//Write a menu driven code to implement Linear Queue ADT using arrays

#include <stdio.h>

#include <stdlib.h>

#define MAX 10 // Changing this value will change length of array

int queue[MAX];

int front = -1, rear = -1;

void Enqueue(void);

int Dequeue(void);

int GetFront(void);

int GetRear(void);

void size(void);

void display(void);

int main()

{

int option, val;

do {

printf("\n\n\*\*\*\*List of Operations\*\*\*\*");

printf("\n 1. Enqueue");

printf("\n 2. Dequeue");

printf("\n 3. Get Front");

printf("\n 4. Get Rear");

printf("\n 5. Size");

printf("\n 6. Display");

printf("\n 7. EXIT");

printf("\n Enter your option: ");

scanf("%d", &option);

switch (option) {

case 1:

Enqueue();

break;

case 2:

val = Dequeue();

if (val != -1)

printf("\n The number deleted is: %d", val);

break;

case 3:

val = GetFront();

if (val != -1)

printf("\n The first value in queue is: %d", val);

break;

case 4:

val = GetRear();

if (val != -1)

printf("\n The last value in queue is: %d", val);

break;

case 5:

size();

break;

case 6:

display();

break;

case 7:

printf("\n\tEXIT POINT");

break;

}

} while (option != 7);

return 0;

}

int isEmpty() {

return (front == -1 && rear == -1);

}

int isFull() {

return rear == MAX - 1;

}

void Enqueue()

{

int num;

printf("\n Enter the number to be inserted in the queue: ");

scanf("%d", &num);

if (isFull())

printf("\n OVERFLOW");

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

front = rear = 0;

else

rear++;

queue[rear] = num;

}

int Dequeue()

{

int val;

if (isEmpty())

{

printf("\n UNDERFLOW");

return -1;

}

else

{

val = queue[front];

if (front == rear) {

front = rear = -1;

}

else {

front++;

}

return val;

}

}

int GetFront()

{

if (isEmpty()) {

printf("\nQUEUE IS EMPTY");

return -1;

}

else {

return queue[front];

}

}

int GetRear(void)

{

if (isEmpty()) {

printf("\nQUEUE IS EMPTY");

return -1;

}

else {

return queue[rear];

}

}

void size(void)

{

int count=0;

int i;

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

{

printf("The number of elements in queue: ");

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

count++;

}

printf("%d\n",count);

}

else

{

printf("\n The Queue is empty");

}

}

void display()

{

int i;

printf("\n");

if (isEmpty())

printf("\nQUEUE IS EMPTY");

else

{

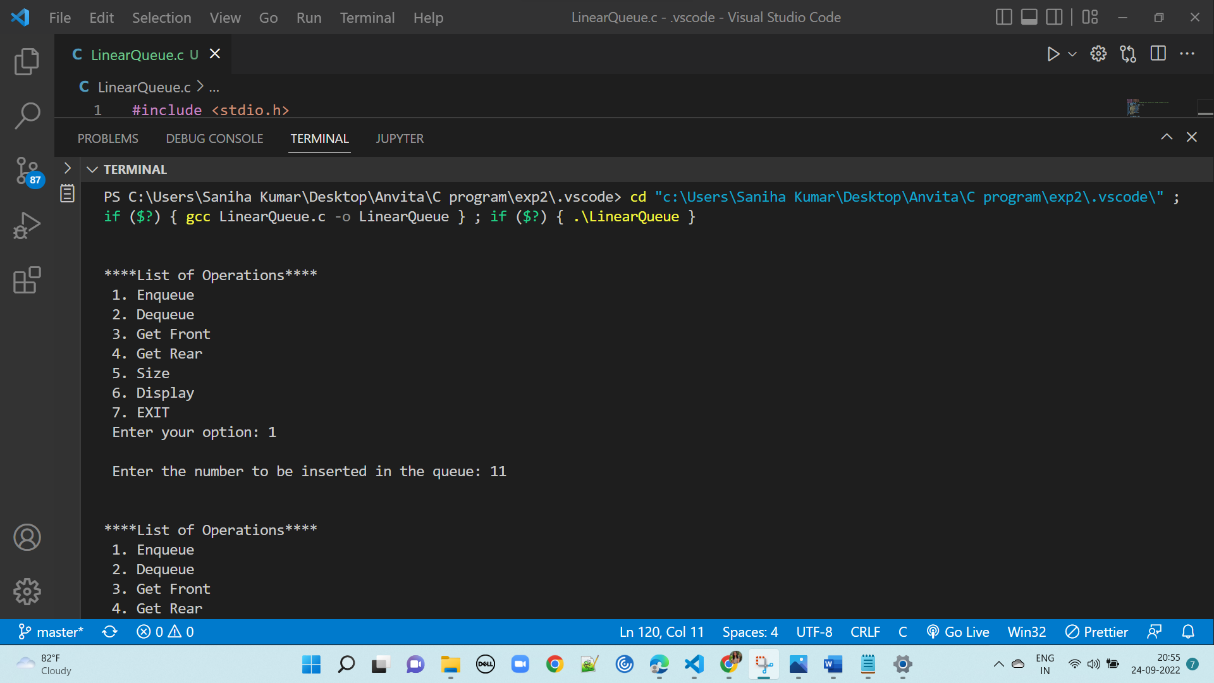
printf("\nThe Linear Queue is: ");

