**ALGORITHM TO PERFORM OPERATIONS OF QUEUE (PUSH, POP, DISPLAY):**

STEP 1: Start.

STEP 2: Ask the user to enter the choice.

1. push
2. pop
3. display
4. quit

STEP 3: If user selected choice 1 (push),

STEP 4: Ask the user to enter the value to be inserted, provided it is integer

value only.

STEP 5: Checks if the queue is in overflow condition or not,

If yes, print Queue is overflow(rear=MAX-1).

STEP 6: If the queue is not full, then the element is inserted(front===-1).

STEP 7: Returns to step 2 (Step 2 is displayed again).

STEP 8: If user selected choice 2 (pop),

STEP 9: Checks if the queue is in underflow condition or not, if yes print

Queue is underflow.

STEP 10: If no, the first element from front end is popped (deleted)

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

STEP 11: Returns to step 2.

STEP 12: If user selected choice 3 (display),

STEP 13: If elements are there in queue, it displays the elements present in

Queue(front==-1).

STEP 14: If the queue has no elements, print Queue is empt.

STEP 14: Returns to step 2.

STEP 15: If user selected choice 4 (quit),

STEP 16: Program execution ends (Execution is completed).

STEP 17: Stop.

**SOURCE CODE FOR QUEUE OPERATIONS:**

#include <stdio.h>

#define MAX 50

void insert();

void delete();

void display();

int queue\_array[MAX];

int rear = - 1;int front = - 1;

main(){

int choice;

while (1) {

printf("1.Insert element to queue \n");

printf("2.Delete element from queue \n");

printf("3.Display all elements of queue \n");

printf("4.Quit \n");

printf("Enter your choice : ");

scanf("%d", &choice);

switch (choice) {

case 1: insert(); break;

case 2: delete(); break;

case 3: display(); break;

case 4: exit(1);

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

} /\* End of switch \*/

} /\* End of while \*/

} /\* End of main() \*/

void insert(){

int add\_item;

if (rear == MAX - 1)

printf("Queue Overflow \n");

else { if (front == - 1) /\*If queue is initially empty \*/

front = 0;

printf("Inset the element in queue : ");

scanf("%d", &add\_item); rear = rear + 1;

queue\_array[rear] = add\_item; }} /\* End of insert() \*/

void delete(){

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

{ printf("Queue Underflow \n");

return ; }

else {

printf("Element deleted from queue is : %d\n", queue\_array[front]);

front = front + 1;

}

} /\* End of delete() \*/

void display(){

int i;

if (front == - 1)

printf("Queue is empty \n");

else { printf("Queue is : \n");

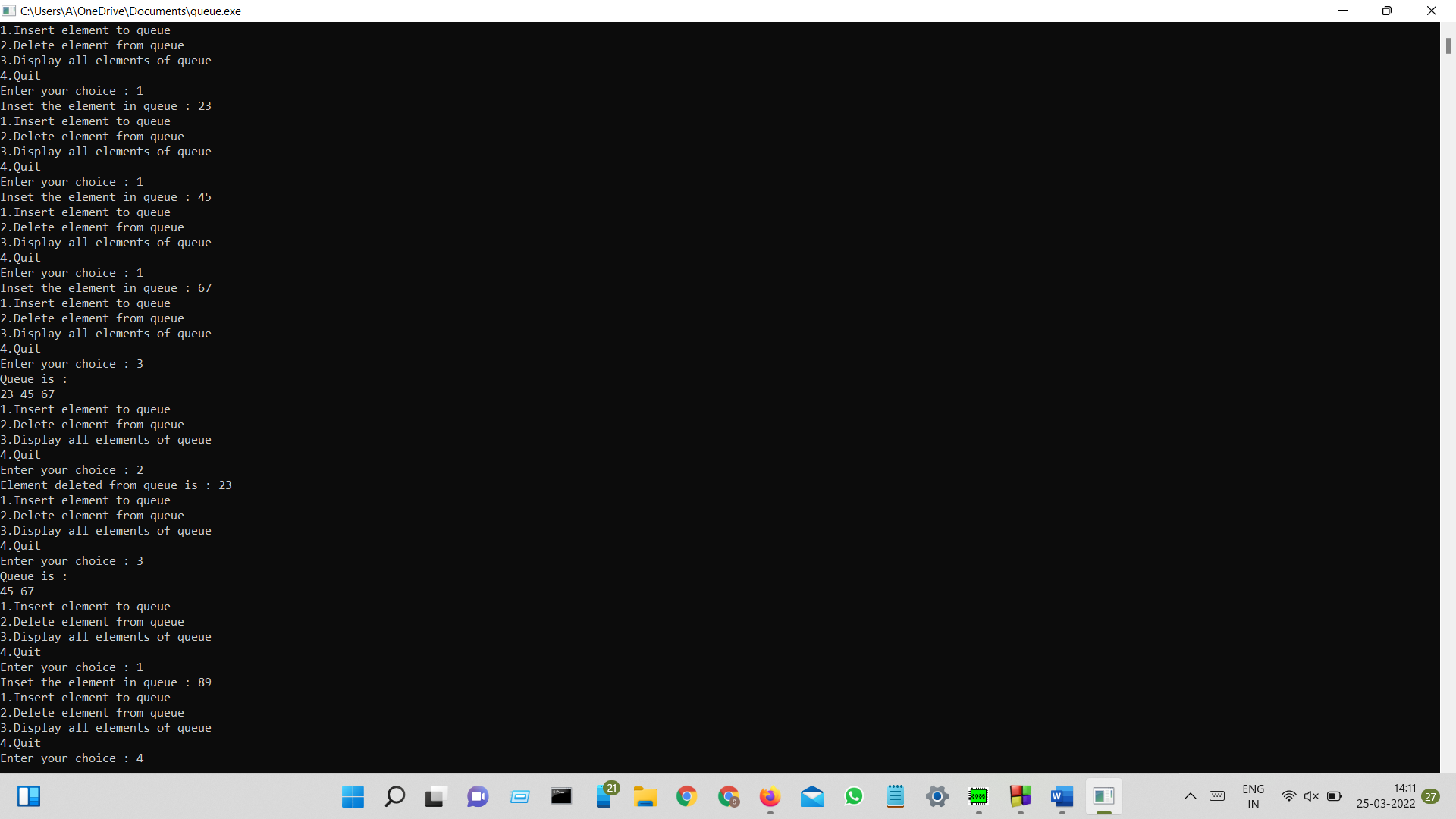
for (i = front; i <= rear; i++)

