# **Name: Abdurrahman Qureshi**

# **Roll No: 242466**

Practical No: 2

1) To implement stack and Queue using array.

CODE (STACK)

#include<stdio.h>

#include<stdlib.h>

#define MAX 4

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

void Push();

void Pop();

void show();

void peek();

int main()

{

int choice;

while(1)

{

printf("\nOperations performed by Stack");

printf("\n1.Push the element\n2.Pop the element\n3.Show\n4. Peek \n5.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: peek();

break;

case 5: exit(0);

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

}

}

}

void Push()

{

int x;

if(Top==MAX-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(i=Top;i>=0;i--)

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

}

}

void peek()

{

if(Top==-1)

{

printf("Stack is empty");

}

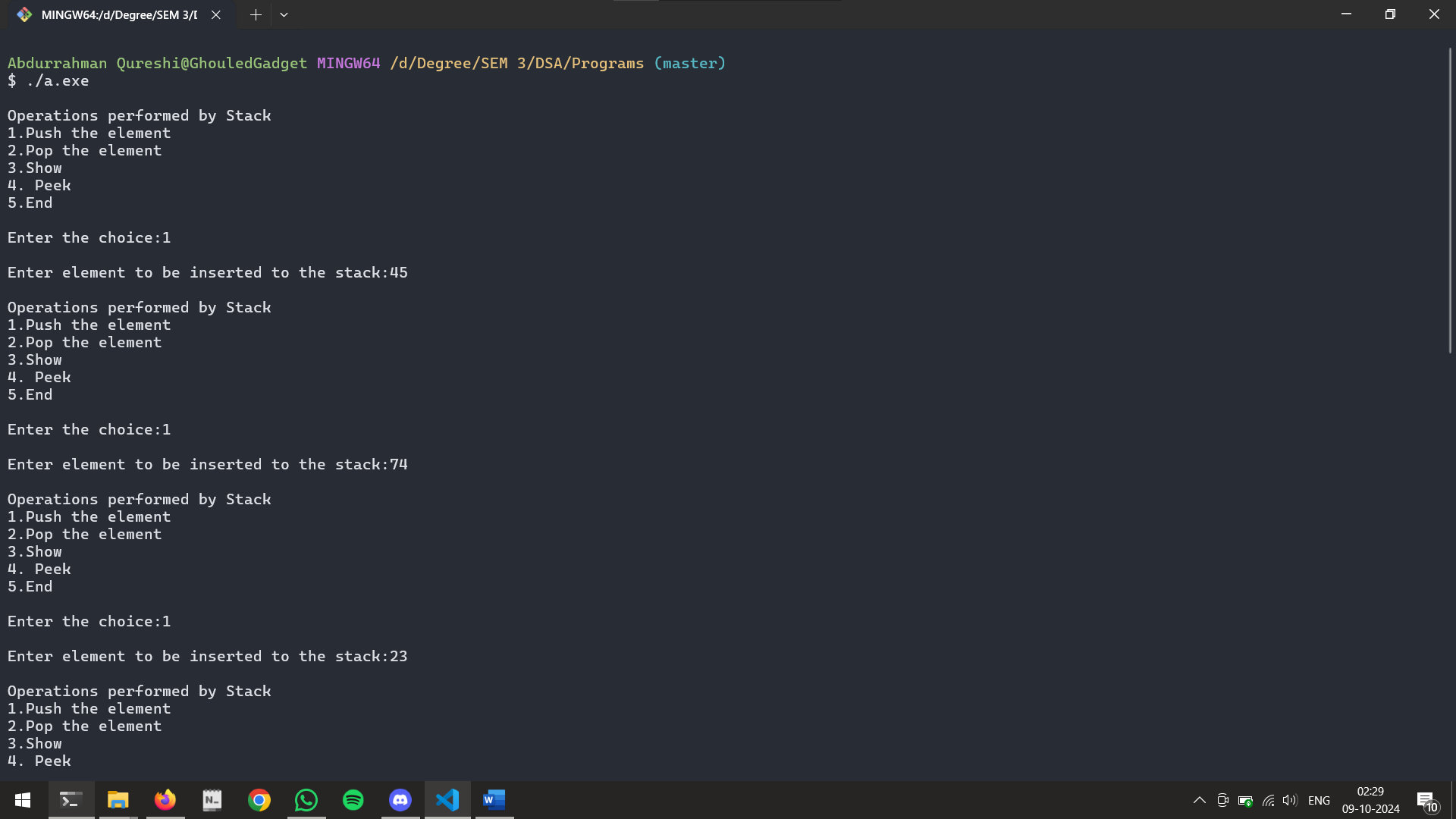
else {

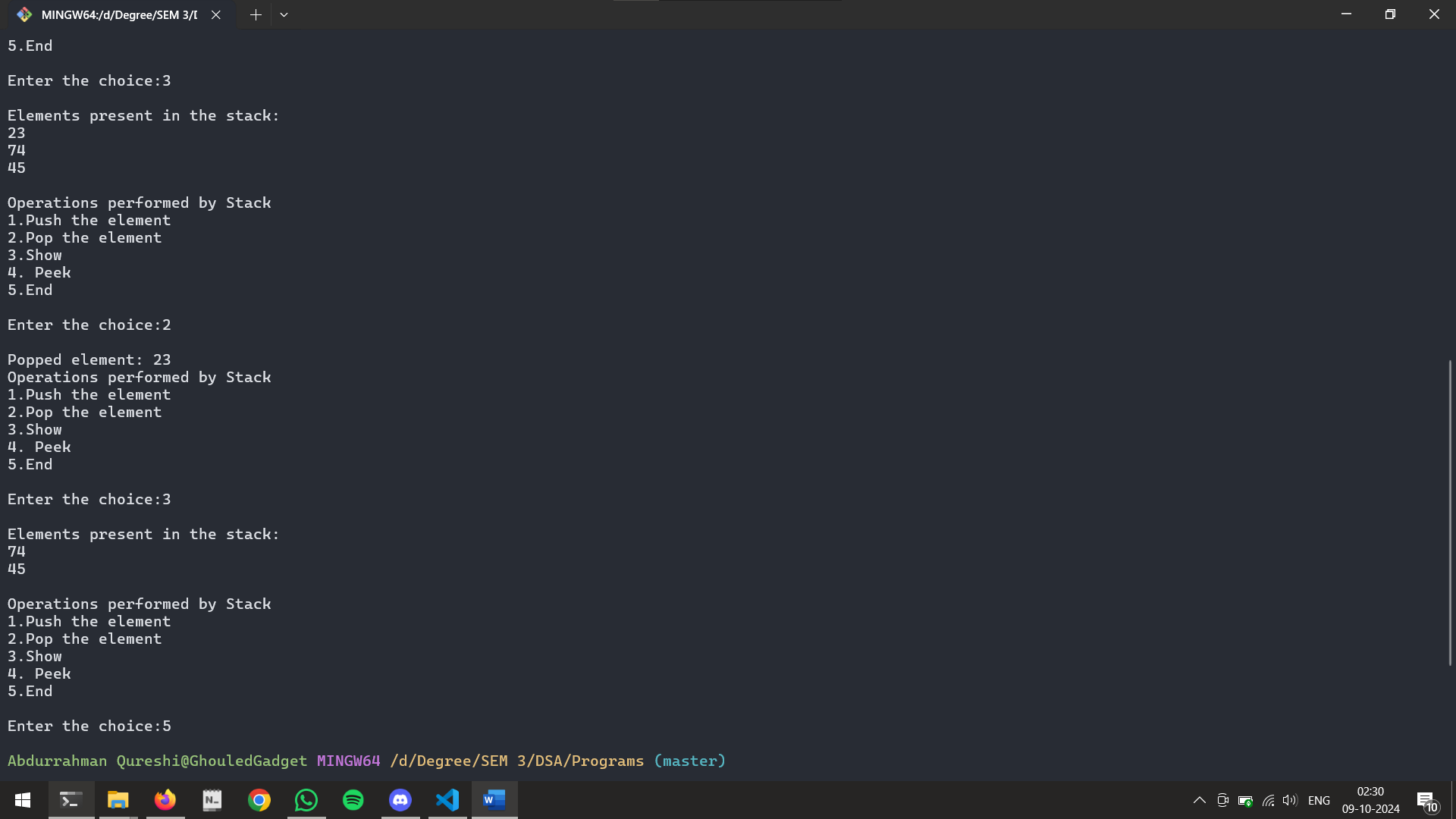
printf("%d",inp\_array[Top]);

}

}

OUTPUT (STACK)





Source code (QUEUE)

#include <stdio.h>

#include <stdlib.h>

#define MAX 10

int q[MAX];

int f=-1,r=-1,i,x;

void insert();

void delet();

void peek();

void display();

int main()

{

int c;

while(1)

{

printf("\n Press 1 for insertion");

printf("\n Press 2 for deletion");

printf("\n Press 3 for peek");

printf("\n Press 4 to display\n");

scanf("%d",&c);

switch(c)

{

case 1: insert();

break;

case 2: delet();

break;

case 3: peek();

break;

case 4: display();

break;

default: printf("\n invalid choice");

}

}

}

void insert()

{

printf("\n Enter value to insert");

scanf("%d",&x);

if (r==MAX-1)

{

printf("OVERFLOW");

