Question-1(a):

#include <stdio.h>

#include <stdbool.h>

#define MAX\_SIZE 100

int queue[MAX\_SIZE];

int front = -1, rear = -1;

bool isEmpty() {

return front == -1;

}

bool isFull() {

return rear == MAX\_SIZE - 1;

}

void enqueue(int value) {

if (isFull()) {

printf("Queue is full. Cannot enqueue.\n");

return;

}

if (isEmpty()) {

front = 0;

}

rear++;

queue[rear] = value;

printf("Enqueued %d\n", value);

}

void dequeue() {

if (isEmpty()) {

printf("Queue is empty. Cannot dequeue.\n");

return;

}

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

if (front == rear) {

front = rear = -1;

} else {

front++;

}

}

bool find(int value) {

if (isEmpty()) {

return false;

}

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

if (queue[i] == value) {

return true;

}

}

return false;

}

int main() {

enqueue(10);

enqueue(20);

enqueue(30);

dequeue();

printf("Is 20 in the queue? %s\n", find(20) ? "Yes" : "No");

printf("Is 40 in the queue? %s\n", find(40) ? "Yes" : "No");

return 0;

}

Question -1(b) :

#include <stdio.h>

#include <stdbool.h>

#define MAX\_SIZE 5

int queue[MAX\_SIZE];

int front = -1, rear = -1;

bool isEmpty() {

return front == -1;

}

bool isFull() {

return (rear + 1) % MAX\_SIZE == front;

}

void enqueue(int value) {

if (isFull()) {

printf("Queue is full. Cannot enqueue.\n");

return;

}

if (isEmpty()) {

front = rear = 0;

} else {

rear = (rear + 1) % MAX\_SIZE;

}

queue[rear] = value;

printf("Enqueued %d\n", value);

}

void dequeue() {

if (isEmpty()) {

printf("Queue is empty. Cannot dequeue.\n");

return;

}

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

if (front == rear) {

front = rear = -1;

} else {

front = (front + 1) % MAX\_SIZE;

}

}

bool find(int value) {

if (isEmpty()) {

return false;

}

int current = front;

do {

if (queue[current] == value) {

return true;

}

current = (current + 1) % MAX\_SIZE;

} while (current != (rear + 1) % MAX\_SIZE);

return false;

}

int main() {

enqueue(10);

enqueue(20);

enqueue(30);

dequeue();

printf("Is 20 in the queue? %s\n", find(20) ? "Yes" : "No");

printf("Is 40 in the queue? %s\n", find(40) ? "Yes" : "No");

return 0;

}

Question -2 :

#include <stdio.h>

#include <stdbool.h>

#define N 4

int board[N][N] = {0};

void printSolution() {

for (int i = 0; i < N; i++) {

for (int j = 0; j < N; j++) {

printf("%d ", board[i][j]);

}

printf("\n");

}

}

bool isSafe(int row, int col) {

for (int i = 0; i < col; i++) {

if (board[row][i])

return false;

}

for (int i = row, j = col; i >= 0 && j >= 0; i--, j--) {

if (board[i][j])

return false;

}

for (int i = row, j = col; j >= 0 && i < N; i++, j--) {

if (board[i][j])

return false;

}

return true;

}

bool solveNQueens(int col) {

if (col >= N) {

printSolution();

return true;

}

for (int i = 0; i < N; i++) {

if (isSafe(i, col)) {

board[i][col] = 1;

if (solveNQueens(col + 1))

return true;

board[i][col] = 0;

}

}

return false;

}

int main() {

if (!solveNQueens(0)) {

printf("Solution does not exist");

}

return 0;

}