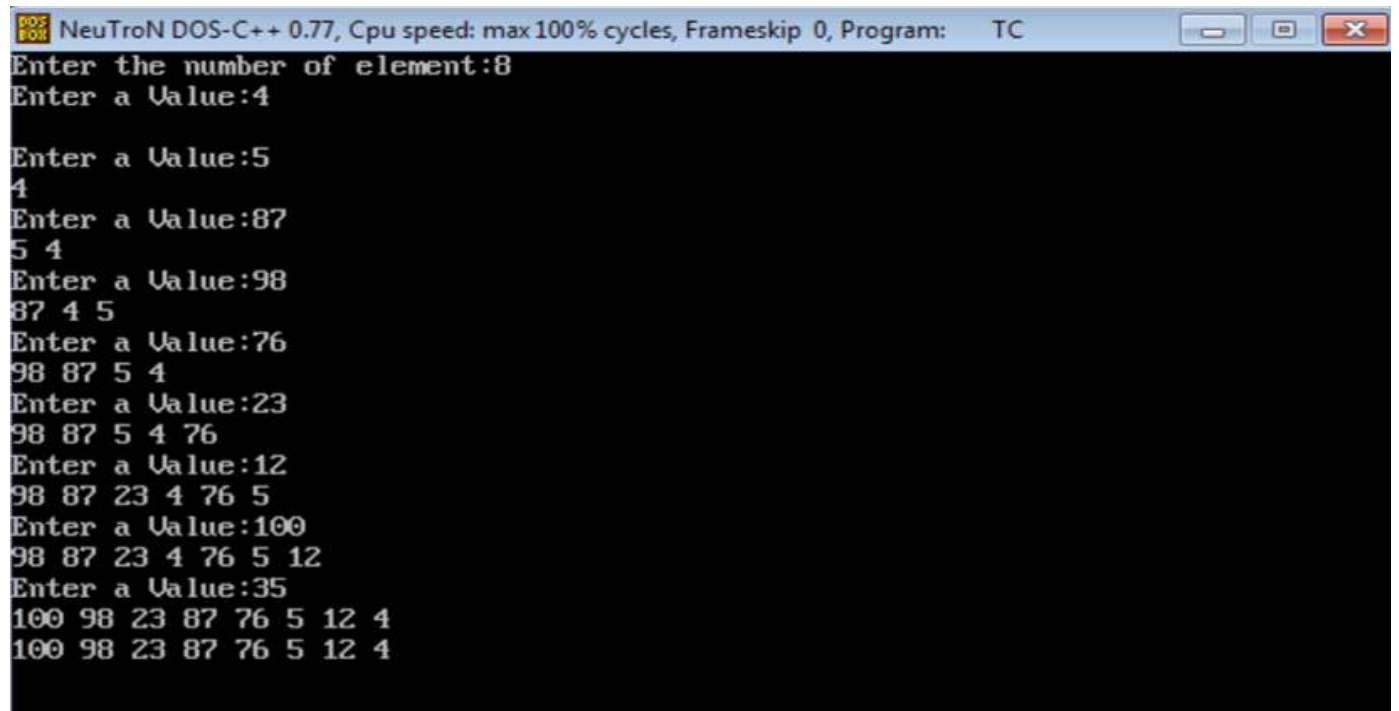
int max,l,r;
max=i;
l=2*i+1;
r=2*i+2;
if(l<n1 && r<n1)
{
if(a[l]>a[max])
{
max=l;
}
if(a[r]>a[max])
{
max=r;
}
}
else if(l<n1 && r>=n1)
{
if(a[l]>a[max])
{
max=l;
}
}
else if(l>=n1 && r<n1)
{
if(a[r]>a[max])
{
max=r;
}
}
if(i!=max)
{
swap(i,max);
maxHeapify(a,max,n1);
}
}
void swap(int i,int j)
{
int temp=a[i];
a[i]=a[j];
a[j]=temp;
}

int length(int a[])
{
int i=0;

while(a[i]!='\0')
{
i++;
}
return i;
}

```

OUTPUT:



```
NeuTroN DOS-C++ 0.77, Cpu speed: max 100% cycles, Frameskip 0, Program: TC
Enter the number of element:8
Enter a Value:4
Enter a Value:5
4
Enter a Value:87
5 4
Enter a Value:98
87 4 5
Enter a Value:76
98 87 5 4
Enter a Value:23
98 87 5 4 76
Enter a Value:12
98 87 23 4 76 5
Enter a Value:100
98 87 23 4 76 5 12
Enter a Value:35
100 98 23 87 76 5 12 4
100 98 23 87 76 5 12 4
```


B. Write a program to insert the element into minimum heap.

INPUT:

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#define SIZE 30
int a[SIZE],n;
void minHeapify(int a[],int i,int n1);
void buildHeap(int a[],int n1);
void heap_sort(int a[]);
void swap(int i,int j);
int length(int a[]);
void main()
{
    int i,j;
    clrscr();
    printf("Enter the number of element:");
    scanf("%d",&n);
    for(i=0;i<=n;i++)
    {
        printf("Enter a Value:");
        scanf("%d",&a[i]);
        buildHeap(a,i);
    }
    for(j=0;j<n;j++)
    {
        printf("%d ",a[j]);
    }
    printf("\n");
    getch();
}

void buildHeap(int a[],int n1)
{
    int i,j;
    for(i=(n1/2)-1;i>=0;i--)
    {
        minHeapify(a,i,n1);
    }
    for(j=0;j<n1;j++)
    {
        printf("%d ",a[j]);
    }
    printf("\n");
}

void minHeapify(int a[],int i,int n1)
{
    int min,l,r;
    min=i;
    l=2*i+1;
```