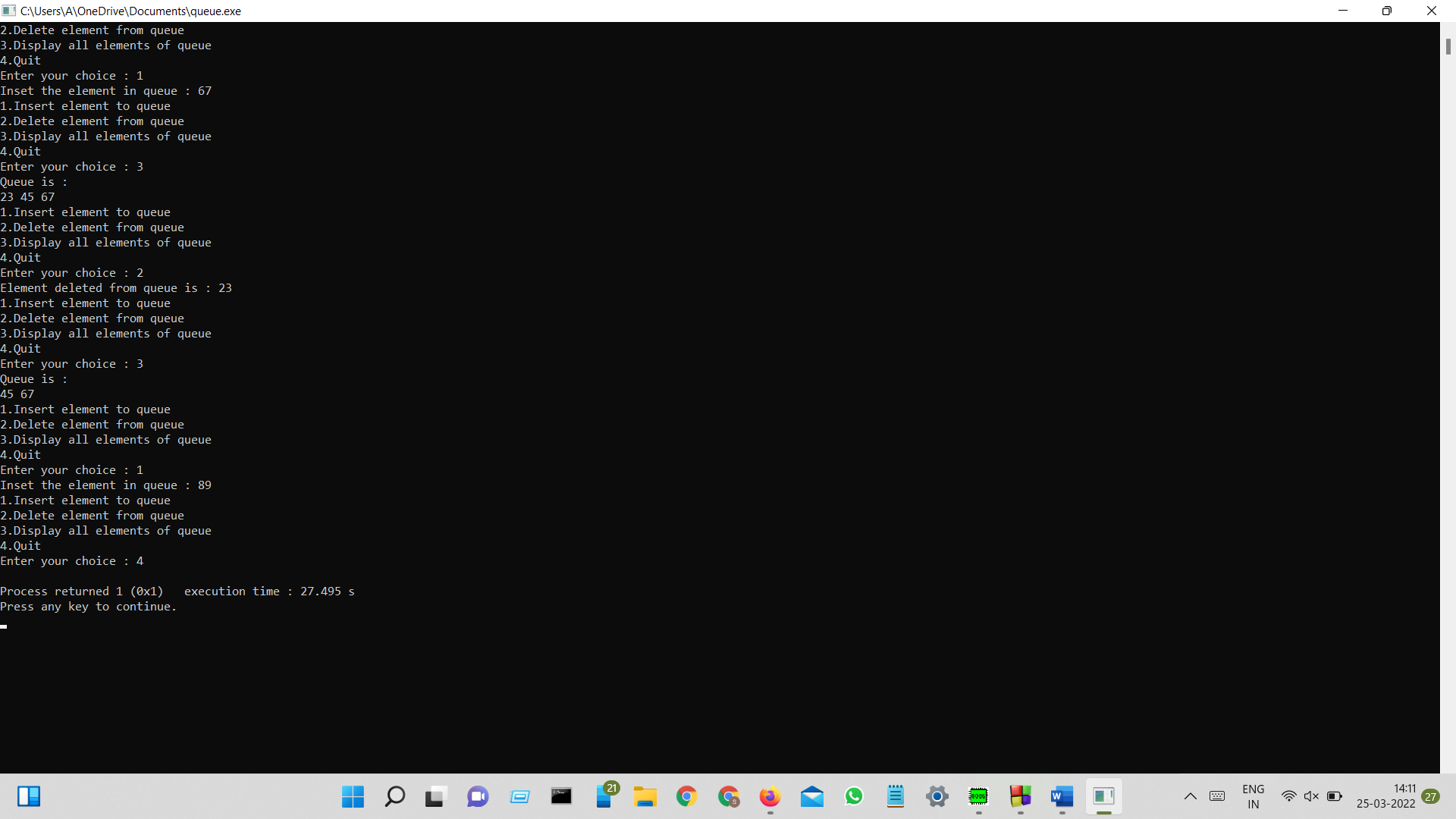
printf("%d ", queue\_array[i]);

