**Assingmnet-4**

**NAME:- NAMAN ARORA**

**ROLL\_NO:- 1024030788**

**Q(1)**#include <iostream>

using namespace std;

#define SIZE 5

class Queue {

private:

int arr[SIZE];

int front, rear;

public:

Queue() {

front = -1;

rear = -1;

}

bool isEmpty() {

return (front == -1);

}

bool isFull() {

return (rear == SIZE - 1);

}

void enqueue(int value) {

if (isFull()) {

cout << "Queue is FULL! Cannot insert " << value << endl;

return;

}

if (isEmpty()) {

front = 0;

}

rear++;

arr[rear] = value;

cout << value << " inserted into queue." << endl;

}

void dequeue() {

if (isEmpty()) {

cout << "Queue is EMPTY! Cannot dequeue." << endl;

return;

}

cout << arr[front] << " removed from queue." << endl;

if (front == rear) {

front = -1;

rear = -1;

} else {

front++;

}

}

void peek() {

if (isEmpty()) {

cout << "Queue is EMPTY!" << endl;

} else {

cout << "Front element is: " << arr[front] << endl;

}

}

void display() {

if (isEmpty()) {

cout << "Queue is EMPTY!" << endl;

return;

}

cout << "Queue elements: ";

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

cout << arr[i] << " ";

}

cout << endl;

}

};

int main() {

Queue q;

int choice, value;

do {

cout << "\n--- Queue Menu ---\n";

cout << "1. Enqueue (Insert)\n";

cout << "2. Dequeue (Remove)\n";

cout << "3. Peek (Front element)\n";

cout << "4. Display Queue\n";

cout << "5. Check if Empty\n";

cout << "6. Check if Full\n";

cout << "0. Exit\n";

cout << "Enter your choice: ";

cin >> choice;

if (choice == 1) {

cout << "Enter value to enqueue: ";

cin >> value;

q.enqueue(value);

}

else if (choice == 2) {

q.dequeue();

}

else if (choice == 3) {

q.peek();

}

else if (choice == 4) {

q.display();

}

else if (choice == 5) {

if (q.isEmpty())

cout << "Queue is EMPTY.\n";

else

cout << "Queue is NOT empty.\n";

}

else if (choice == 6) {

if (q.isFull())

cout << "Queue is FULL.\n";

else

cout << "Queue is NOT full.\n";

}

else if (choice == 0) {

cout << "Exiting program...\n";

}

else {

cout << "Invalid choice! Try again.\n";

}

} while (choice != 0);

return 0;

}

**Q(2)**#include <iostream>

using namespace std;

#define SIZE 5

class CircularQueue {

private:

int arr[SIZE];

int front, rear;

public:

CircularQueue() {

front = -1;

rear = -1;

}

bool isEmpty() {

return (front == -1);

}

bool isFull() {

return ((rear + 1) % SIZE == front);

}

void enqueue(int value) {

if (isFull()) {

cout << "Queue is FULL! Cannot insert " << value << endl;

return;

}

if (isEmpty()) {

front = 0;

}

rear = (rear + 1) % SIZE;

arr[rear] = value;

cout << value << " inserted into queue." << endl;

}

void dequeue() {

if (isEmpty()) {

cout << "Queue is EMPTY! Cannot dequeue." << endl;

return;

}

cout << arr[front] << " removed from queue." << endl;

if (front == rear) {

front = -1;

rear = -1;

} else {

front = (front + 1) % SIZE;

}

}

void peek() {

if (isEmpty()) {

cout << "Queue is EMPTY!" << endl;

} else {

cout << "Front element is: " << arr[front] << endl;

}

}

void display() {

if (isEmpty()) {

cout << "Queue is EMPTY!" << endl;

return;

}

cout << "Queue elements: ";

int i = front;

while (true) {

cout << arr[i] << " ";

if (i == rear) break;

i = (i + 1) % SIZE;

}

cout << endl;