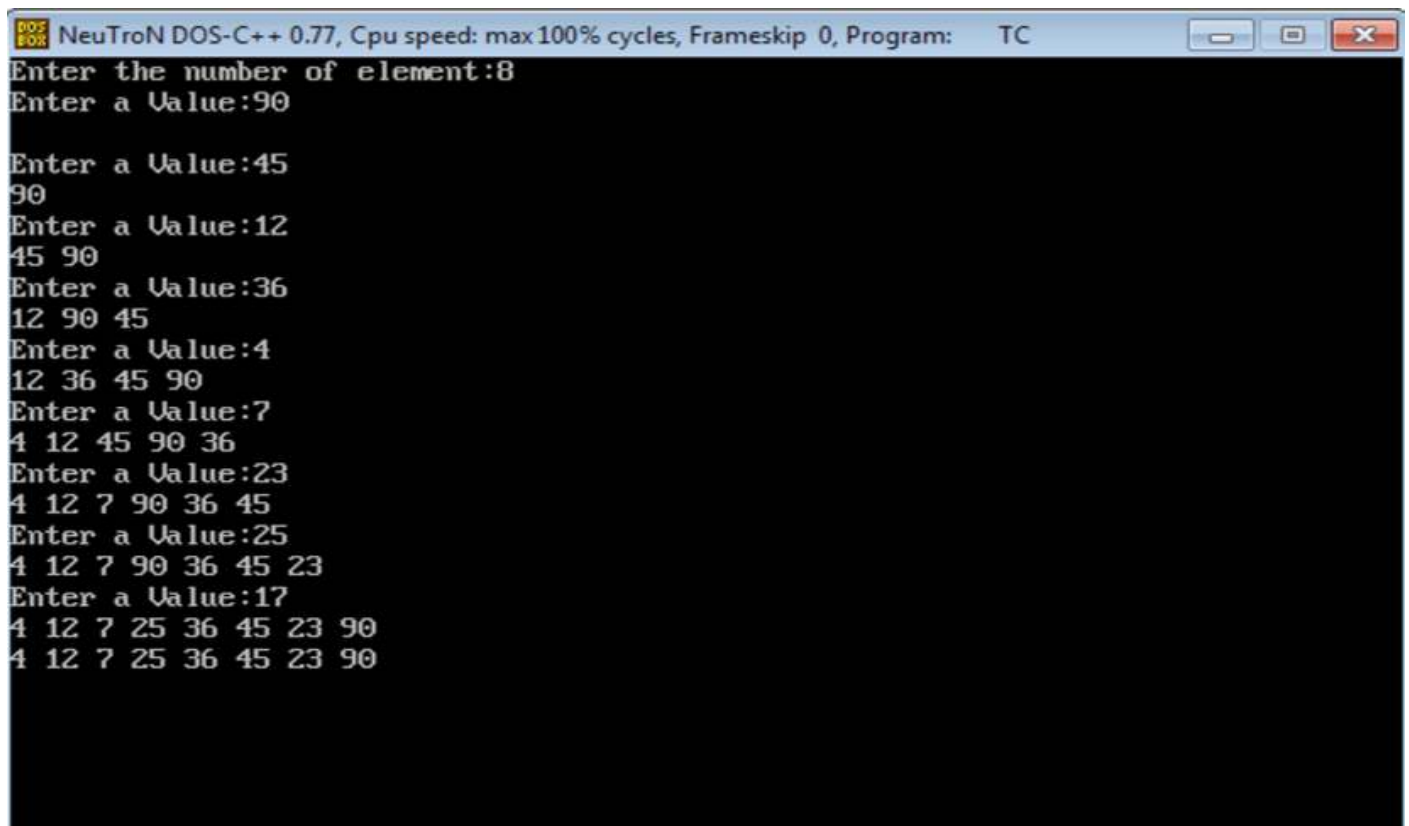
```

r=2*i+2;
if(l<n1 && r<n1)
{
if(a[l]<a[min])
{
min=l;
}
if(a[r]<a[min])
{
min=r;
}
}
else if(l<n1 && r>=n1)
{
if(a[l]<a[min])
{
min=l;
}
}
else if(l>=n1 && r<n1)
{
if(a[r]<a[min])
{
min=r;
}
}
if(i!=min)
{
swap(i,min);
minHeapify(a,min,n1);
}
}

void swap(int i,int j)
{
int temp=a[i];
a[i]=a[j];
a[j]=temp;
}
int length(int a[])
{
int i=0;
while(a[i]!='\0')
{
i++;
}
return i;
}

```

OUTPUT:



```
NeuTroN DOS-C++ 0.77, Cpu speed: max 100% cycles, Frameskip 0, Program: TC
Enter the number of element:8
Enter a Value:90
Enter a Value:45
90
Enter a Value:12
45 90
Enter a Value:36
12 90 45
Enter a Value:4
12 36 45 90
Enter a Value:7
4 12 45 90 36
Enter a Value:23
4 12 7 90 36 45
Enter a Value:25
4 12 7 90 36 45 23
Enter a Value:17
4 12 7 25 36 45 23 90
4 12 7 25 36 45 23 90
```

Implement the following data structure techniques:

PRACTICAL 9

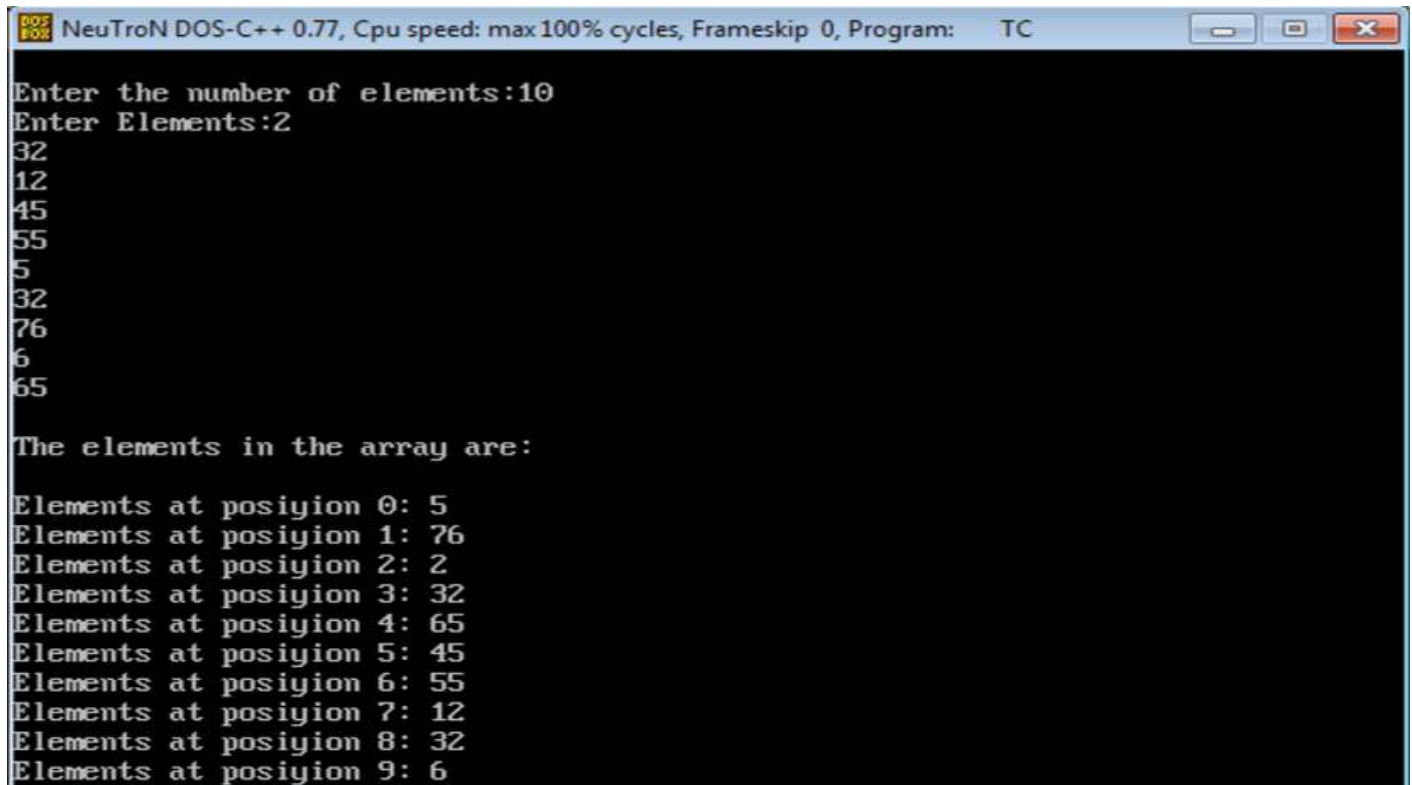
A. Write a program to implement the collision technique.

INPUT:

```
#include<stdio.h>
#include<conio.h>
#define size 10
int hash[20];
int arr[size];
int hasht(int key)
{
    int i;
    i=key % size;
    return i;
}
int rehashq(int key,int j)
{
    int i;
    i=(key+(j*j))%size;
    return i;
}
void main()
{
    int key,i,n,s,op,j,k;
    clrscr();
    printf("\nEnter the number of elements:");
    scanf("%d",&n);
    for(i=0;i<size;i++)
        hash[i]=-1;
    printf("Enter Elements:");
    for(i=0;i<size;i++)
    {
        scanf("%d",&arr[i]);
    }
    for(i=0;i<size;i++)
        hash[i]=-1;
    for(k=0;k<n;k++)
    {
        j=1;
        key=arr[k];
        i=hasht(key);
        while(hash[i]!=-1)
        {
            i=rehashq(i,j);
            j++;
        }
        hash[i]=key;
    }
}
```

```
printf("\nThe elements in the array are:\n");
for(i=0;i<size;i++)
{
printf("\nElements at posiyn %d: %d",i,hash[i]);
}
getch();
}
```

