Institute of Distance and Open Learning

Vidya Nagari, Kalina, Santacruz East – 400098.

CERTIFICATE

This is to certify that Mr/Ms. ____________________________________ of **Master in Computer Application** (MCA) Semester _______ has completed the specified term work in the subject of ____________________________________ satisfactorily within this institute as laid down by University of Mumbai during the academic year 20___ to 20___.

__________ __________ __________

Subject In-charge External Examiner Coordinator – M.C.A

University of Mumbai

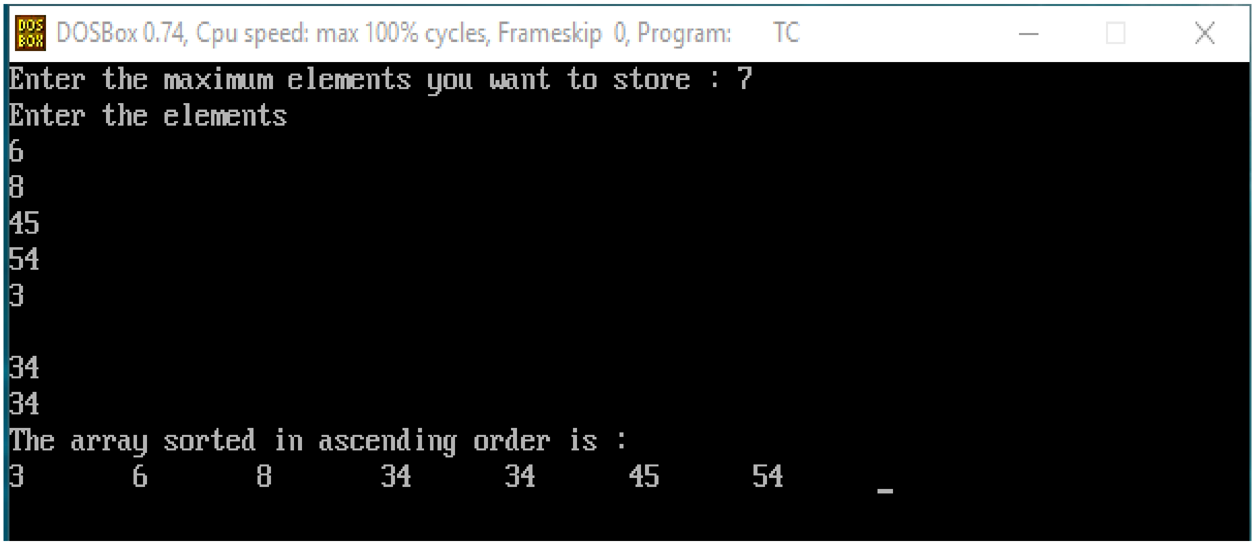
Institute of Distance and Open Learning (IDOL)

PCP CENTER – DTSS Malad(E)

INDEX

Subject: ____

Sr. No.	Experiment Name	Sign.	Remarks
1	Implement program for Bubble sort		
2	Implement program for Insertion sort		
3	Implement program for Selection Sort		
4	Implement program for Shell Sort		
5	Implement program for Radix Sort		
6	Implement program for Linear Search		
7	Implement program for Binary Search		
8	Implement program for Modulo Division		
9	Implement program for Digit Extraction.		
10	Implement program for Fold Shift.		
11	Implement program for Stack usingArrays.		
12	Implement program for Stack using LinkedList.		
13	Implement program for Evaluation of Postfix Expression.		
14	Implement a singly linked list		
15	Implement a doubly linked list		
16	Implement a circular linked list		



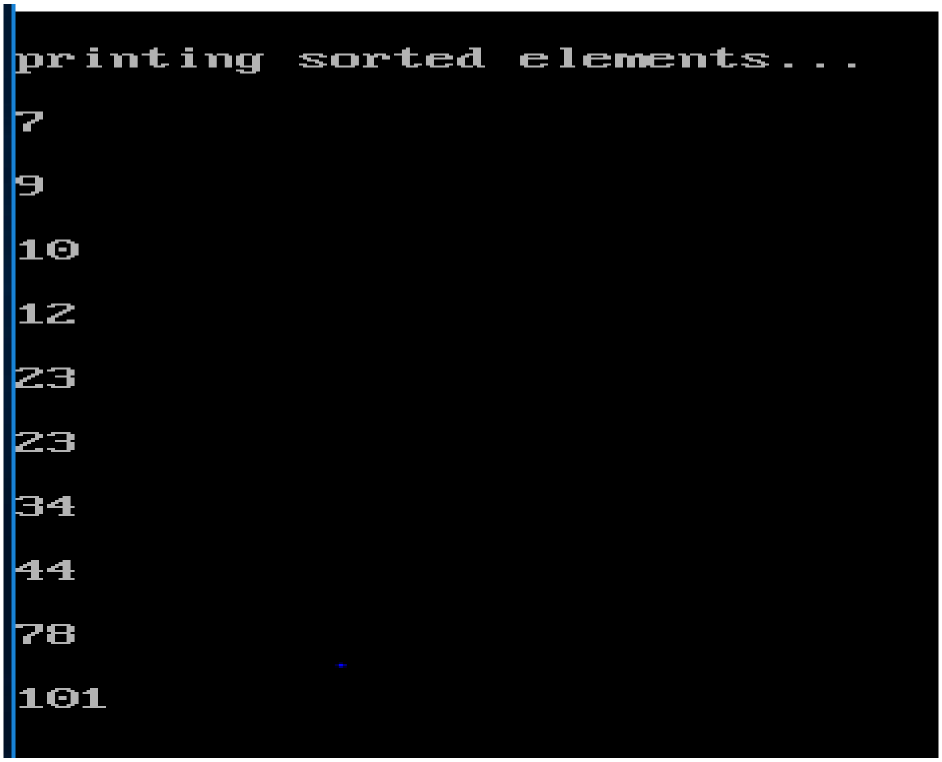
DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Program: TC

```
Enter the maximum elements you want to store : 7
Enter the elements
6
8
45
54
3

34
34
The array sorted in ascending order is :
3       6       8       34      34      45      54      _
```

Practical No : 1

Aim : Implement program for Bubble sort

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
 int I, n, temp, j, arr[10];
 printf(“Enter the maximum elements you want to store : “);
 scanf(“%d”, &n);
 printf(“Enter the elements \n”);
 for(i=0;i<n;i++)
 {
 scanf(“%d”, & arr[i]);
 }
for(i=0;i<n;i++)
 {
 for(j=0;j<n-1;j++)
 {
 if(arr[j]>arr[j+1])
 {
 temp = arr[j];
 arr[j] = arr[j+1];
 arr[j+1] = temp;
 }
 }
 }
 printf(“The array sorted in ascending order is :\n”);
 for(i=0;i<n;i++)
 printf(“%d\t”, arr[i]);
 getch();
 return 0;
}
```

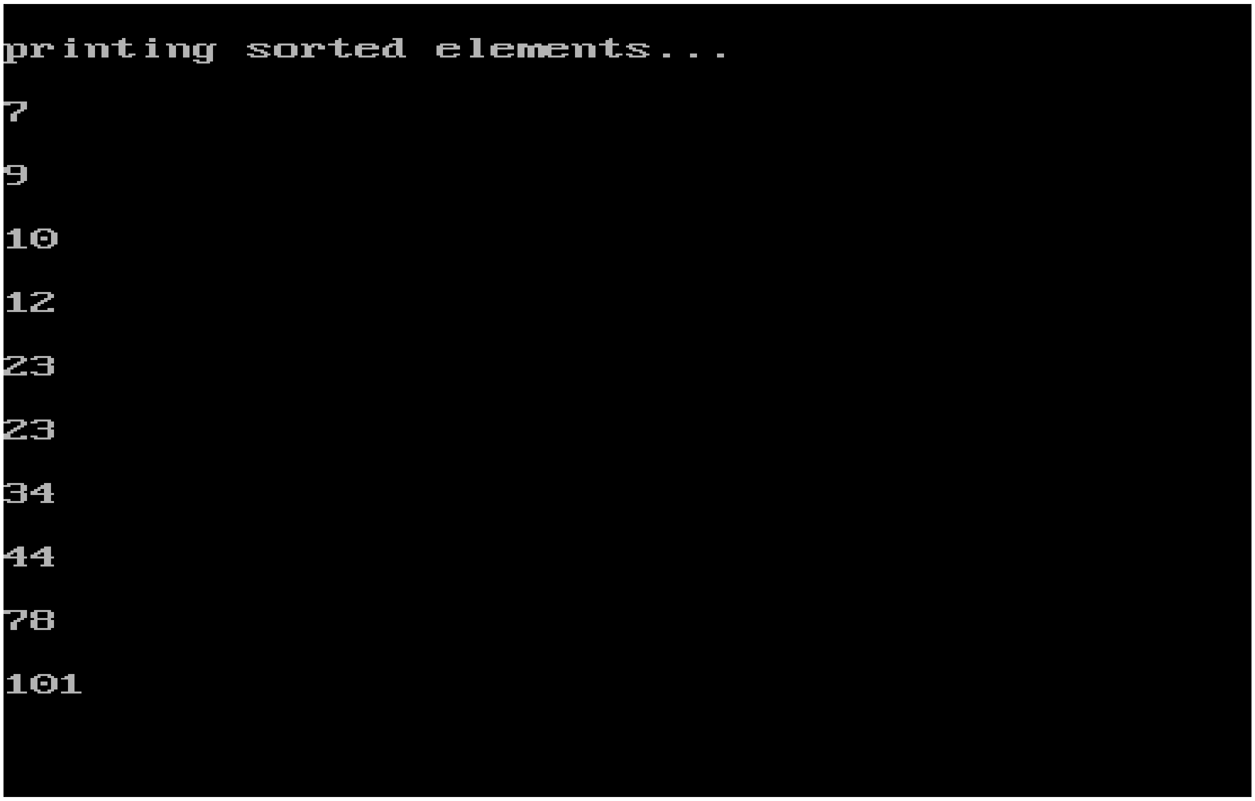