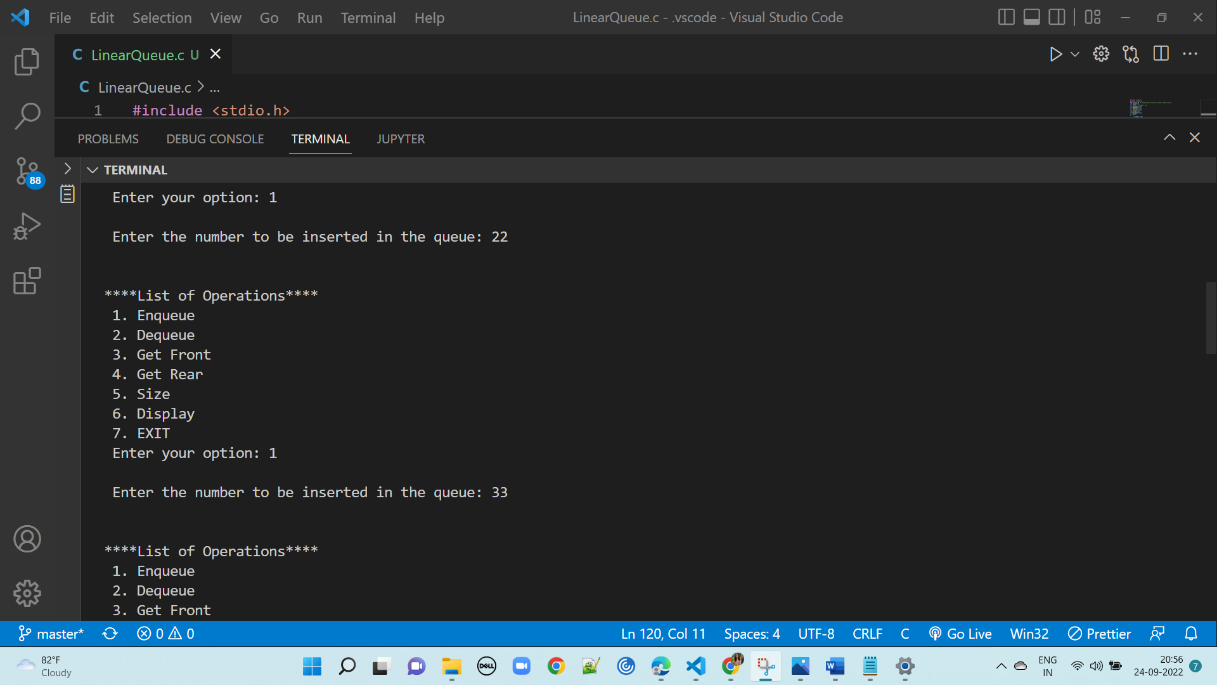
for (i=front; i<=rear; i++)

