7. Write a C program to implement array operations such as Insert, Delete and Display.

#include <stdio.h>

int main()

{

int i,n,pos,ele,c,position,arr[100];

printf("Enter array size:");

scanf("%d",&n);

printf("Enter elements:");

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

{

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

}

printf("Enter the position to be inserted:");

scanf("%d",&pos);

printf("Enter the element to be inserted:");

scanf("%d",&ele);

if(pos > n)

printf("Invalid Input");

else

for(i=n-1;i>=pos-1;i--)

arr[i+1] = arr[i];

arr[pos-1] = ele;

printf("Array after insertion is:\n");

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

printf("%d\n", arr[i]);

printf("Enter the position to be deleted:");

scanf("%d",&position);

if (position > n+1)

printf("\nDeletion not possible.\n");

else

for(c=position-1;c<n-1;c++)

arr[c] = arr[c+1];

printf("\nArray after deletion :\n");

for(c=0;c<n-1;c++)

printf("%d\n", arr[c]);

printf("\nEnter element :");

scanf("%d", &ele);

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

{

if(arr[c] == ele)

{

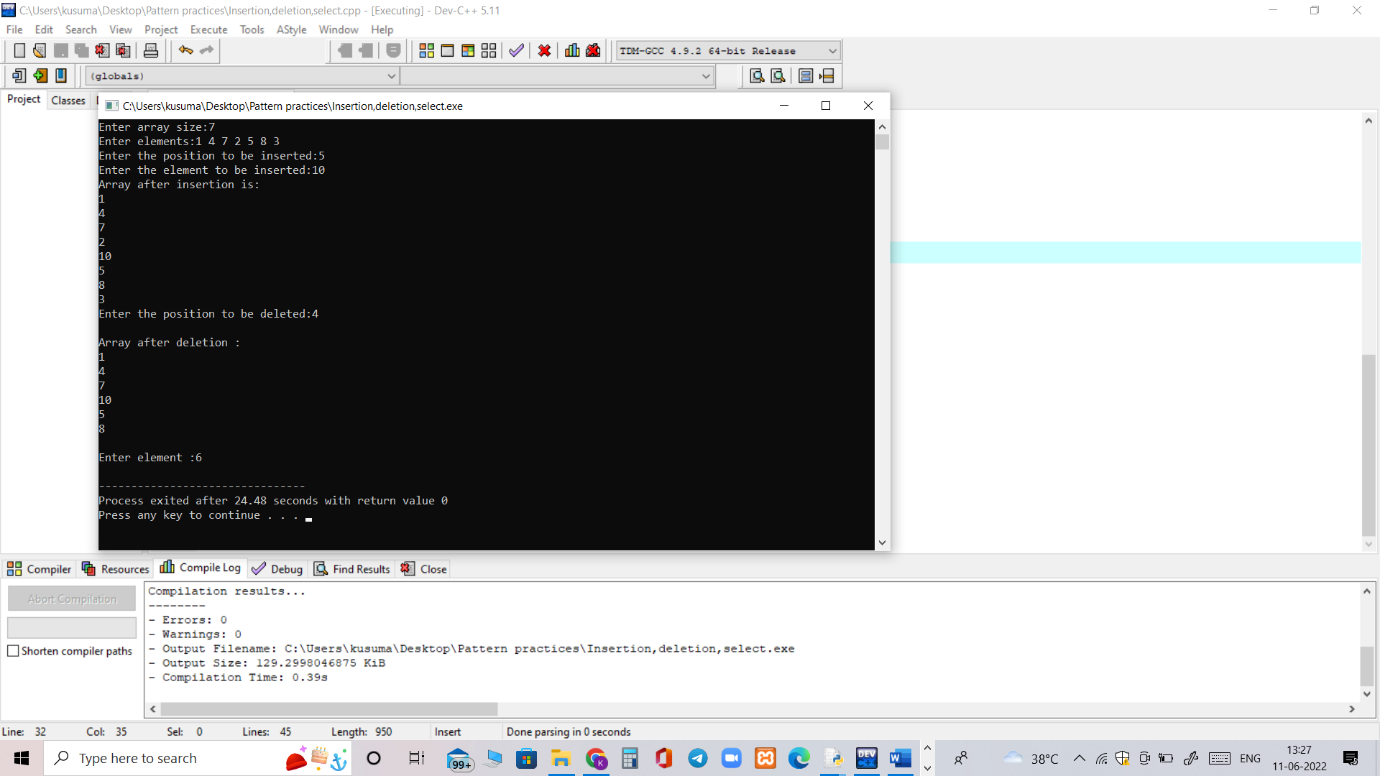
printf("\nElement found\n");