```
printing sorted elements...
7
9
10
12
23
23
34
44
78
101
```

Practical No : 2

Aim : Implement program for Insertion sort

```
#include<stdio.h>
# include<conio.h>
void main()
{
int I,j,k,temp;
int a[10]=
{10,9,7,101,23,44,12,78,34,23};
printf (“\n printing sorted
elements...\n”);
for (k=1;k<10;k++)
{temp = a[k];
j=k-1;
while (j>=0&&temp<=a[j])
{a[j+1]= a[j];
j=j-1;

}
a[j+1]=temp;

}
for (i=0;i<10;i++)
{printf(“\n%d\n”,a[i]);
}
getch();

}
```



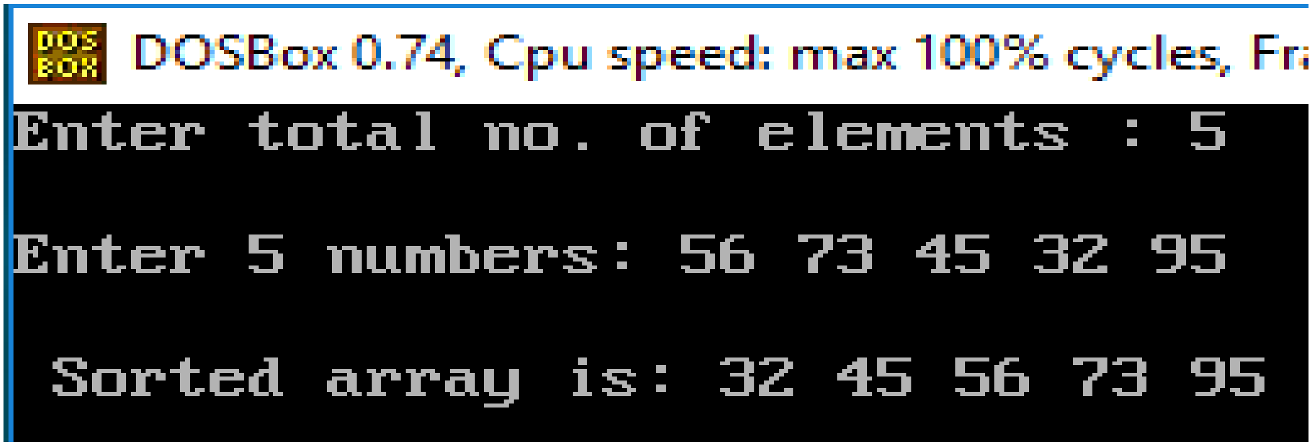
```
printing sorted elements...
7
9
10
12
23
23
34
44
78
101
```

Practical No : 3

Aim : Implement program for Selection Sort

```
#include<stdio.h>
# include<conio.h>
int smallest(int[],int,int);
void main()
{
int
a[10]={10,9,7,101,23,44,12,78,34,23};
int i,j,k,pos,temp;
for(i=0;i<10;i++){
pos=smallest(a,10,i);
temp = a[i];
a[i] = a[pos];
a[pos] = temp;
}
printf("\nprinting sorted element\n");
for(i=0;i<10;i++){
printf("%d\n",a[i]);

}
}
```

```
int smallest(int a[],int
n,int i){
 int small, pos, j;
 small =a[i];
 pos=i;
 for(j=i+1;j<10;j++){
 if(a[j]<small){
 small=a[j];
 pos=j;

 }
 }
 getch();
 return pos;
}
```



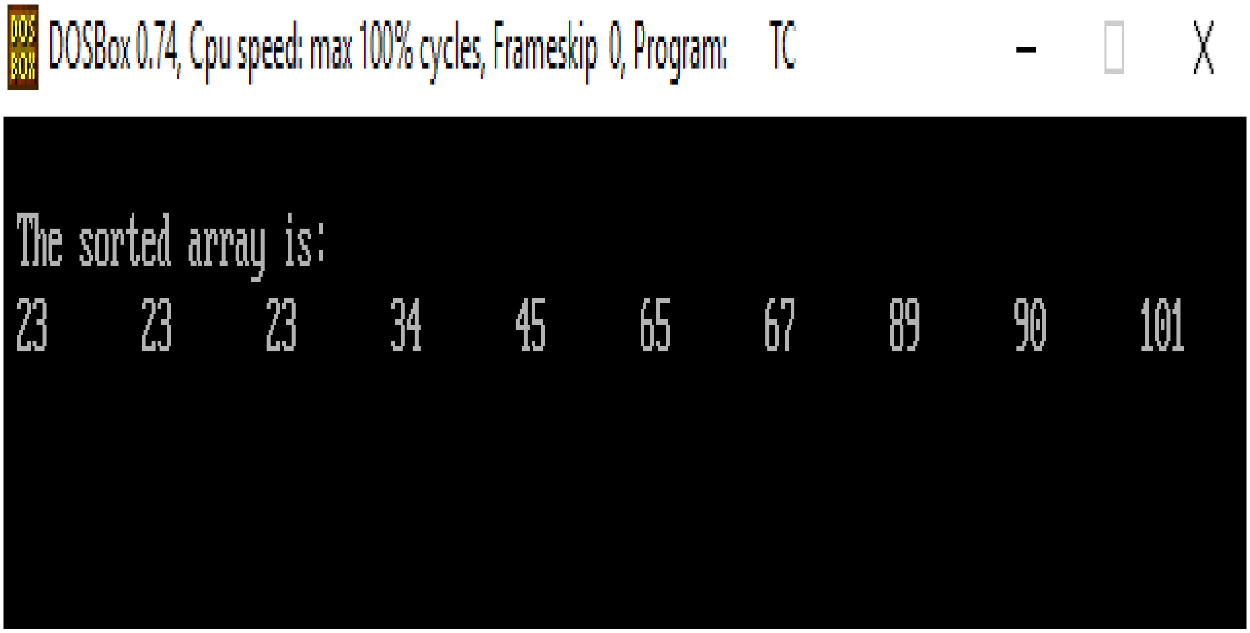
```
Enter total no. of elements : 5

Enter 5 numbers: 56 73 45 32 95

 Sorted array is: 32 45 56 73 95
```

Practical No : 4

Aim : Implement program for Shell Sort

```
#include <stdio.h>
#include <conio.h>
void shellsort(int arr[], int num)
{
int i, j, k, tmp;
for (i = num / 2; i > 0; i = i / 2)
{
for (j = i; j < num; j++)
{
for(k = j - i; k >= 0; k = k - i)
{
if (arr[k+i] >= arr[k])
break;
else
{
tmp = arr[k];
arr[k] = arr[k+i];
arr[k+i] = tmp;
}
}
```

```
int main()
{
int arr[30];
int k, num;
printf("Enter total no. of elements : ");
scanf("%d", &num);
printf("\nEnter %d numbers: ", num);
for (k = 0 ; k < num; k++)
{
scanf("%d", &arr[k]);
}
shellsort(arr, num);
printf("\n Sorted array is: ");
for (k = 0; k < num; k++)
printf("%d ", arr[k]);
return 0;
}
}
}
}
```



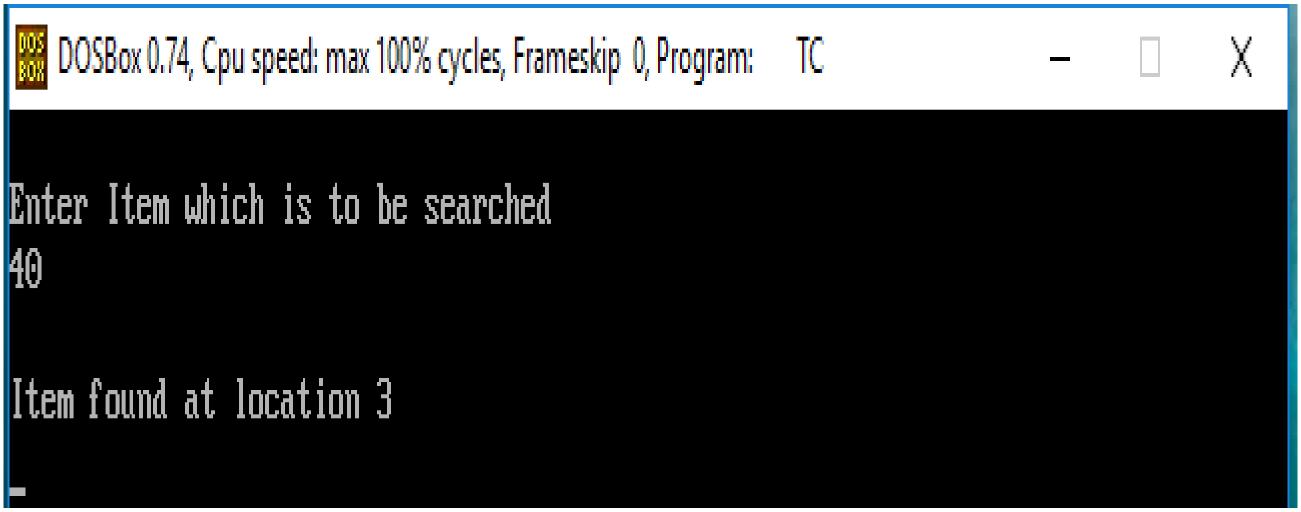
DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Program: TC