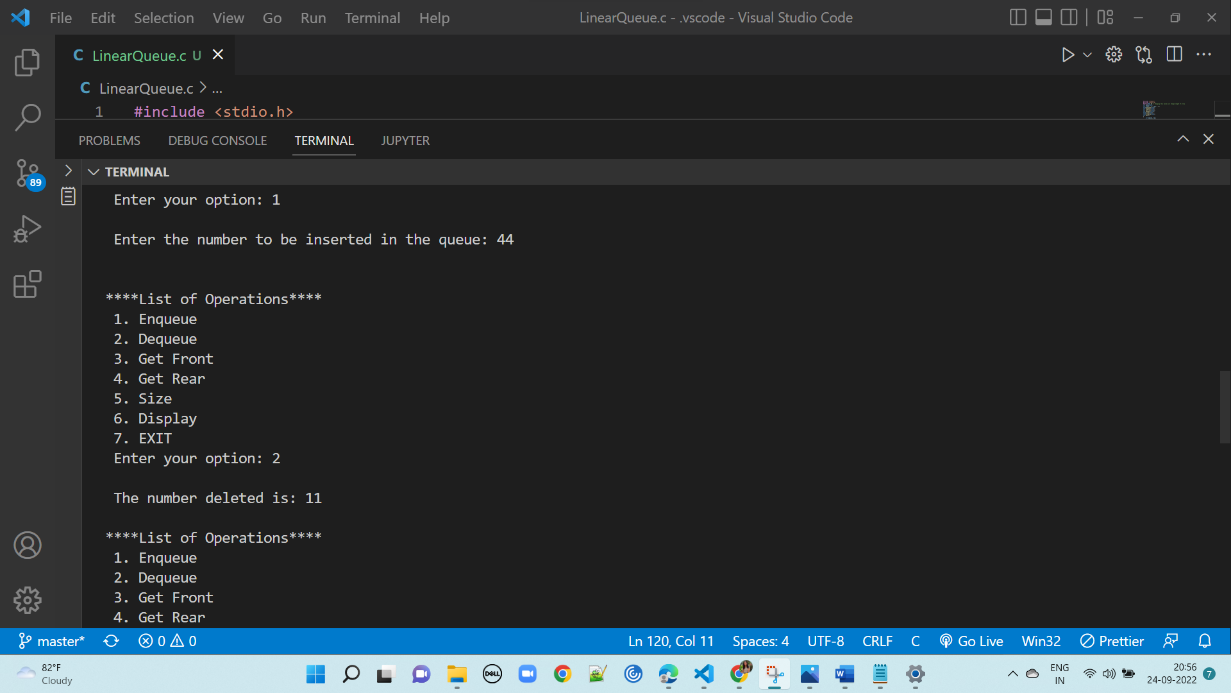
printf("\t%d", queue[i]);