}

else

{

if(f==-1)

f=0;

r=r+1;

q[r]=x;

printf("Inserted element is %d\n",q[r]);

}

}

void delet()

{

if (f==-1 || f>r)

{

printf("UNDERFLOW");

}

else

{

x = q[f];

printf("deleted value is %d\n",q[f]);

f=f+1;

}

}

void peek()

{

if (f==-1 || f>r)

{

printf("UNDERFLOW");

}

else

{

printf("rear value is : %d",q[r]);

printf("front value is : %d",q[f]);

}

}

void display()

{

if (f==-1 || f>r)

{

printf("UNDERFLOW");

}

else {

printf("Queue elements are \n");

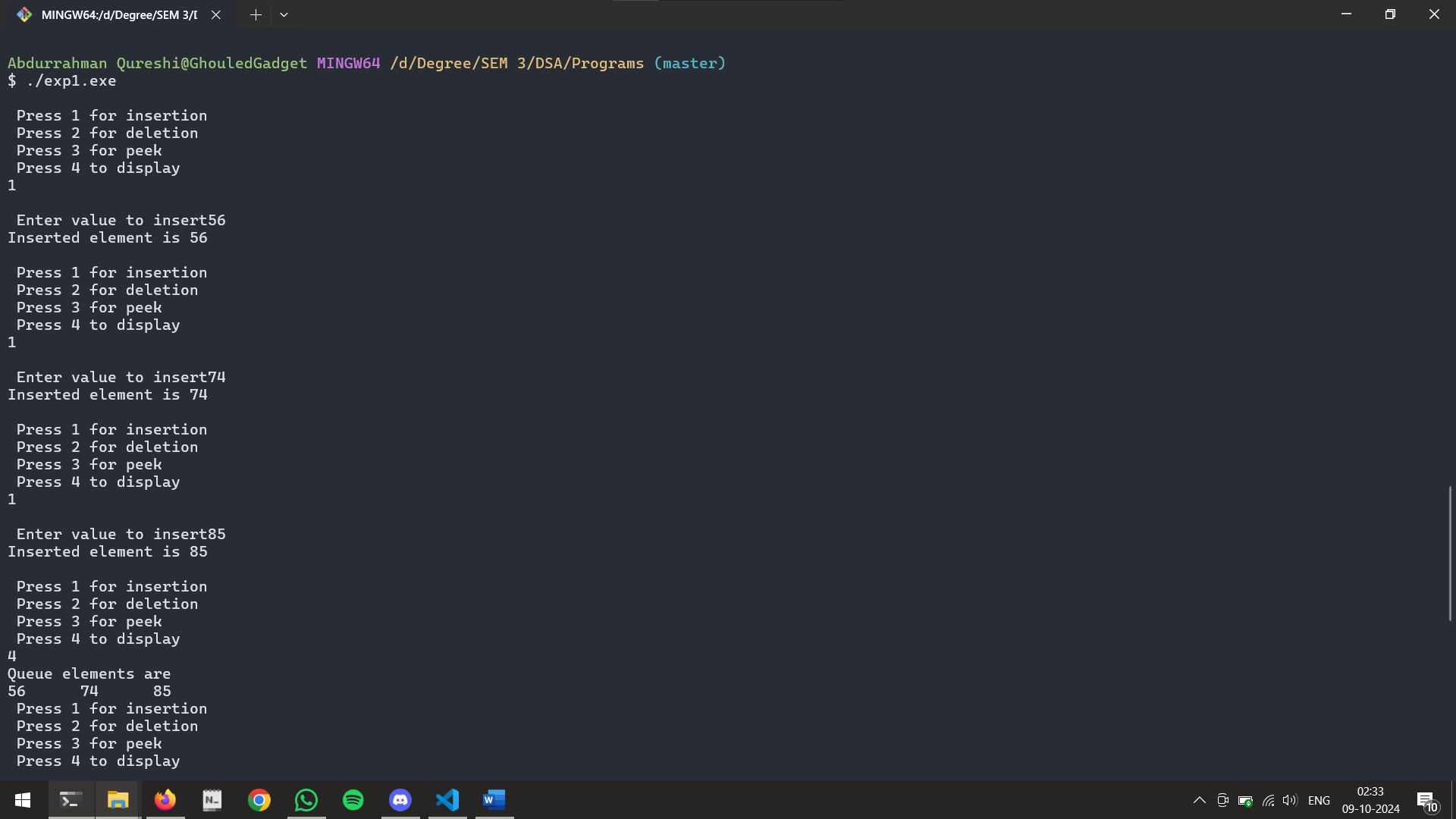
for(i=f;i<=r;i++)