```
The sorted array is:
23      23      23      34      45      65      67      89      90      101
```

Practical No : 5

Aim : Implement program for Radix Sort

```
#include <stdio.h>
#include <conio.h>
int largest(int a[]);
void radix_sort(int a[]);
void main()
{
int i;
 int a[10]={90,23,101,45,65,23,67,89,34,23};
 radix_sort(a);
 printf("\n The sorted array is: \n");
 for(i=0;i<10;i++)
 printf(" %d\t", a[i]);
}
int largest(int a[])
{
 int larger=a[0], i;
 for(i=1;i<10;i++)
 {
 if(a[i]>larger)
 larger = a[i];
 }
 return larger;
}
void radix_sort(int a[])
{
 int bucket[10][10], bucket_count[10];
 int i, j, k, remainder, NOP=0, divisor=1, larger, pass;
larger = largest(a);
 while(larger>0)
 {
 NOP++;
 larger/=10;
 }
 for(pass=0;pass<NOP;pass++) // Initialize the buckets
 {
 for(i=0;i<10;i++)
 bucket_count[i]=0;
 for(i=0;i<10;i++)
 {
 // sort the numbers according to the digit at passth place
 remainder = (a[i]/divisor)%10;

bucket[remainder][bucket_count[remainder]] = a[i];
 bucket_count[remainder] += 1;
 }
 // collect the numbers after PASS pass
 i=0;
 for(k=0;k<10;k++)
 {
 for(j=0;j<bucket_count[k];j++)
{
 a[i] = bucket[k][j];
 i++;
 }
 }
 divisor *= 10;
}
getch();
}
```



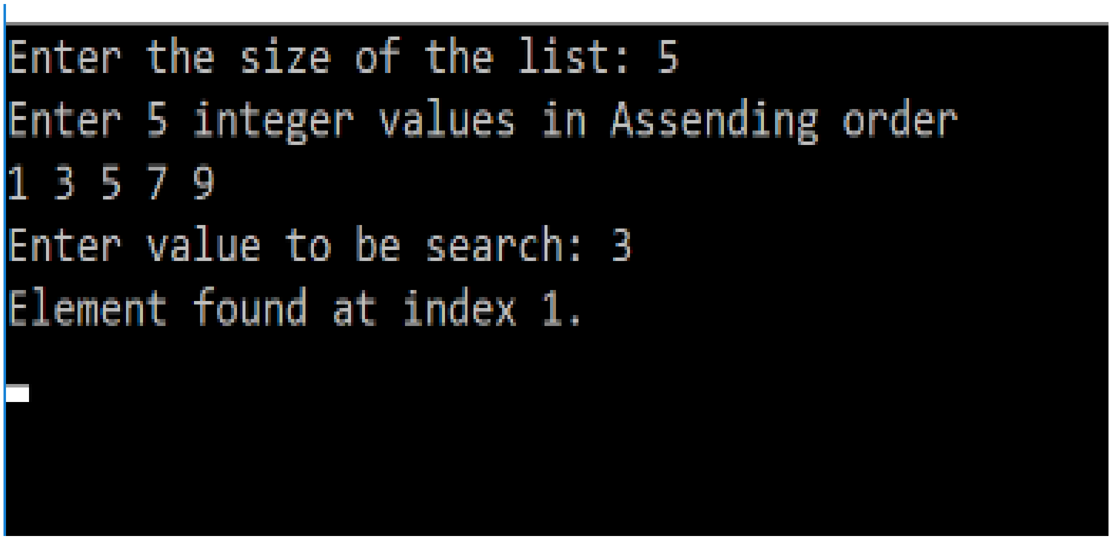
DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Program: TC

```
Enter Item which is to be searched
40

Item found at location 3

_
```

Practical No : 6

Aim : Implement   program for Linear Search

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int a[10] = {10, 23, 40, 1, 2, 0,
14, 13, 50, 9};
int item, i, flag;
printf("\nEnter Item which is
to be searched\n");
scanf("%d",&item);
for (i = 0; i< 10; i++)
{
if(a[i] == item)
{
flag = i+1;
break;
}
else
{
flag = 0;
}
}
if(flag != 0)
{
printf("\nItem found at
location %d\n",flag);
}
else
{
printf("\nItem not found\n");
}
getch();
}
```

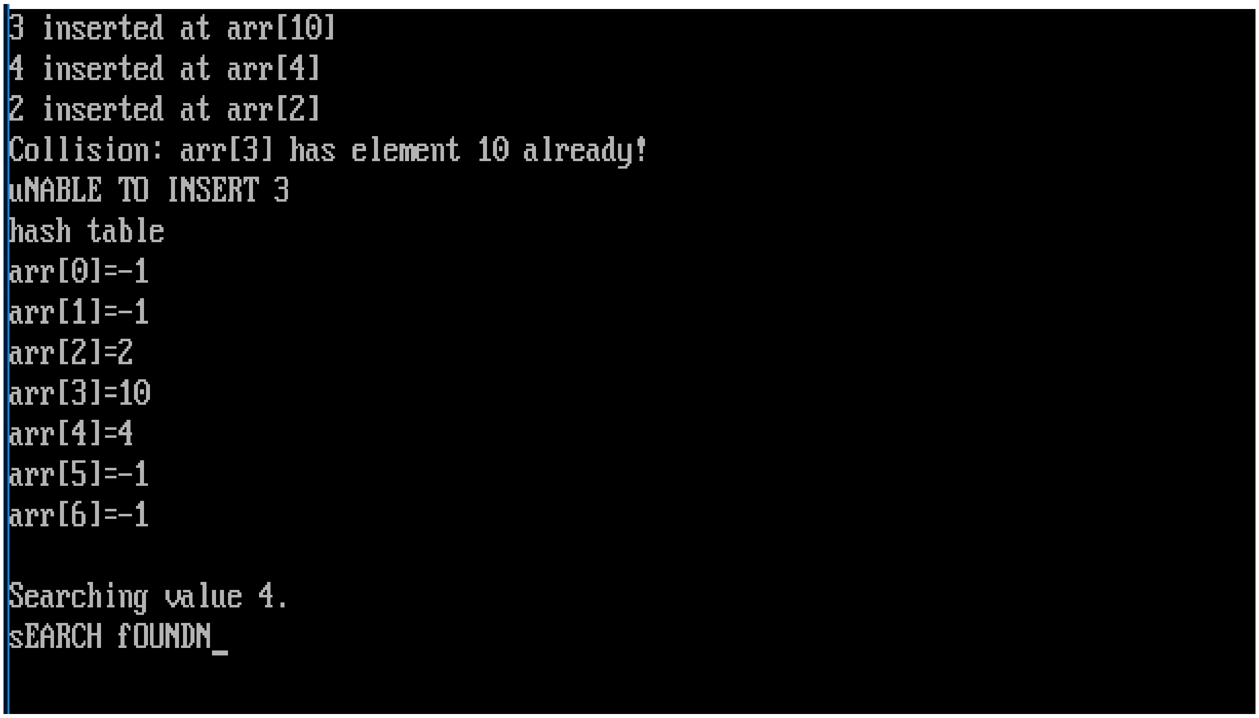


```
Enter the size of the list: 5
Enter 5 integer values in Assending order
1 3 5 7 9
Enter value to be search: 3
Element found at index 1.
```

Practical No : 7

Aim : Implement program for Binary Search.

```
#include<stdio.h>
#include<conio.h>
void main()
{
   int first, last, middle, size, i, sElement, list[100];
   clrscr();

   printf("Enter the size of the list: ");
   scanf("%d",&size);

   printf("Enter %d integer values in Assending order\n", size);

   for (i = 0; i < size; i++)
      scanf("%d",&list[i]);

   printf("Enter value to be search: ");
   scanf("%d", &sElement);

   first = 0;
   last = size - 1;

   middle = (first+last)/2;

   while (first <= last) {
      if (list[middle] < sElement)
         first = middle + 1;
      else if (list[middle] == sElement) {
         printf("Element found at index %d.\n",middle);
         break;
      }
      else
         last = middle - 1;

      middle = (first + last)/2;
   }
   if (first > last)
      printf("Element Not found in the list.");
   getch();
}
```