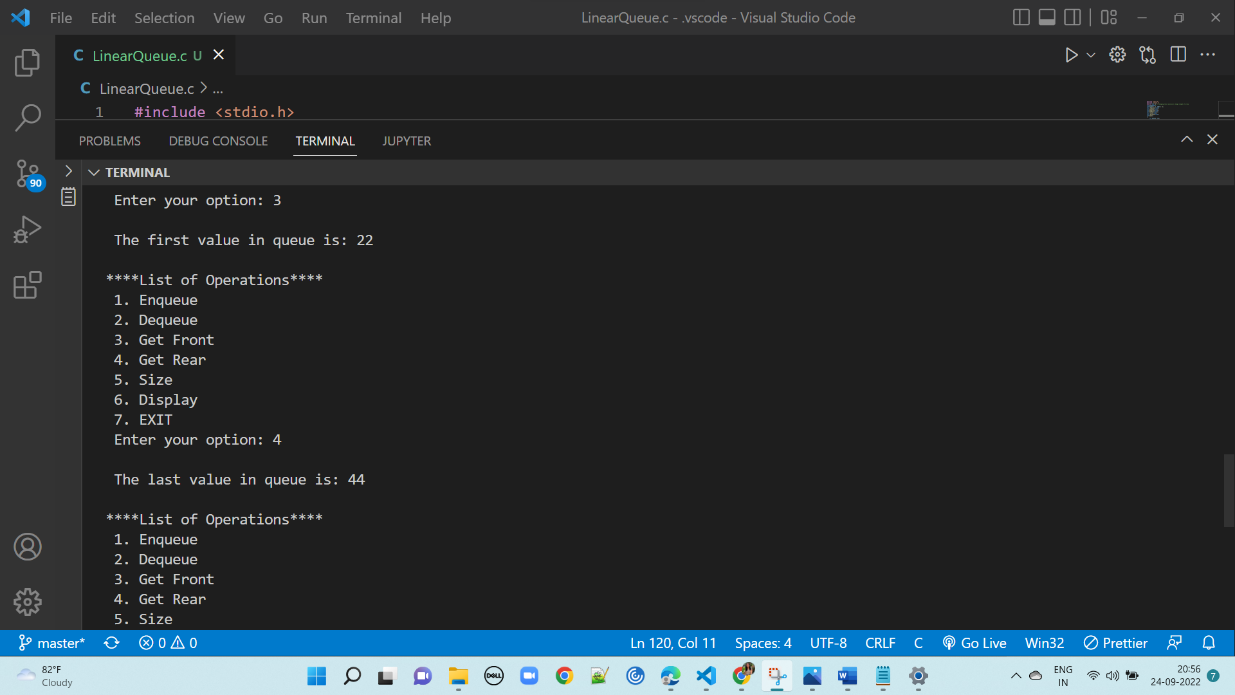
}

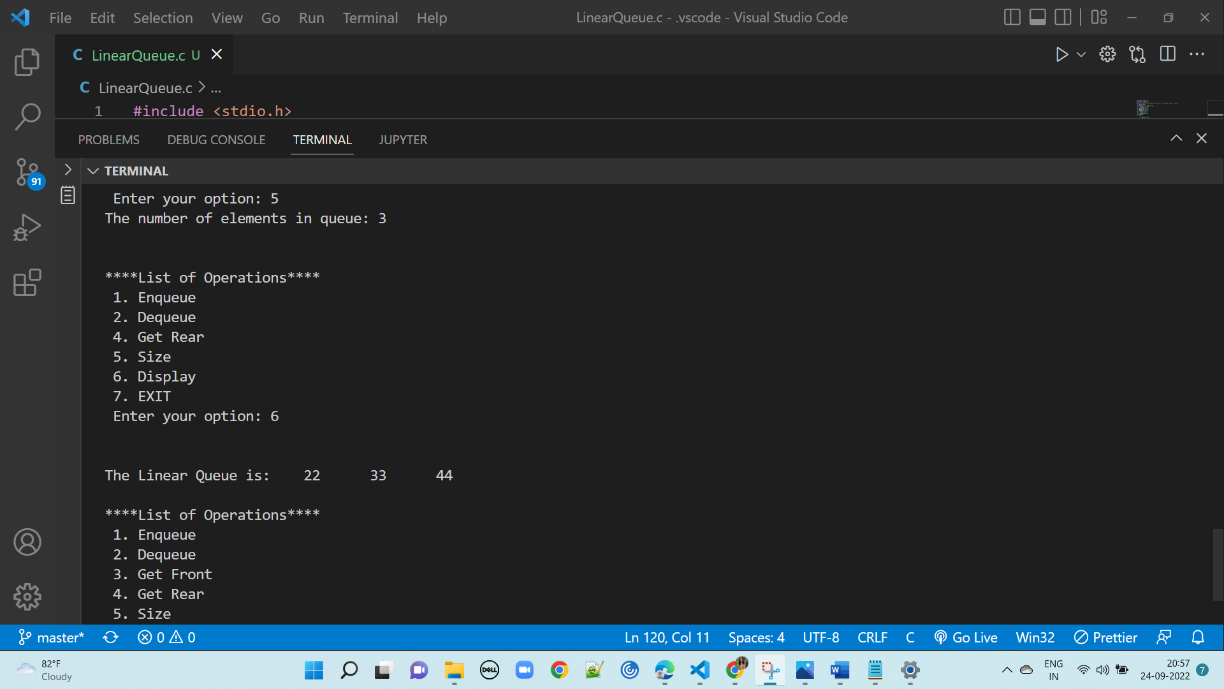
}

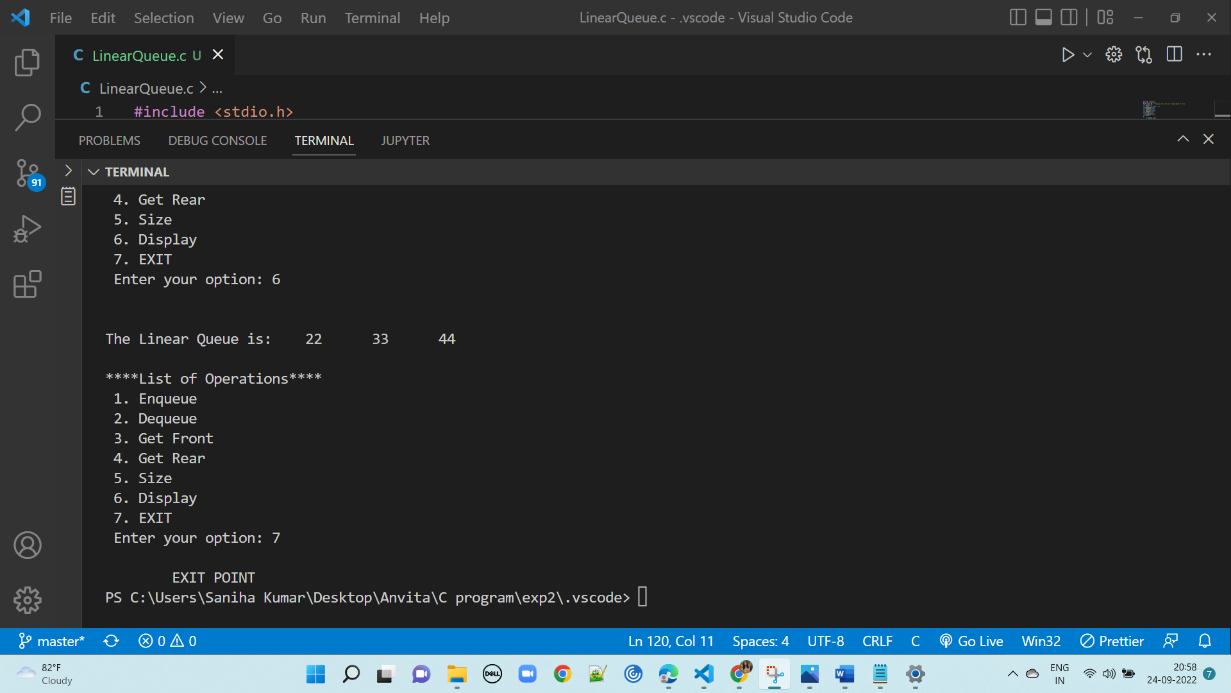












//Write a menu driven code to implement Circular Queue ADT using arrays

#include <stdio.h>

#include <stdlib.h>

#define MAX 10

int queue[MAX];

int front = -1, rear = -1;

void Enqueue(void);

int Dequeue(void);

int GetFront(void);

int GetRear(void);

void size(void);

void display(void);

int main()

{

int option, val;

do {

printf("\n\n\*\*\*\*List of Operations\*\*\*\*");

printf("\n 1. Enqueue");

printf("\n 2. Dequeue");

printf("\n 3. Get Front");

printf("\n 4. Get Rear");

printf("\n 5. Size");

printf("\n 6. Display");

printf("\n 7. EXIT");

printf("\n Enter your option: ");

scanf("%d", &option);

switch (option) {

case 1:

Enqueue();

break;

case 2:

val = Dequeue();

if (val != -1)

printf("\n The number deleted is: %d", val);

break;

case 3:

val = GetFront();

if (val != -1)

printf("\n The first value in queue is: %d", val);

break;

case 4:

val = GetRear();

if (val != -1)

printf("\n The last value in queue is: %d", val);

break;

case 5:

size();

break;

case 6:

display();

break;

case 7:

printf("\n\tEXIT POINT");

break;

}

} while (option != 7);

return 0;

}

int isEmpty() {

return (front == -1 && rear == -1);

}

int isFull() {

return (front == 0 && rear == MAX-1);

}

void Enqueue()

{

int num;

printf("\n Enter the number to be inserted in the queue : ");