}

}

return 0;

}



8. Write a C program to search a number using Linear Search method

#include <stdio.h>

int main()

{

int num;

int i, keynum, found = 0;

printf("Enter the number of elements ");

scanf("%d", &num);

int array[num];

printf("Enter the elements one by one \n");

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

{

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

}

printf("Enter the element to be searched ");

scanf("%d", &keynum);

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

{

if (keynum == array[i] )

{

found = 1;

break;

}

}

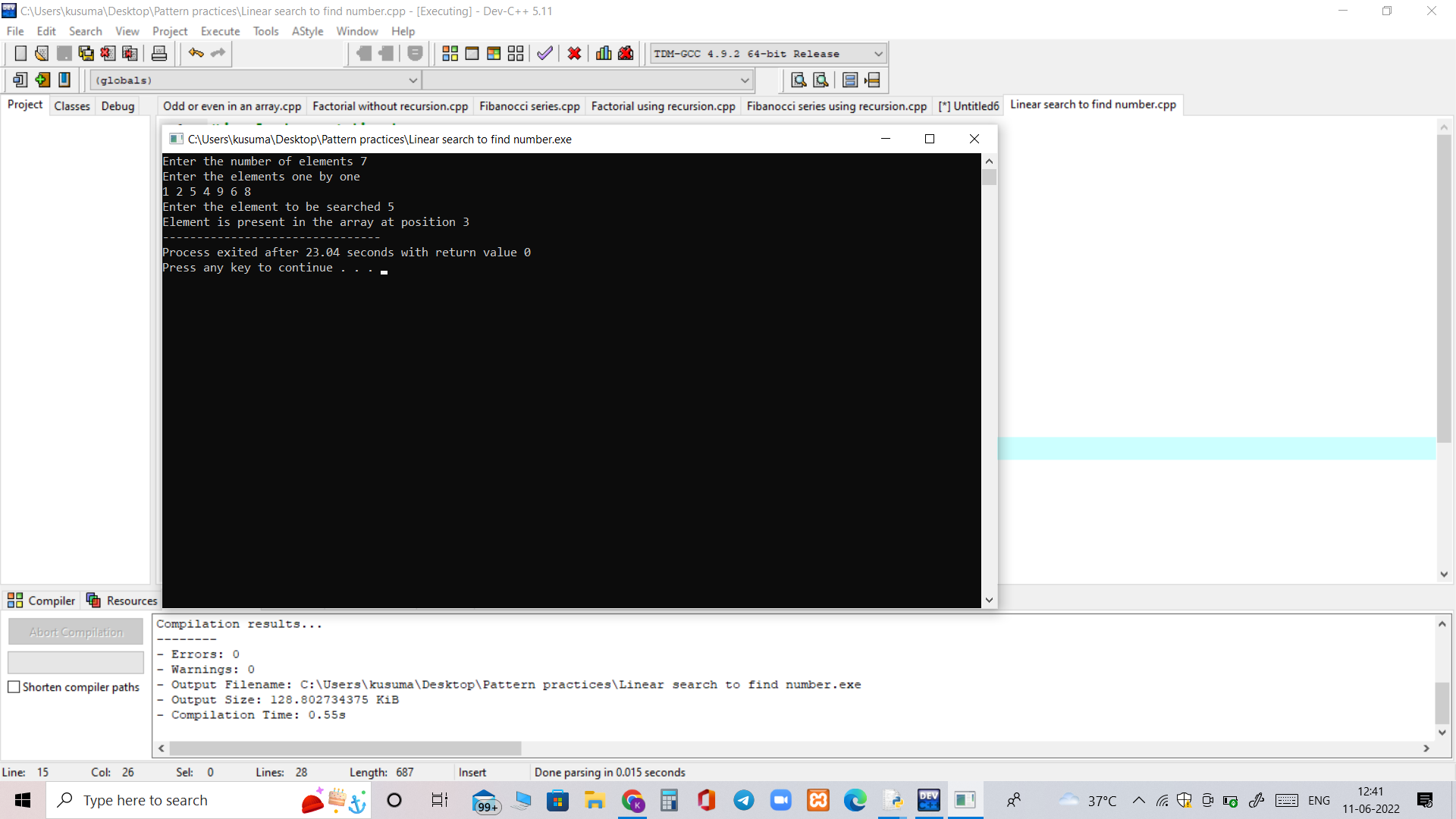
if (found == 1)