OUTPUT:



```
NeuTroN DOS-C++ 0.77, Cpu speed: max 100% cycles, Frameskip 0, Program: TC
Enter the number of elements:10
Enter Elements:2
32
12
45
55
5
32
76
6
65

The elements in the array are:

Elements at posiyn 0: 5
Elements at posiyn 1: 76
Elements at posiyn 2: 2
Elements at posiyn 3: 32
Elements at posiyn 4: 65
Elements at posiyn 5: 45
Elements at posiyn 6: 55
Elements at posiyn 7: 12
Elements at posiyn 8: 32
Elements at posiyn 9: 6
```

B. Write a program to implement the concept of linear probing.

INPUT:

```
#include<stdio.h>
#define SIZE 10
#define FALSE 0
#define TRUE 1
#define h(x) x%SIZE
int data[SIZE],flag[SIZE],chain[SIZE];
void insert(int x);
int search(int x);
void print();
void deleteeL(int x);
void main()
{
    int i,op,loc,x;
    clrscr();
    for(i=0;i<SIZE;i++)
    {
        flag[i]=FALSE;
        chain[i]=-1;
    }
    do
    {
        printf("\n\n1-Insert\n2-Search\n3-Delete\n4-Print\n5-Quit");
        printf("\nEnter your choice:");
        scanf("%d",&op);
        switch(op)
        {
            case 1:
            {
                printf("\nEnter a number to be insert:");
                scanf("%d",&x);
                insert(x);
                break;
            }
            case 2:
            {
                printf("Enter the number to be searched:");
                scanf("%d",&x);
                if((loc=search(x))!=-1)
                {

                    printf("\nElements not found");
                }
                else
                {
                    printf("\n***Found at location=%d",loc);
                }
                break;
            }
        }
    }
```

```

case 3:
{
printf("Enter the value to be delete:");
scanf("%d",&x);
deleteeL(x);
break;
}
case 4:
{
print();
break;
}
case 5:
{
exit(0);
break;
}
default:
{
printf("\nInvalid Choice..");
}
}
}while(op!=5);
getch();
}
void insert(int x)
{
int i=0,j,start;
start=h(x);
if(flag[start]==0)
{
data[start]=x;
flag[start]=1;
return;
}
i=0;
j=start;
while(flag[j]&& i<SIZE)
{
j=(j+1)%SIZE;
i++;
}
if(i==SIZE)
{
printf("\nTable is full..");
return;
}
data[j]=x;
flag[j]=1;
chain[j]=-1;
i=start;
while(chain[i]!=-1)

```

```

{
i=chain[i];
chain[i]=j;
}
}
int search(int x)
{
int i=0,j;
j=h(x);
while((i<SIZE) && (flag[j]) && (data[j]%SIZE) !=(x%SIZE))
{
i++;
j=(j+1)%SIZE;
}
if((!flag[j]) || (i==SIZE))
{
return(-1);
}
while(j!=-1)
{
if(data[j]==x)
return(j);

j=chain[j];
}
return(-1);
}
void deleteeL(int x)
{
int i=0,j=0,loc;
loc=search(x);
if(loc==-1)
{
printf("Elements not present");
}
else
{
data[loc]=0;
flag[loc]=FALSE;
printf("Values deleted!!");
while(chain[i]!=loc)
{
i++;
}
chain[i]=chain[loc];
chain[loc]=-1;
}
}
void print()
{
int i;
for(i=0;i<SIZE;i++)
{

```



```
if(flag[i])
{
printf("\n(%d)%d %d",i,data[i],chain[i]);
}
else
{
printf("\n(%d)--- %d",i,chain[i]);
}
}
}
```

OUTPUT:

Implement the following data structure techniques:

PRACTICAL 10

A. Write a program to generate the adjacency matrix.

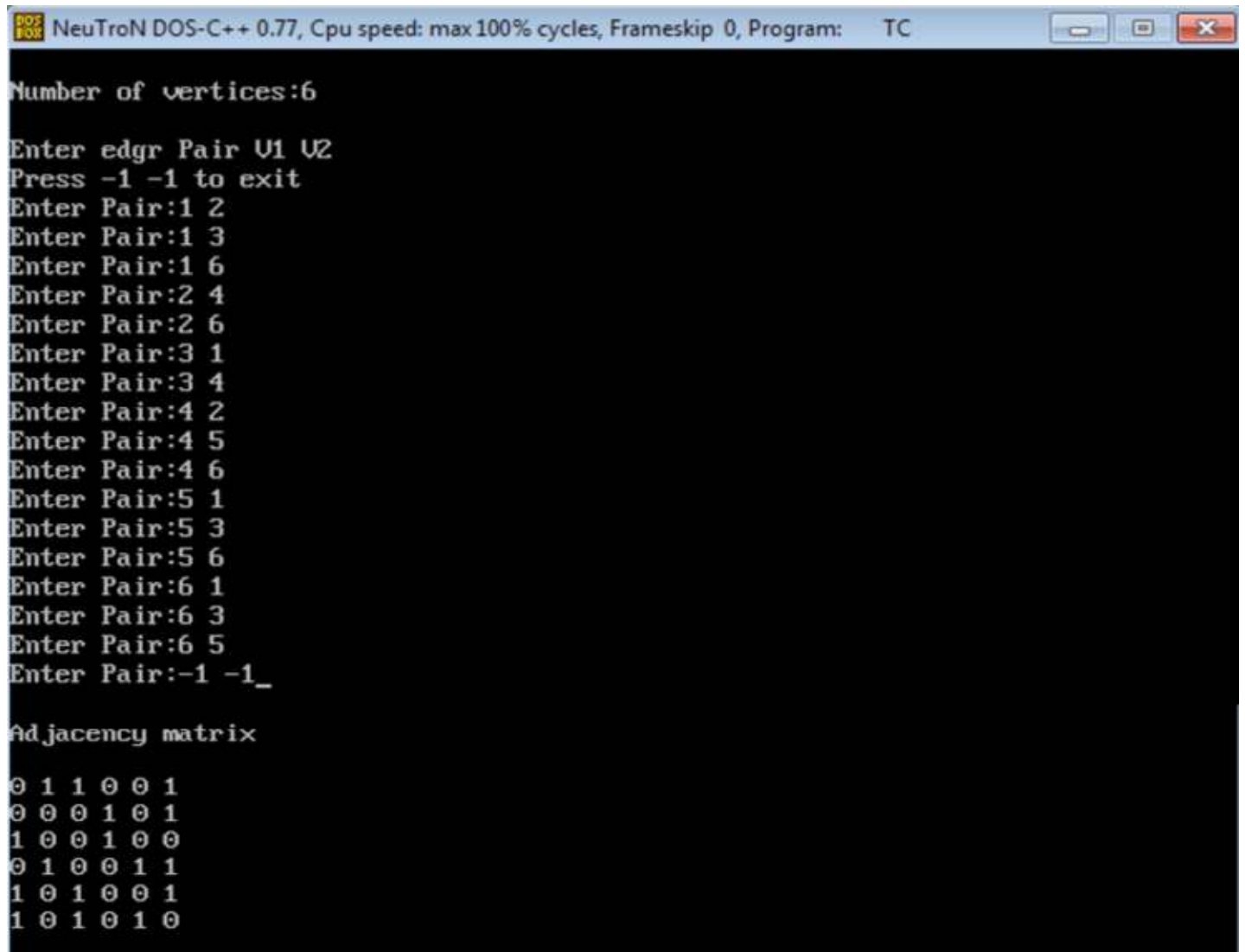
INPUT:

```
#include<stdio.h>
#include<stdlib.h>
void main()
{
int **adjfmatrix;
int r,c,v;
clrscr();
printf("\nNumber of vertices:");
scanf("%d",&v);
adjfmatrix=(int **)malloc(sizeof(int **)*v);
for(r=0;r<v;r++)
{
adjfmatrix[r]=(int *)malloc(sizeof(int)*v);
}
for(r=0;r<v;r++)
{
for(c=0;c<v;c++)
{
adjfmatrix[r][c]=0;
}
}
r=0;
c=0;
printf("\nEnter edge Pair V1 V2\n");
printf("Press -1 -1 to exit\n");
do
{
printf("Enter Pair:");
scanf("%d %d",&r,&c);
if(r>0 && r<=v && c>0 && c<=v)
{
adjfmatrix[r-1][c-1]=1;
}
}while(r>0 && c>0);
printf("\nAdjacency matrix\n");
printf(" ");
printf("\n");

for(r=0;r<v;r++)
{
for(c=0;c<v;c++)
{
printf("%d ",adjfmatrix[r][c]);
}
printf("\n");
}
```

```
}  
getch();  
}
```