scanf("%d", &num);

if (isFull())

printf("\n OVERFLOW");

else if (isEmpty()) {

front = rear = 0;

queue[rear] = num;

}

else if (front != 0 && rear == MAX-1) {

rear = 0;

queue[rear] = num;

}

else {

rear++;

queue[rear] = num;

}

}

int Dequeue()

{

int val;

if (isEmpty()) {

printf("\n UNDERFLOW");

return -1;

}

else {

val = queue[front];

if (front == rear)

front = rear =-1;

else if(front == MAX-1)

front=0;

else

front++;

}

return val;

}

int GetFront()

{

if (isEmpty()) {

printf("\n QUEUE IS EMPTY");

return -1;

}

else {

return queue[front];

}

}

int GetRear(void)

{

if (isEmpty()) {

printf("\nQUEUE IS EMPTY");

return -1;

}

else {

return queue[rear];

}

}

void size(void) {

int count=0;

int i;

if(front > -1 && rear > -1) {

printf("The number of elements in queue: ");

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

count++;

}

printf("%d\n",count);

}

else {

printf("\n The Queue is empty");

}

}

void display() {

int i;

printf("\n");

if (isEmpty())

printf("\n QUEUE IS EMPTY");

else {

printf("\nThe Circular Queue is: ");

if (front < rear) {

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

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

}

else {

for (i = front; i < MAX-1; i++)

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

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

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

}

}

}