printf("Element is present in the array at position %d",i+1);

else

printf("Element is not present in the array\n");

}



9. Write a C program to search a number using Binary Search method

#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

last = middle - 1;

middle = (first + last)/2;

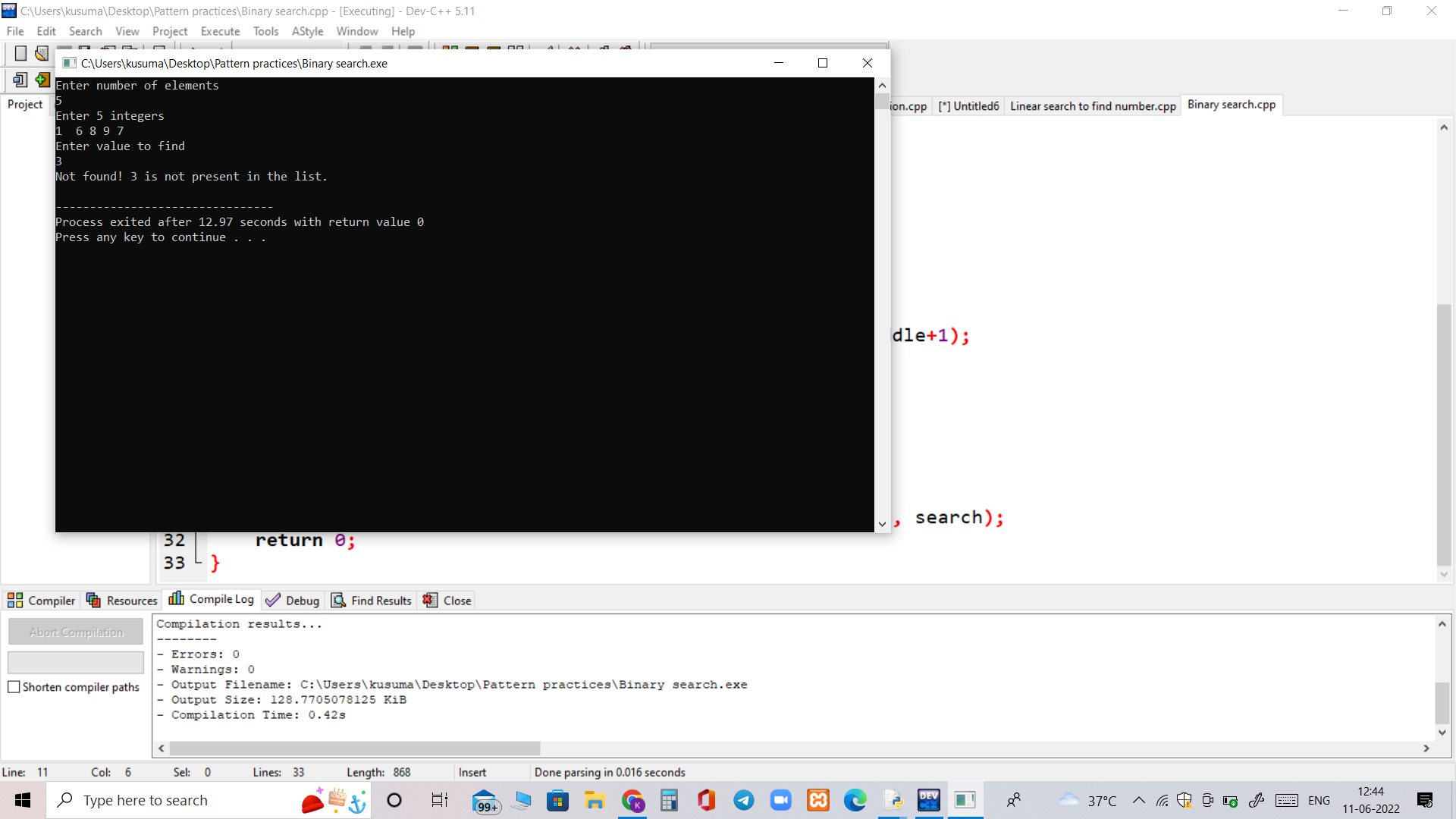
}

if ( first > last )

printf("Not found! %d is not present in the list.\n", search);

return 0;

}



10. Push , peek , pop and exit program

#include<stdio.h>

#include<stdlib.h>

#define Size 4

int Top=-1, inp\_array[Size];

void Push();

void Pop();

void show();

int main()

{

int choice;

while(1)

{

printf("\nOperations performed by Stack");

printf("\n1.Push the element\n2.Pop the element\n3.Show\n4.End");

printf("\n\nEnter the choice:");

scanf("%d",&choice);

switch(choice)

{

case 1: Push();

break;

case 2: Pop();

break;

case 3: show();

break;

case 4: exit(0);

default: printf("\nInvalid choice!!");

}

}

}

void Push()

{

int x;

if(Top==Size-1)

{

printf("\nOverflow!!");

}

else

{

printf("\nEnter element to be inserted to the stack:");

scanf("%d",&x);

Top=Top+1;

inp\_array[Top]=x;

}

}

void Pop()

{

if(Top==-1)

{

printf("\nUnderflow!!");

}

else

{

printf("\nPopped element: %d",inp\_array[Top]);

Top=Top-1;

}

}

void show()

{

if(Top==-1)

{

printf("\nUnderflow!!");