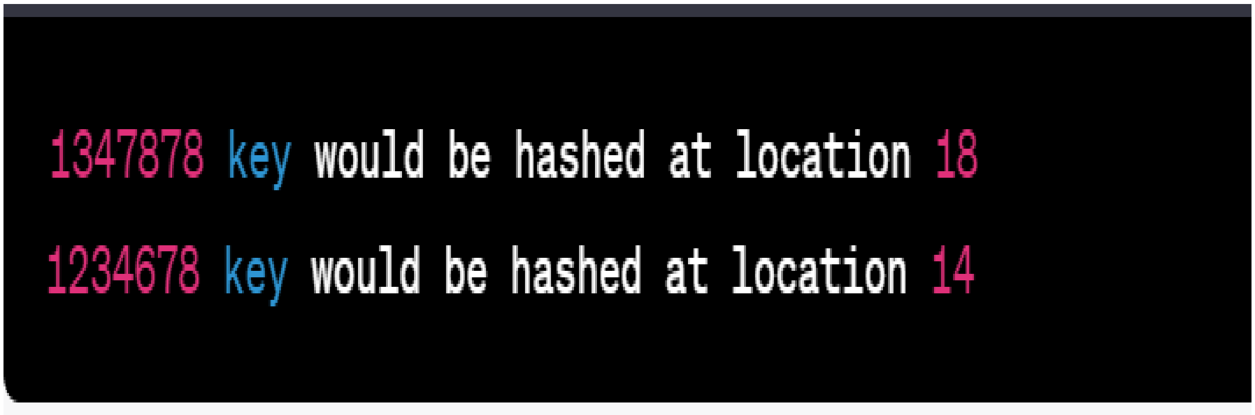


```
3 inserted at arr[10]
4 inserted at arr[4]
2 inserted at arr[2]
Collision: arr[3] has element 10 already!
uNABLE TO INSERT 3
hash table
arr[0]=-1
arr[1]=-1
arr[2]=2
arr[3]=10
arr[4]=4
arr[5]=-1
arr[6]=-1

Searching value 4.
sEARCH fOUNDN_
```

Practical No : 8

Aim : Implement program for Modulo Division

```
#include<stdio.h>
#include<conio.h>
#define size 7
int arr[size];
void init()
{
int i;
for(i=0;i<size;i++){
arr[i]=-1;
}
}
void insert(int value)
{
int key = value%size;
if(arr[key]==-1)
{
 arr[key]=value;
 printf("%d inserted at arr[%d]\n",key,arr[key]);

}
else{
printf("Collision: arr[%d] has element %d already!\n",key,arr[key]);
printf("uNABLE TO INSERT %d\n",value);
}
}
void search(int value)
{
int key = value%size;
if(arr[key]==value)
{
printf("sEARCH fOUND\N");
}
else{
printf("Search not found\n");
}
}

void display(){
int i;
for(i=0;i<size;i++)
{
printf("arr[%d]=%d\n",i,arr[i]);
}
}
int main(){
init();
insert(10);
insert(4);
insert(2);
insert(3);
printf("hash table\n");
display();
printf("\n");
printf("Searching value 4.\n");
search(4);
getch();
return 0;
}
```

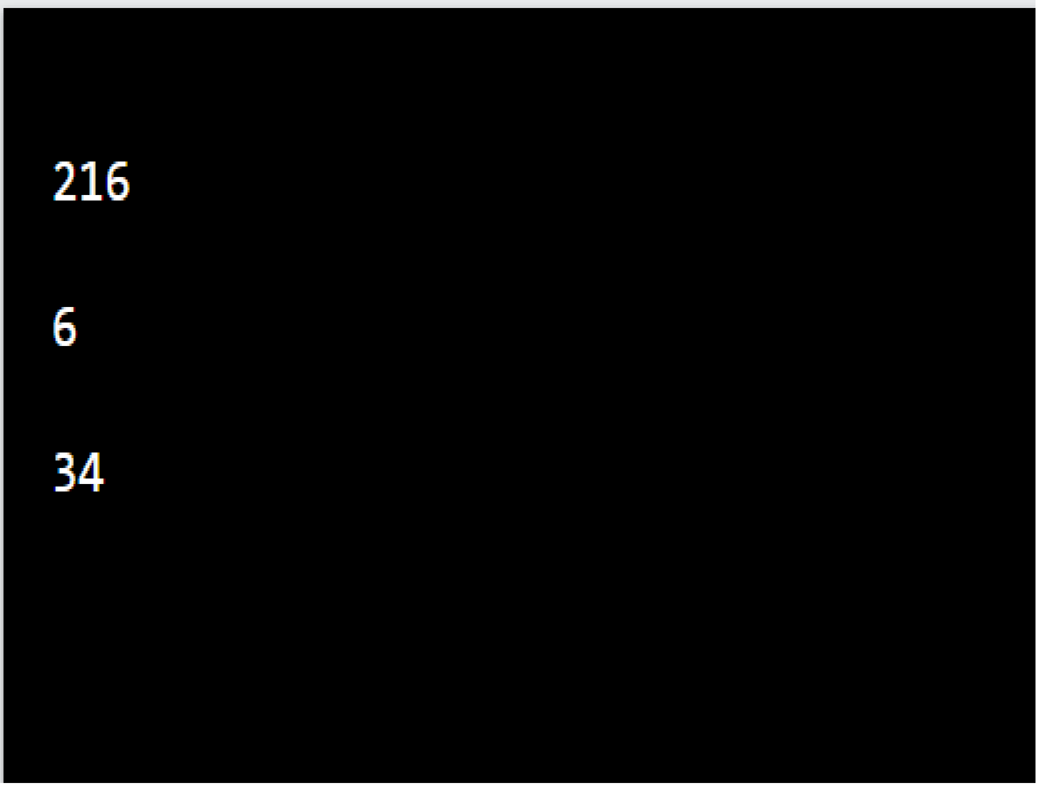


```
1347878 key would be hashed at location 18
1234678 key would be hashed at location 14
```

Practical No : 9

Aim: Implement program for Digit Extraction.

```
#include<stdio.h>

void digit_extraction(int key)
{
   int first_digit = key % 10000000 / 1000000;
   int fourth_digit = key % 10000 / 100;
   printf("%d key would be hashed at location %d%d\n", key, first_digit, fourth_digit);
}

int main()
{
   digit_extraction(1347878); //18
   digit_extraction(1234678); //14
   return 0;
```