OUTPUT:



```
NeuTroN DOS-C++ 0.77, Cpu speed: max 100% cycles, Frameskip 0, Program: TC  
Number of vertices:6  
Enter edge Pair U1 U2  
Press -1 -1 to exit  
Enter Pair:1 2  
Enter Pair:1 3  
Enter Pair:1 6  
Enter Pair:2 4  
Enter Pair:2 6  
Enter Pair:3 1  
Enter Pair:3 4  
Enter Pair:4 2  
Enter Pair:4 5  
Enter Pair:4 6  
Enter Pair:5 1  
Enter Pair:5 3  
Enter Pair:5 6  
Enter Pair:6 1  
Enter Pair:6 3  
Enter Pair:6 5  
Enter Pair:-1 -1_  
Adjacency matrix  
0 1 1 0 0 1  
0 0 0 1 0 1  
1 0 0 1 0 0  
0 1 0 0 1 1  
1 0 1 0 0 1  
1 0 1 0 1 0
```

B. Write a program for shortest path diagram.

INPUT:

```
#include<stdio.h>
#define INF 999
#define SIZE 10
void dijkstra(int A[SIZE][SIZE],int n,int s);
void main()
{
    int A[SIZE][SIZE],i,j,n,u;
    clrscr();
    printf("Enter no of vertices:");
    scanf("%d",&n);
    printf("\nEnter the adjacency matrix:\n");
    for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
        {
            scanf("%d",&A[i][j]);
        }
    }
    printf("\nEnter the string node:");
    scanf("%d",&u);
    dijkstra(A,n,u);
    getch();
}

void dijkstra(int A[SIZE][SIZE],int n,int s)
{
    int cost[SIZE][SIZE],dist[SIZE],parent[SIZE];
    int visited[SIZE],count,mindist,next,i,j;