}

};

int main() {

CircularQueue cq;

int choice, value;

do {

cout << "\n--- Circular Queue Menu ---\n";

cout << "1. Enqueue (Insert)\n";

cout << "2. Dequeue (Remove)\n";

cout << "3. Peek (Front element)\n";

cout << "4. Display Queue\n";

cout << "5. Check if Empty\n";

cout << "6. Check if Full\n";

cout << "0. Exit\n";

cout << "Enter your choice: ";

cin >> choice;

switch (choice) {

case 1:

cout << "Enter value to enqueue: ";

cin >> value;

cq.enqueue(value);

break;

case 2:

cq.dequeue();

break;

case 3:

cq.peek();

break;

case 4:

cq.display();

break;

case 5:

if (cq.isEmpty())

cout << "Queue is EMPTY.\n";

else

cout << "Queue is NOT empty.\n";

break;

case 6:

if (cq.isFull())

cout << "Queue is FULL.\n";

else

cout << "Queue is NOT full.\n";

break;

case 0:

cout << "Exiting program...\n";

break;

default:

cout << "Invalid choice! Try again.\n";

}

} while (choice != 0);

return 0;

}

**Q(3)**#include <iostream>

#include <queue>

using namespace std;

void interleaveQueue(queue<int>& q) {

if (q.size() % 2 != 0) {

cout << "Queue must have even number of elements!" << endl;

return;

}

int halfSize = q.size() / 2;

queue<int> firstHalf;

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

firstHalf.push(q.front());

q.pop();

}

while (!firstHalf.empty()) {

q.push(firstHalf.front());

firstHalf.pop();

q.push(q.front());

q.pop();

}

}

int main() {

queue<int> q;

int n, value;

cout << "Enter number of elements (even): ";

cin >> n;

if (n % 2 != 0) {

cout << "Number of elements must be even!" << endl;

return 0;

}

cout << "Enter " << n << " elements:\n";

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

cin >> value;

q.push(value);

}

interleaveQueue(q);

cout << "Interleaved Queue: ";

while (!q.empty()) {

cout << q.front() << " ";

q.pop();

}

cout << endl;

return 0;

}

**Q(4)**#include <iostream>

#include <queue>

#include <unordered\_map>

using namespace std;

void firstNonRepeating(string str) {

queue<char> q;

unordered\_map<char, int> freq;

for (char ch : str) {

freq[ch]++;

q.push(ch);

while (!q.empty() && freq[q.front()] > 1) {

q.pop();

}

if (q.empty())

cout << -1 << " ";

else

cout << q.front() << " ";

}

cout << endl;

}

int main() {

string input;

cout << "Enter a string (without spaces): ";

cin >> input;

cout << "First non-repeating characters: ";

firstNonRepeating(input);

return 0;

}

**Q5(A)**#include <iostream>

#include <queue>

using namespace std;

class Stack {

queue<int> q1, q2;

public:

void push(int x) {

q2.push(x);

while (!q1.empty()) {

q2.push(q1.front());

q1.pop();

}

swap(q1, q2);

}

void pop() {

if (q1.empty()) {

cout << "Stack is EMPTY!" << endl;

return;

}

cout << "Popped: " << q1.front() << endl;

q1.pop();

}

void top() {

if (q1.empty()) {

cout << "Stack is EMPTY!" << endl;

return;

}

cout << "Top element: " << q1.front() << endl;

}

void display() {

if (q1.empty()) {

cout << "Stack is EMPTY!" << endl;

return;

}

cout << "Stack elements (top to bottom): ";

queue<int> temp = q1;

while (!temp.empty()) {

cout << temp.front() << " ";

temp.pop();

}

cout << endl;

}

};

int main() {

Stack s;

s.push(10);

s.push(20);

s.push(30);

s.display();

s.top();

s.pop();

s.display();

return 0;

}

**Q5(B)**#include <iostream>

#include <queue>

using namespace std;

class Stack {

queue<int> q;

public:

void push(int x) {

int size = q.size();

q.push(x);

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

q.push(q.front());

q.pop();

}

}

void pop() {

if (q.empty()) {

cout << "Stack is EMPTY!" << endl;

return;

}

cout << "Popped: " << q.front() << endl;

q.pop();

}

void top() {

if (q.empty()) {

cout << "Stack is EMPTY!" << endl;

return;

}

cout << "Top element: " << q.front() << endl;

}

void display() {

if (q.empty()) {

cout << "Stack is EMPTY!" << endl;

return;

}

cout << "Stack elements (top to bottom): ";

queue<int> temp = q;

while (!temp.empty()) {

cout << temp.front() << " ";

temp.pop();

}

cout << endl;

}

};

int main() {

Stack s;

s.push(100);

s.push(200);

s.push(300);

s.display();

s.top();

s.pop();

s.display();

return 0;

}