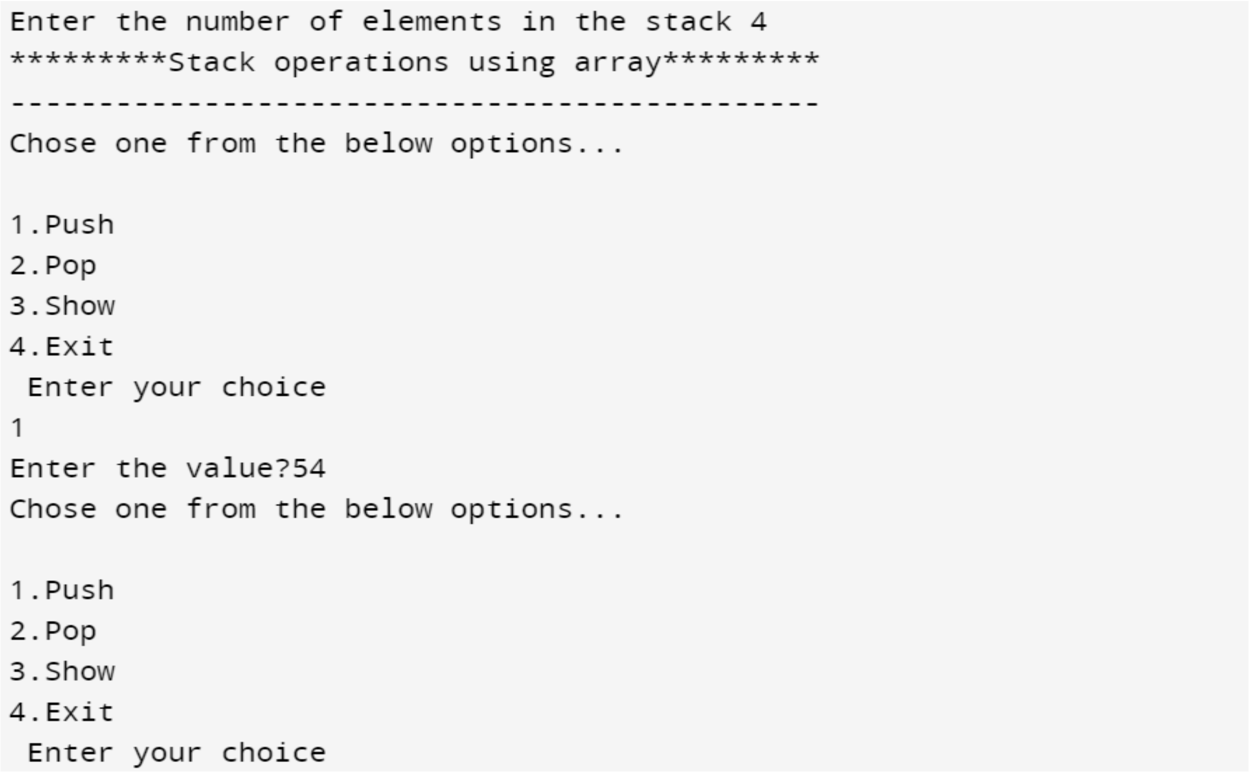


```
216

6

34
```

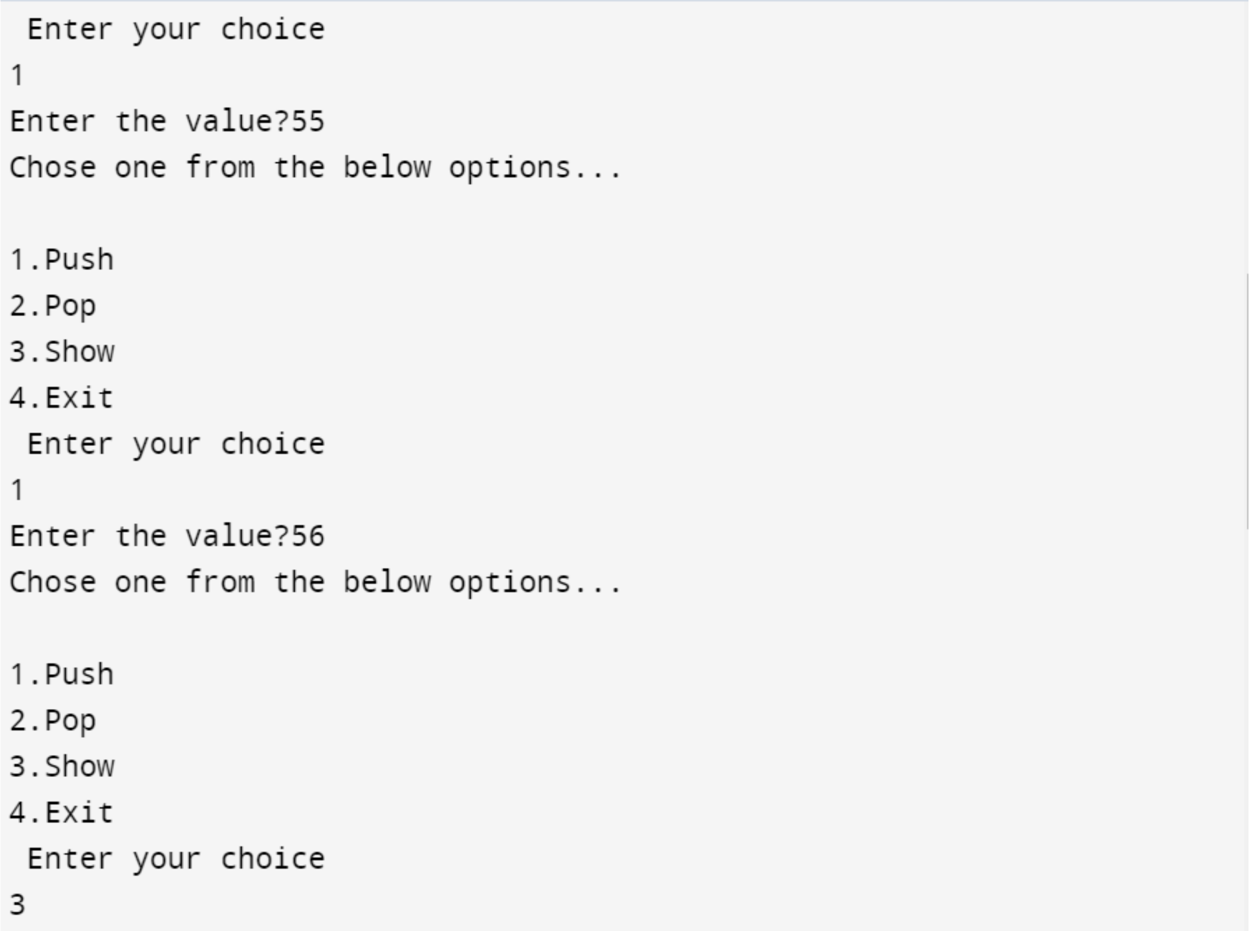
Practical No : 10

Aim: Implement program for Fold Shift.

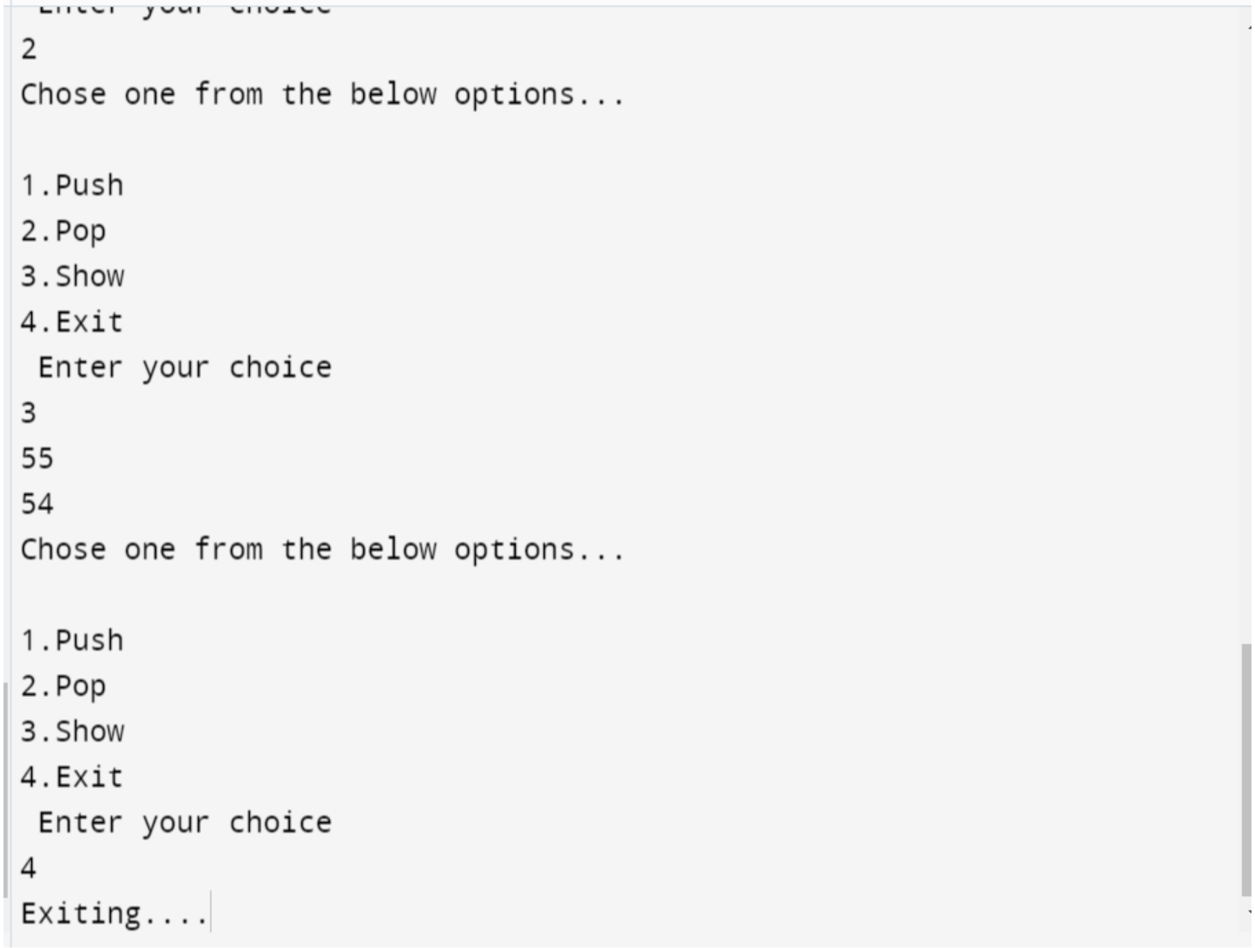
```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#include <math.h>
int count_digits(int key)
{
int count=0;
 while(key != 0)
 {
 key /= 10;
 ++count;
 }
return count;
}
int fold_shift(int key, int size)
{
int key_roll=key;
int key_sum=0;
int key_frac=0;
int key_length=0;
int fraction = size;
key_length =
count_digits(key_roll);
while (key_length > 0)
{
 if (key_length >fraction)
 {
 key_frac = key_roll / (int)pow(10,
(key_length - fraction));
 key_sum += key_frac;
 key_roll = key_roll % (int)pow(10,
(key_length -
fraction));
 key_length = key_length -
fraction;
 }
 else
 {
 key_sum += key_roll;
 break;
 }
}
return key_sum % (int)pow(10,
(fraction));
}
int main()
{
clrscr();
printf("\n\n%d",fold_shift(12789,
3)); //216
printf("\n\n%d",fold_shift(123456
78, 1)); //6
printf("\n\n%d",fold_shift(5678,
2)); //34
getch();
return 0;
}
```



```
Enter the number of elements in the stack 4
*********Stack operations using array*********
----------------------------------------------
Chose one from the below options...

1.Push
2.Pop
3.Show
4.Exit
 Enter your choice
1
Enter the value?54
Chose one from the below options...

1.Push
2.Pop
3.Show
4.Exit
 Enter your choice
```



```
 Enter your choice
1
Enter the value?55
Chose one from the below options...

1.Push
2.Pop
3.Show
4.Exit
 Enter your choice
1
Enter the value?56
Chose one from the below options...

1.Push
2.Pop
3.Show
4.Exit
 Enter your choice
3
```