    for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
        {
            if(A[i][j]==0)
            {
                cost[i][j]=INF;
            }
            else
            {
                cost[i][j]=A[i][j];
            }
        }
    }

    for(i=0;i<n;i++)
    {
        dist[i]=cost[s][i];
        parent[i]=s;
        visited[i]=0;
    }
```

```

dist[s]=0;
visited[s]=1;
count=1;
while(count<n-1)
{
mindist=INF;
for(i=0;i<n;i++)
{
if(dist[i]<mindist&&!visited[i])
{
mindist=dist[i];
next=i;
}
}
visited[next]=1;
for(i=0;i<n;i++)
{
if(!visited[i])
if(mindist+cost[next][i]<dist[i])
{
dist[i]=mindist+cost[next][i];
parent[i]=next;
}
}
count++;
}
for(i=0;i<n;i++)
{
if(i!=s)
{
printf("\nDist of node%d=%d",i,dist[i]);
printf("\nPath=%d",i);
j=i;
do
{
j=parent[j];

printf("<

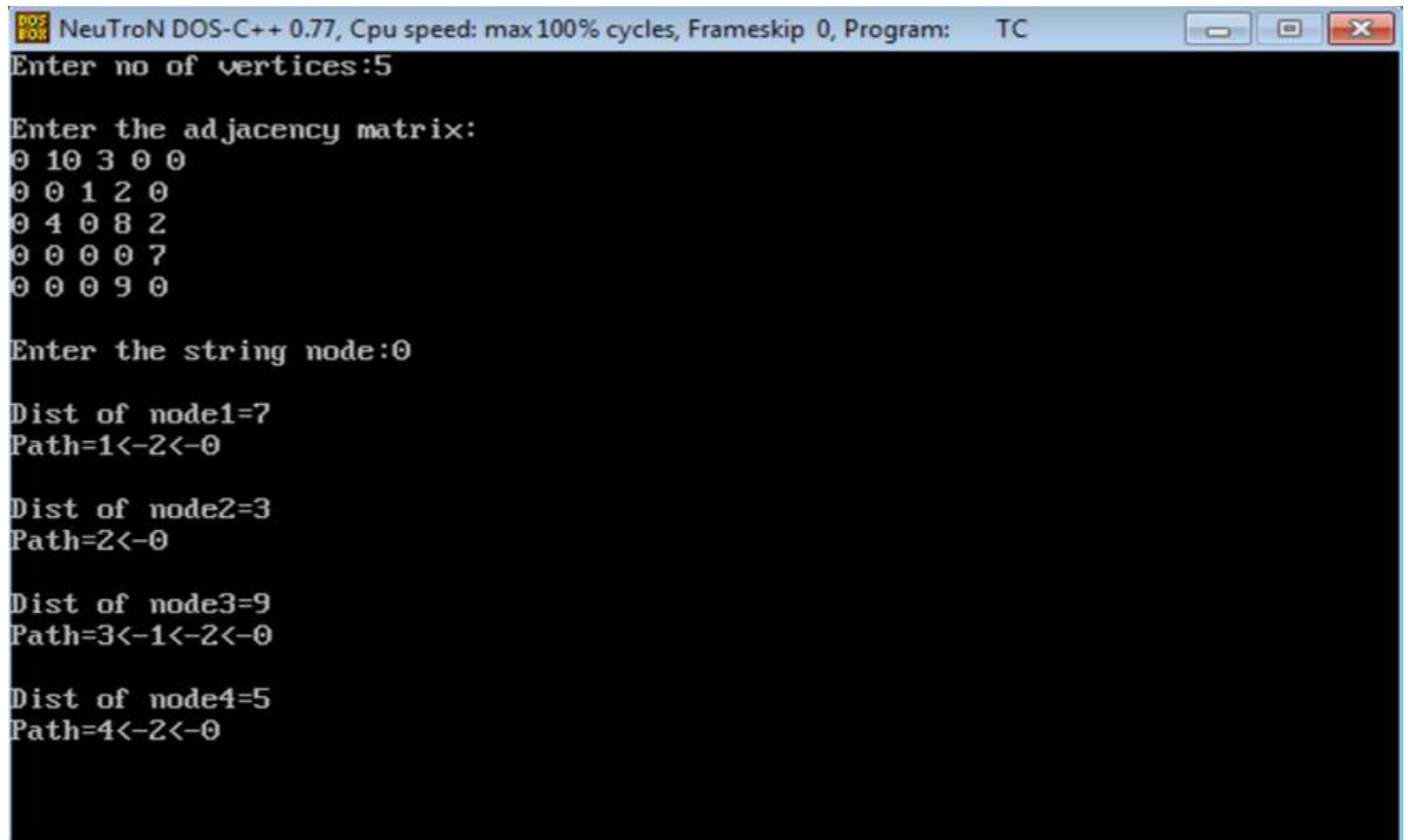
-%d",j);

}while(j!=s);
printf("
\n");

}
}
}

```

OUTPUT:



```
NeuTroN DOS-C++ 0.77, Cpu speed: max 100% cycles, Frameskip 0, Program: TC
Enter no of vertices:5
Enter the adjacency matrix:
0 10 3 0 0
0 0 1 2 0
0 4 0 8 2
0 0 0 0 7
0 0 0 9 0
Enter the string node:0
Dist of node1=7
Path=1<-2<-0
Dist of node2=3
Path=2<-0
Dist of node3=9
Path=3<-1<-2<-0
Dist of node4=5
Path=4<-2<-0
```