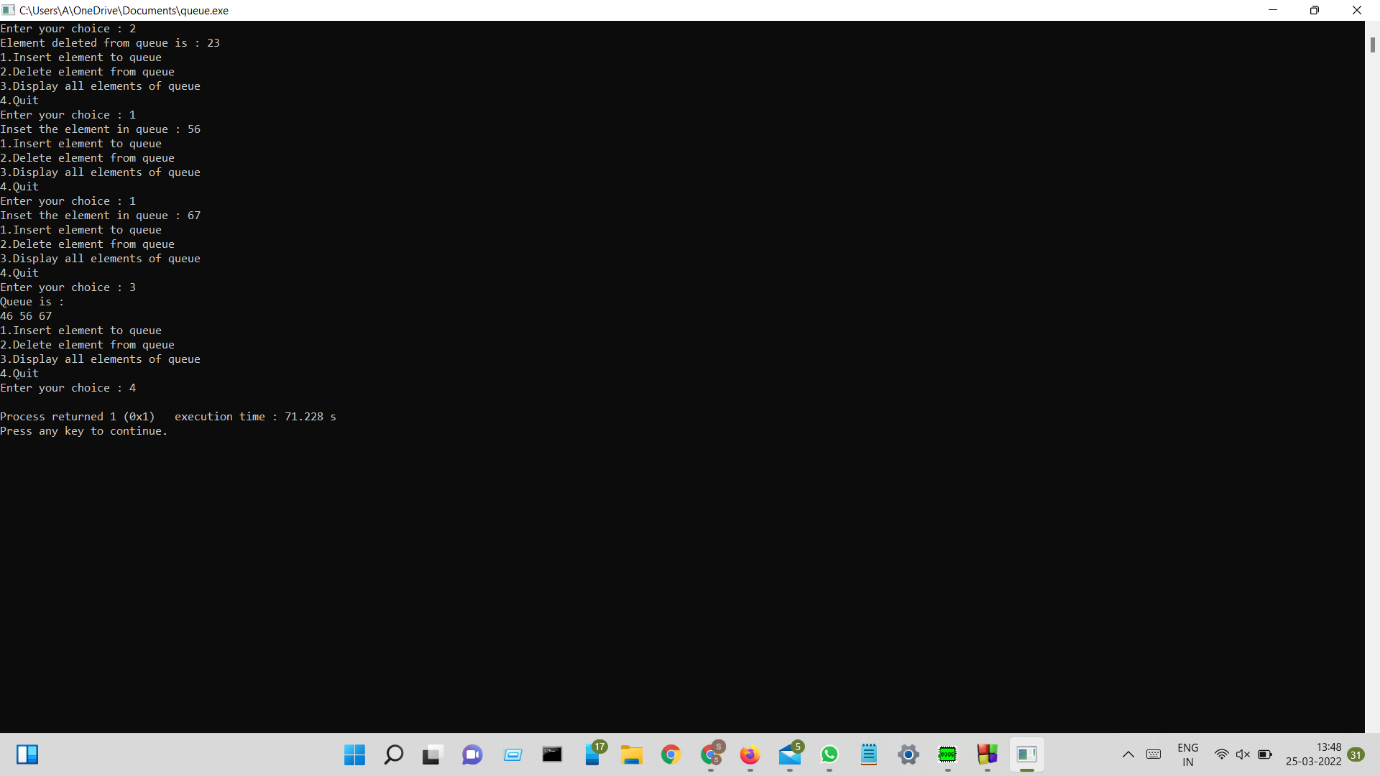
printf("\n");

}} /\* End of display() \*/

**OUTPUT:**







DONE BY,

Lakshmi Priya K,

CSE Section K,

AP21110010670.