}

else

{

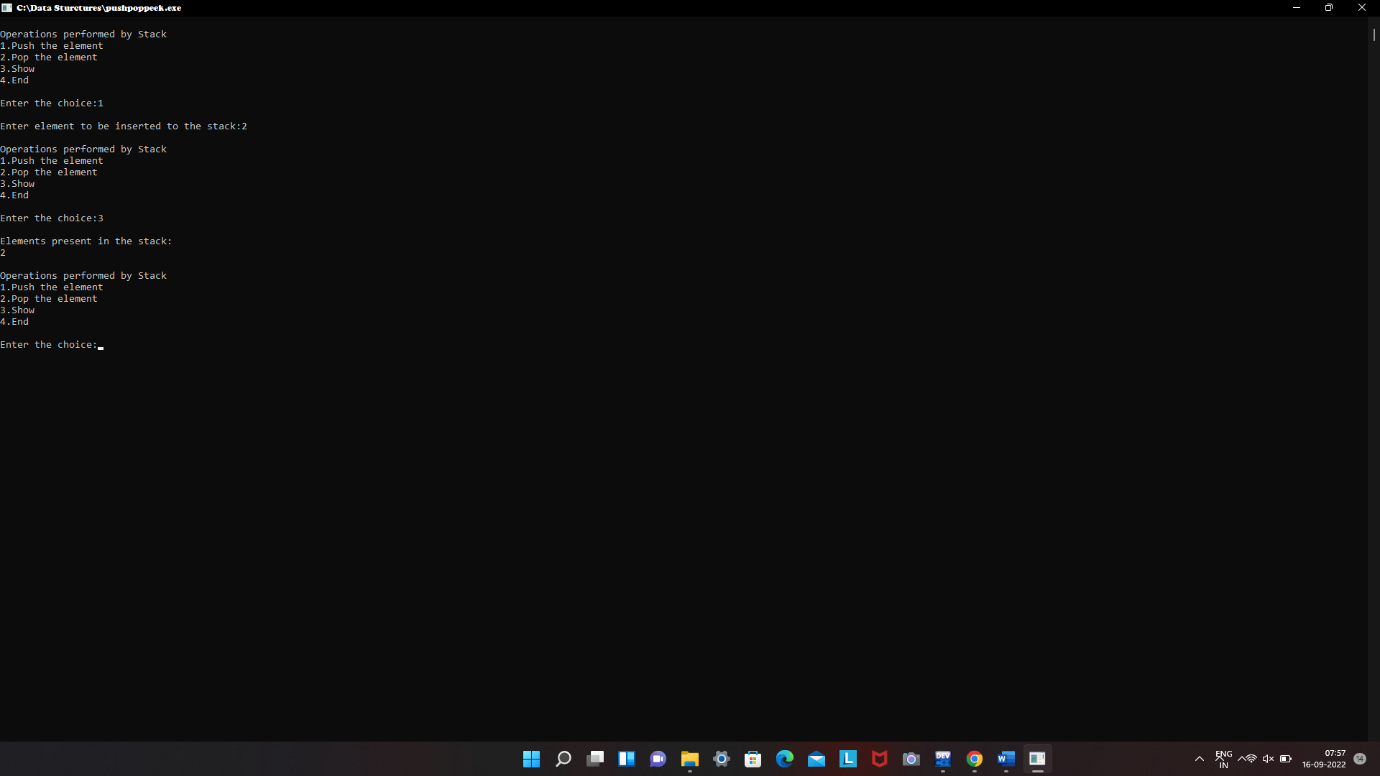
printf("\nElements present in the stack: \n");

for(int i=Top;i>=0;--i)

printf("%d\n",inp\_array[i]);

}

}



11. Enqueue and Dequeue Operations

#include<stdio.h>

#include<stdlib.h>

void insert();

void dequeue();

void display();

int front = -1, rear = -1 ,maxsize;

int queue[100];

int main ()

{

int choice;

printf("\n Enter the size of QUEUE : ");

scanf("%d",&maxsize);

printf("\n QUEUE OPERATIONS USING ARRAY");

printf("\n1.insert an element\n2.Delete an element\n3.Display the queue\n4.Exit");

while(choice != 4)

{

printf("\nEnter your choice : ");

scanf("%d",&choice);

switch(choice)

{

case 1:

insert();

break;

case 2:

dequeue();

break;

case 3:

display();

break;

case 4:

exit(0);

break;

default:

printf("\nEnter valid choice??\n");

}

}

return 0;

}

void insert()

{

int item;

printf("\nEnter the element\n");

scanf("\n%d",&item);

if(rear == maxsize-1)

{

printf("\nOVERFLOW\n");

return;

}

if(front == -1 && rear == -1)

{

front = 0;

rear = 0;

}

else

{

rear = rear+1;

}

queue[rear] = item;

printf("\nValue inserted ");

}

void dequeue()

{

int item;

if (front == -1 || front > rear)

{

printf("\nUNDERFLOW\n");

return;

}

else

{

item = queue[front];

if(front == rear)

{

front = -1;

rear = -1 ;

}

else

{

front = front + 1;

}

printf("\nvalue deleted ");

}

}

void display()

{

int i;

if(rear == -1)

{

printf("\nEmpty queue\n");

}

else

{ printf("\n Elements in the queue are\n");

for(i=front;i<=rear;i++)

{

printf("\n%d",queue[i]);

}

}

}