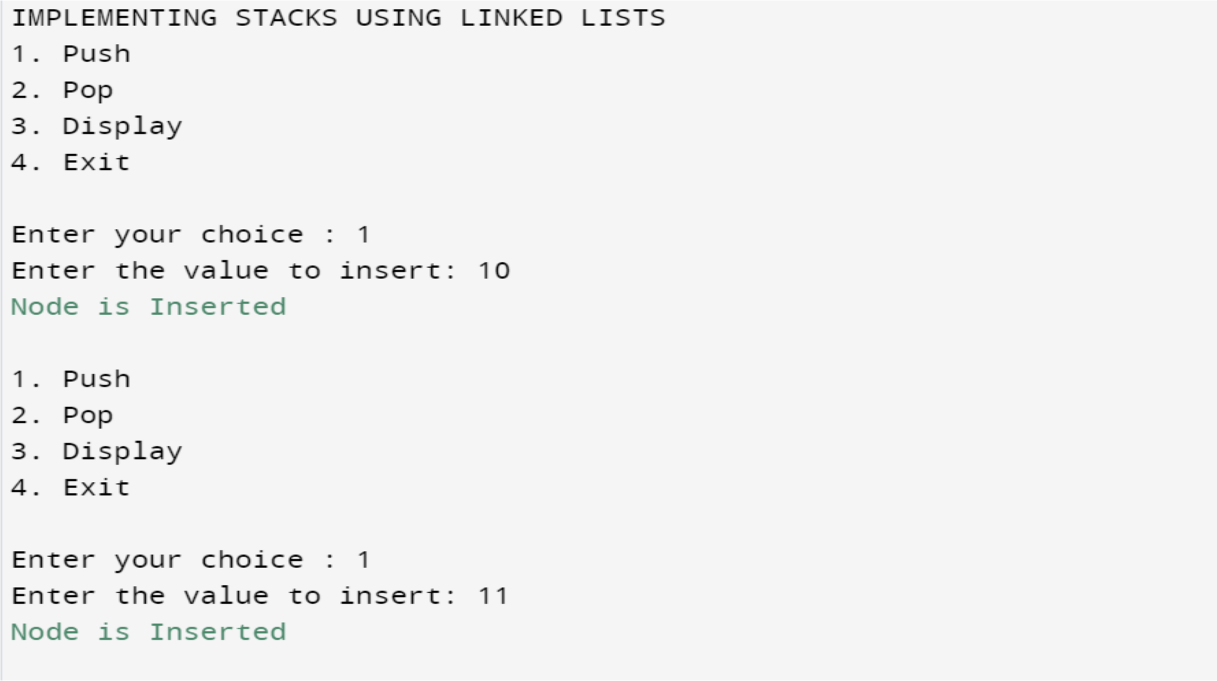


```
 Enter your choice
2
Chose one from the below options...

1.Push
2.Pop
3.Show
4.Exit
 Enter your choice
3
55
54
Chose one from the below options...

1.Push
2.Pop
3.Show
4.Exit
 Enter your choice
4
Exiting....
```

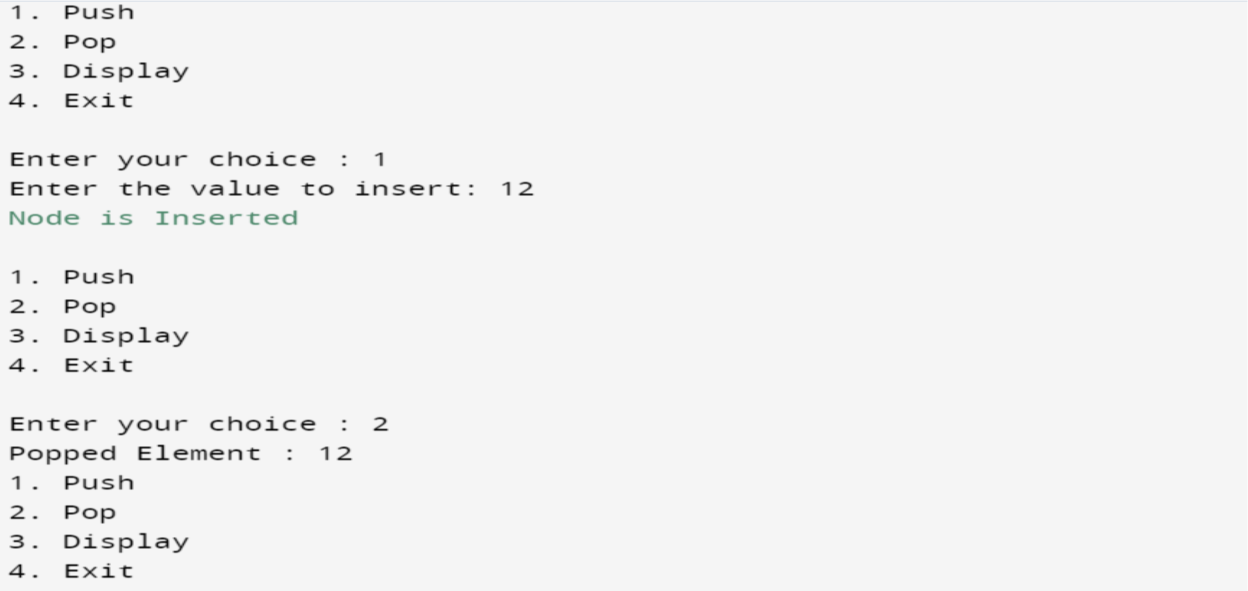
Practical No : 11

Aim: Implement program for Stack using Arrays.

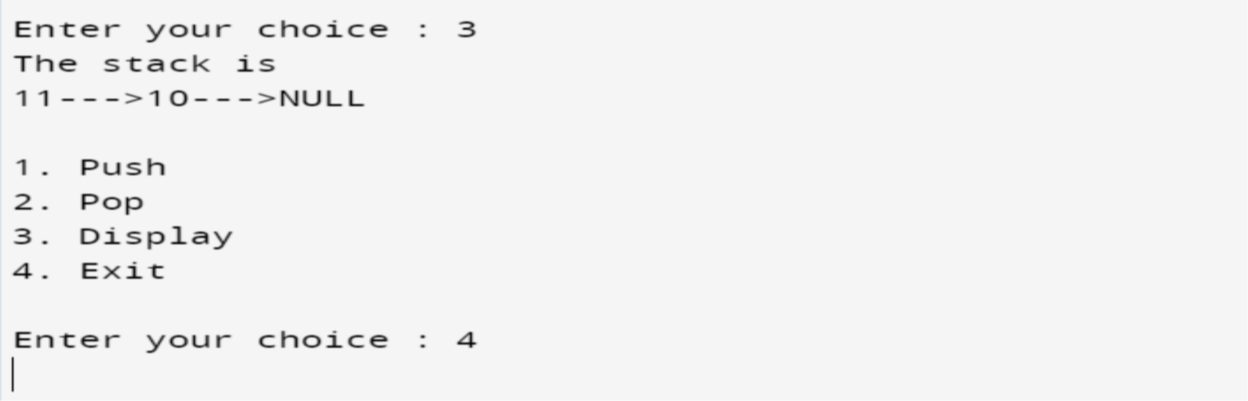
```
#include <stdio.h>
int stack[100],i,j,choice=0,n,top=-1;
void push();
void pop();
void show();
void main ()
{

    printf("Enter the number of elements in the
stack ");
    scanf("%d",&n);
    printf("*********Stack operations using
array*********");
printf("\n----------------------------------------------
\n");
    while(choice != 4)
    {
        printf("Chose one from the below
options...\n");
        printf("\n1.Push\n2.Pop\n3.Show\n4.Exit");
        printf("\n Enter your choice \n");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1:
            {
                push();
                break;
            }
            case 2:
            {
                pop();
                break;
            }
            case 3:
{
                show();
                break;
            }
            case 4:
            {
printf("Exiting....");
                break;
            }

    default:
            {
                printf("Please Enter valid
choice ");
            }
        };
    }
}
void push ()
{
    int val;
    if (top == n )
    printf("\n Overflow");
    else
    {
        printf("Enter the value?");
        scanf("%d",&val);
        top = top +1;
        stack[top] = val;
    }
}

void pop ()
{
    if(top == -1)
    printf("Underflow");
    else
    top = top -1;
}
void show()
{
    for (i=top;i>=0;i--)
    {
        printf("%d\n",stack[i]);
    }
    if(top == -1)
    {
        printf("Stack is empty");
    }
}
```



```
IMPLEMENTING STACKS USING LINKED LISTS
1. Push
2. Pop
3. Display
4. Exit

Enter your choice : 1
Enter the value to insert: 10
Node is Inserted

1. Push
2. Pop
3. Display
4. Exit

Enter your choice : 1
Enter the value to insert: 11
Node is Inserted
```



```
1. Push
2. Pop
3. Display
4. Exit

Enter your choice : 1
Enter the value to insert: 12
Node is Inserted

1. Push
2. Pop
3. Display
4. Exit

Enter your choice : 2
Popped Element : 12
1. Push
2. Pop
3. Display
4. Exit
```