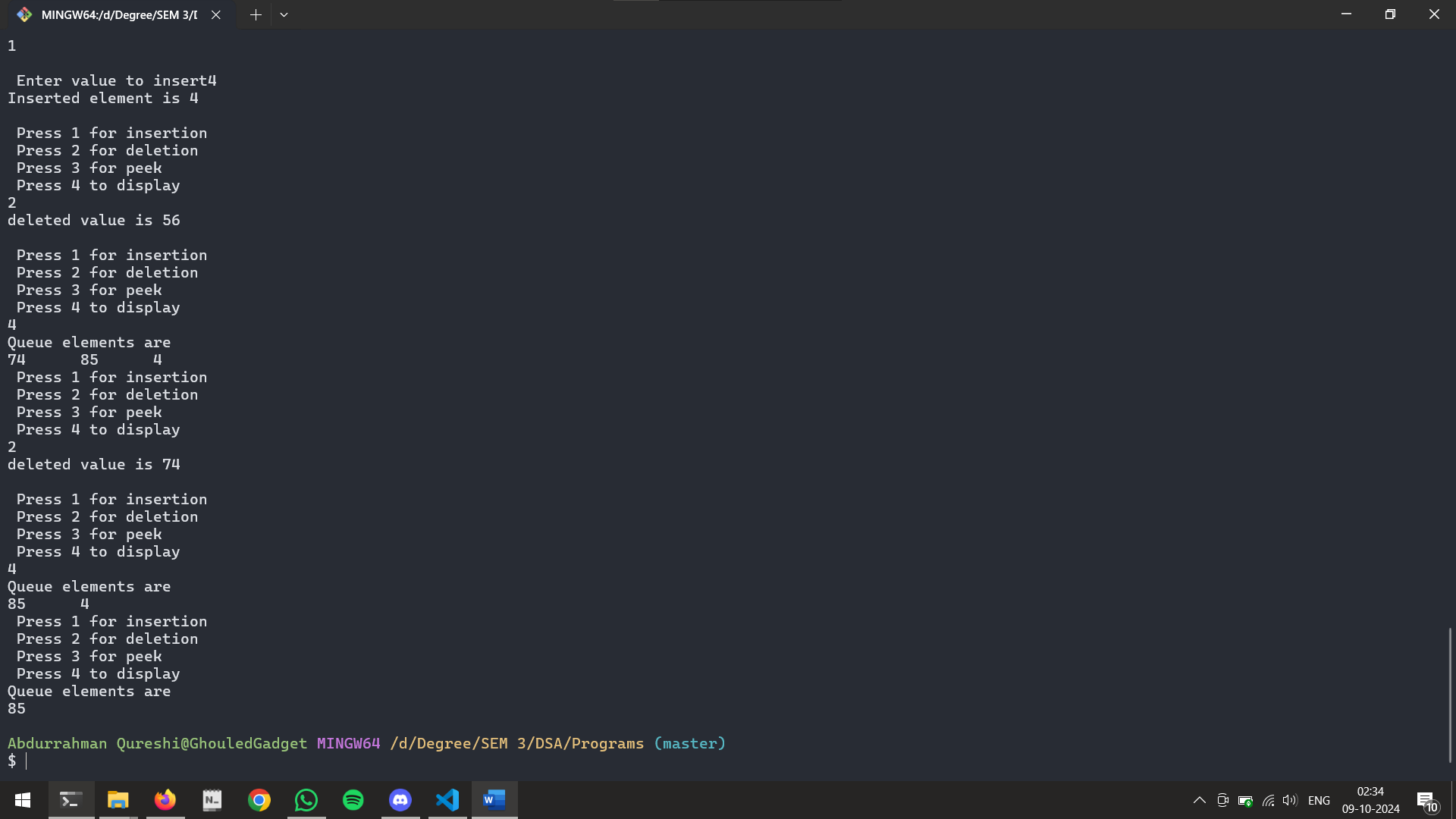
printf("%d\t",q[i]);

}

}

OUTPUT (QUEUE)





Tools used :

Software: Dev c++

Hardware: Lab Computers

References: Mam notes.

Conclusion:

(A) STACK

Stack implements the LIFO mechanism i.e. the element that is pushed at the end is popped out first.

Push: This operation adds an item to the stack; it will throw an exception if the stack’s capacity is full.

Pop: This removes the last inserted element from a stack that is removed in reverse order of their insertion; it will throw an underflow exception if the given stack is empty.

Peek: This operation returns the topmost element of the stack.

(B) QUEUE

It is a type of linear data structure which follow first in first out(FIFO) approach. This means the element that is inserted first will be removed first. The queue

is said to have 2 ends, one from where the element is inserted is known as the REAR end, and the end where the element is deleted is known as FRONT. This means in the queue, one end (REAR) is always used to enqueue, and the other end (FRONT) is used to dequeue.