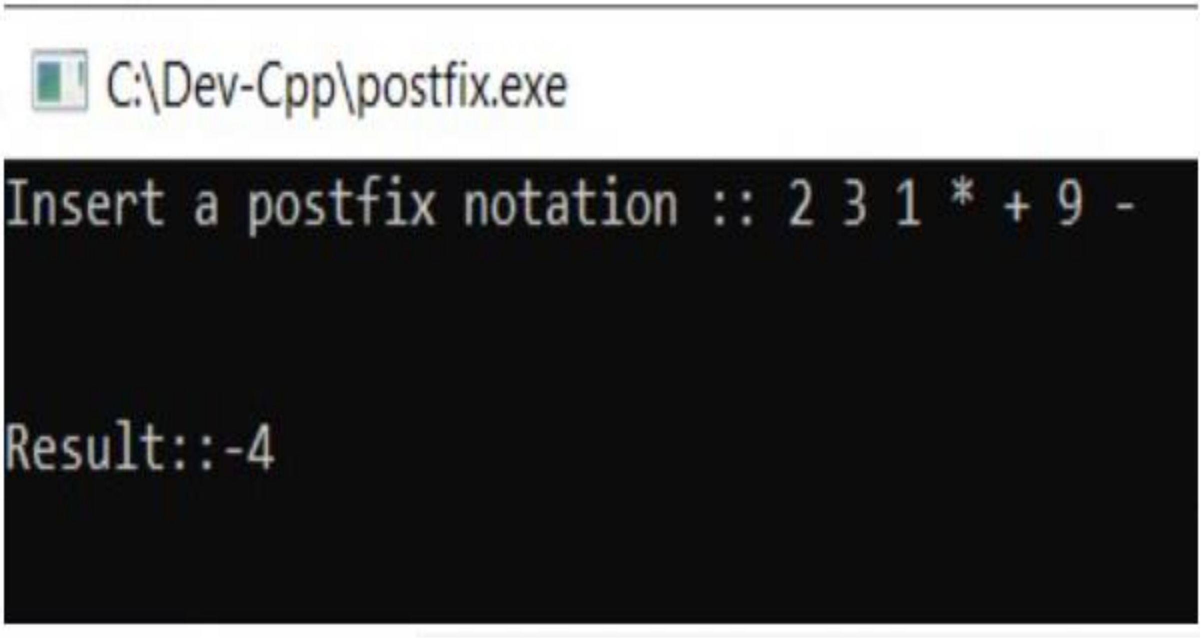


```
Enter your choice : 3
The stack is
11--->10--->NULL

1. Push
2. Pop
3. Display
4. Exit

Enter your choice : 4
```

Practical No : 12

Aim: Implement program for Stack using Linked List.

```
#include<stdio.h>
#include<stdlib.h>
/* Structure to create a node with data and pointer */
struct Node
{
int data;
struct Node *next;
}
*top = NULL; // Initially the list is empty
void push(int);
void pop();
void display();
int main()
{
 int choice, value;
 printf("\nIMPLEMENTING STACKS USING LINKED LISTS\n");
while(1)
{
printf("1. Push\n2. Pop\n3. Display\n4. Exit\n");
printf("\nEnter your choice : ");
scanf("%d",&choice);
switch(choice)
{
case 1: printf("\nEnter the value to insert: ");
scanf("%d", &value);
push(value);
break;

case 2: pop();
break;
case 3: display();
break;
case 4: exit(0);
break;
default: printf("\nInvalid Choice\n");
}
}
}
void push(int value)
{
struct Node *newNode;
newNode = (struct Node*)malloc(sizeof(struct Node));
newNode->data = value; // get value for the node
if(top == NULL)
newNode->next = NULL;
else
newNode->next = top; // Make the node as TOP
top = newNode;
printf("Node is Inserted\n\n");
}
void pop()
{
if(top == NULL)
printf("\nEMPTY STACK\n");
else{
struct Node *temp = top;
printf("\nPopped Element : %d", temp->data);
printf("\n");
top = temp->next; // After popping, make the next node as TOP
free(temp);
}
}
void display()
{
if(top == NULL)
printf("\nEMPTY STACK\n");
else

{
printf("The stack is \n");
struct Node *temp = top;
while(temp->next != NULL){
printf("%d--->",temp->data);
temp = temp -> next;
}
printf("%d--->NULL\n\n",temp->data);
}
}
```

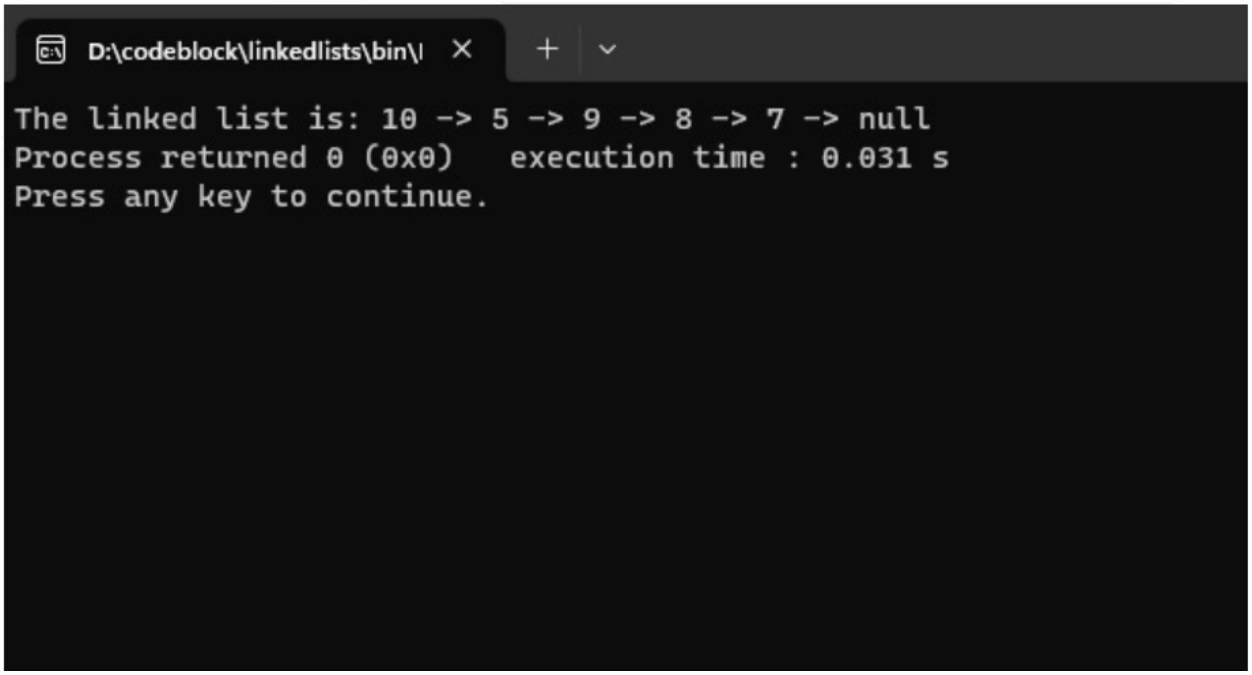


```
Insert a postfix notation :: 2 3 1 * + 9 -


Result::-4
```

Practical No : 13

Aim: Implement program for Evaluation of Postfix Expression.

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#define MAX 50 //max size defined
int stack[MAX]; //a global stack
char post[MAX]; //a global postfix stack
int top=-1; //initializing top to -1
void pushstack(int tmp); //push function
void evaluate(char c); //calculate function
void main()
{
 int i,l;
 //clrscr();
 printf("Insert a postfix notation :: ");
 gets(post); //getting a postfix expression
 l=strlen(post); //string length
 for(i=0;i<l;i++)
 {
 if(post[i]>='0' && post[i]<='9')
 {
 pushstack(i); //if the element is a number
push it
 }
 if(post[i]=='+' || post[i]=='-' || post[i]=='*' ||
 post[i]=='/' || post[i]=='^') //if element is an
operator

 {
 evaluate(post[i]); //pass it to the evaluate
 }
 } //print the result from the top
 printf("\n\nResult :: %d",stack[top]);
 getch();
}
void pushstack(int tmp) //definiton for push
{
top++; //incrementing top
 stack[top]=(int)(post[tmp]-48); //type casting
the string to its integer
value
}
void evaluate(char c) //evaluate function
{
 int a,b,ans; //variables used
 a=stack[top]; //a takes the value stored in the
top
 stack[top]='\0'; //make the stack top NULL as
its a string
 top--; //decrement top's value
 b=stack[top]; //put the value at new top to b
 stack[top]='\0'; //make it NULL
 top--; //decrement top
 switch(c) //check operator been passed to
evaluate
 {
 case '+': //addition
 ans=b+a;
 break;
 case '-': //subtraction
 ans=b-a;
 break;
 ans=b^a;
case '*': //multiplication
 ans=b*a;
 break;
 case '/': //division
 ans=b/a;
break;
 case '^': //power
break;
 default:
ans=0; //else 0
 }
 top++; //increment top
 stack[top]=ans; //store the answer at top
}
```

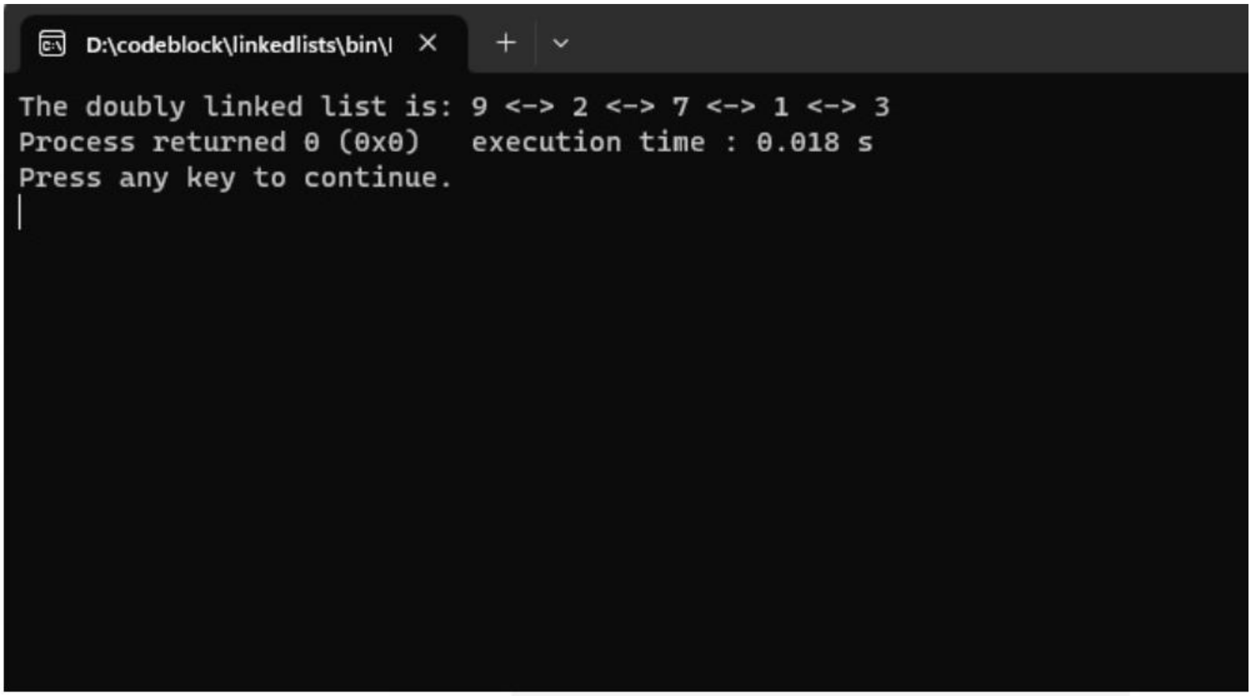


```
The linked list is: 10 -> 5 -> 9 -> 8 -> 7 -> null
Process returned 0 (0x0)   execution time : 0.031 s
Press any key to continue.
```

Practical No : 14

Aim : Implement a singly linked list

```
#include <iostream>
using namespace std;

struct Node
{
    int data;
    struct Node *next;
};

struct Node *head = NULL;

void insert(int new_data)
{
    struct Node *new_node = (struct Node *)malloc(sizeof(struct Node));
    new_node->data = new_data;
    new_node->next = head;
    head = new_node;
}
void display()
{
    struct Node *ptr;
    ptr = head;
    while (ptr != NULL)
    {
        cout << ptr->data << " -> ";
        ptr = ptr->next;
    }
    cout << "null ";
}

int main()
{
    insert(7);
    insert(8);
    insert(9);
    insert(5);
    insert(10);
    cout << "The linked list is: ";
    display();
    return 0;
}
```

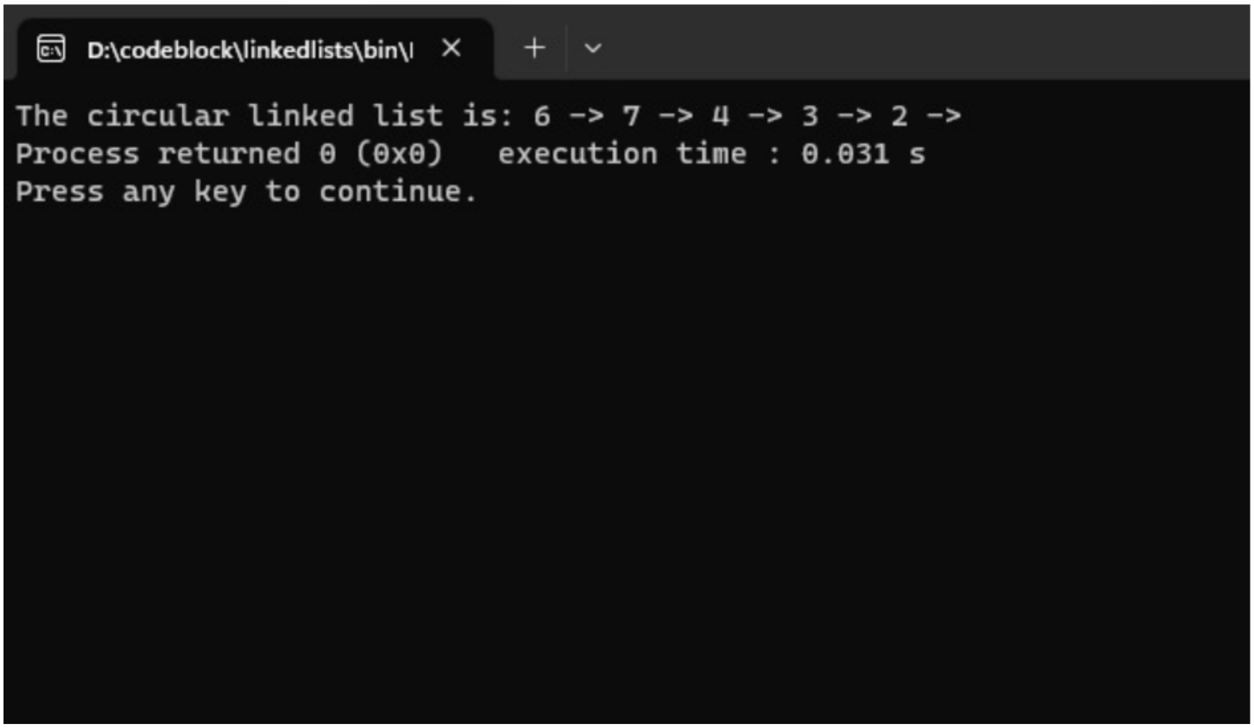


```
The doubly linked list is: 9 <-> 2 <-> 7 <-> 1 <-> 3
Process returned 0 (0x0)   execution time : 0.018 s
Press any key to continue.
```

Practical No : 15

Aim : Implement a doubly linked list

```
#include <iostream>
using namespace std;

struct Node
{
    int data;
    struct Node *prev;
    struct Node *next;
};
struct Node *head = NULL;

void insert(int newdata)
{
    struct Node *newnode = (struct Node *)malloc(sizeof(struct Node));
    newnode->data = newdata;
    newnode->prev = NULL;
    newnode->next = head;
    if (head != NULL)
        head->prev = newnode;
    head = newnode;
}

void display()
{
    struct Node *ptr;
    ptr = head;
    while (ptr != NULL)
    {
        if (ptr->next != NULL)
        {
            cout << ptr->data << " <-> ";
        }
        else
        {
            cout << ptr->data;
        }
        ptr = ptr->next;
    }
}

int main()
{
    insert(3);
    insert(1);
    insert(7);
    insert(2);
    insert(9);
    cout << "The doubly linked list is: ";
    display();
    return 0;
}
```



```
The circular linked list is: 6 -> 7 -> 4 -> 3 -> 2 ->
Process returned 0 (0x0)   execution time : 0.031 s
Press any key to continue.
```

Practical No : 16

Aim : Implement a circular linked list

```
#include <iostream>
using namespace std;

struct Node
{
   int data;
   struct Node *next;
};

struct Node *head = NULL;

void insert(int newdata)
{
   struct Node *newnode = (struct Node *)malloc(sizeof(struct Node));
   struct Node *ptr = head;
   newnode->data = newdata;
   newnode->next = head;
   if (head != NULL)
   {
      while (ptr->next != head)
         ptr = ptr->next;
      ptr->next = newnode;
   }
   else
      newnode->next = newnode;
   head = newnode;
}
```

```
void display()
{
   struct Node *ptr;
   ptr = head;
   do
   {
      cout << ptr->data << " -> ";
      ptr = ptr->next;
   } while (ptr != head);
}

int main()
{
   insert(2);
   insert(3);
   insert(4);
   insert(7);
   insert(6);
   cout << "The circular linked list is: ";
   display();
   return 0;
}
```