1- Write a C++ program that creates a queue using the STL queue container. The program should push an element into the queue and then check if the queue is empty. If the queue is empty, it should print "Queue is empty." Otherwise, it should print "Queue is not empty."

اكتب برنامج C++ يقوم بإنشاء قائمة queue باستخدام حاوية قائمة queue STL. يجب أن يدفع البرنامج عنصرًا إلى قائمة queue ثم يتحقق مما إذا كانت قائمة queue فارغة. إذا كانت قائمة queue فارغة، فيجب طباعة "قائمة queue فارغة". وإلا، فإنه يجب طباعة "قائمة queue ليست فارغة."

Output

```
Queue is not empty.
```

Solution

```cpp
// www.gammal.tech

#include <iostream>
#include <queue>
using namespace std;

int main() {
    // Create a queue using STL queue container
    queue<int> q;

    // Push an element into the queue
    q.push(10);

    // Check if the queue is empty
    if (q.empty()) {
        cout << "Queue is empty." << endl;
    } else {
        cout << "Queue is not empty." << endl;
    }

    return 0;
}
```

2- Write a C++ program that creates a queue using the STL queue container. The program should push three elements (10, 20, and 30) into the queue and then print the size of the queue.

اكتب برنامج C++ يقوم بإنشاء قائمة queue باستخدام حاوية قائمة queue STL. يجب أن يقوم البرنامج بدفع ثلاثة عناصر (10 و20 و30) إلى قائمة queue ثم طباعة حجم قائمة queue .

## Output

```
Size of the queue: 3
```

## Solution

```cpp
// www.gammal.tech

#include <iostream>
#include <queue>
using namespace std;

int main() {
    // Create a queue using STL queue container
    queue<int> q;

    // Push elements into the queue
    q.push(10);
    q.push(20);
    q.push(30);

    // Print the size of the queue
    cout << "Size of the queue: " << q.size() << endl;

    return 0;
}
```

---

3- Write a C++ program that creates a queue using the STL queue container. The program should push three elements (10, 20, and 30) into the queue and then print the back element of the queue.

اكتب برنامج C++ يقوم بإنشاء قائمة queue باستخدام حاوية قائمة queue STL. يجب أن يقوم البرنامج بدفع العناصر الثلاثة (10 و20 و30) إلى قائمة queue ثم طباعة العنصر الخلفي لقائمة queue .

## Output

```
Back element of the queue: 30
```

## Solution

```cpp
// www.gammal.tech

#include <iostream>
#include <queue>
using namespace std;

int main() {
    // Create a queue using STL queue container
    queue<int> q;

    // Push elements into the queue
    q.push(10);
    q.push(20);
    q.push(30);

    // Print the back element of the queue
    cout << "Back element of the queue: " << q.back() << endl;

    return 0;
}
```

---

4- Write a C++ program that creates a queue using the STL queue container. The program should push five elements (10, 20, 30, 40, and 50) into the queue and then clear the queue. After clearing the queue, it should print the size of the queue.

اكتب برنامج C++ يقوم بإنشاء قائمة queue باستخدام حاوية قائمة queue STL. يجب أن يدفع البرنامج خمسة عناصر (10 و20 و30 و40 و50) إلى قائمة queue ثم يقوم بمسح قائمة queue . بعد مسح قائمة queue يجب طباعة حجم قائمة queue .

## Output

```
Size of the queue before clearing: 5
Size of the queue after clearing: 0
```

## Solution

```cpp
// www.gammal.tech

#include <iostream>
#include <queue>
using namespace std;

int main() {
    // Create a queue using STL queue container
    queue<int> q;

    // Push elements into the queue
    q.push(10);
    q.push(20);
    q.push(30);
    q.push(40);
    q.push(50);

    // Print the size of the queue before clearing
    cout << "Size of the queue before clearing: " << q.size() << endl;

    // Clearing the queue
    q = queue<int>();

    // Print the size of the queue after clearing
    cout << "Size of the queue after clearing: " << q.size() << endl;

    return 0;
}
```

5- Write a C++ program that demonstrates the use of a queue with user-defined objects. The program should create a queue of Person objects, where each Person object contains a name and an age. Two Person objects, "John" (age 30) and "Alice" (age 25), should be pushed into the queue. Then, the program should print the name and age of the front element of the queue.

اكتب برنامج C++ يوضح استخدام قائمة queue مع الكائنات المعرفة من قبل المستخدم. يجب أن يقوم البرنامج بإنشاء قائمة queue object الشخص، حيث يحتوي كل object شخص على اسم وعمر. يجب دفع two object من شخصين، "John" (30 عامًا) و"Alice" (25 عامًا) إلى قائمة queue . بعد ذلك، يجب أن يقوم البرنامج بطباعة اسم وعمر العنصر الأمامي في قائمة queue .

Output

```
Front element: John, 30
```

Solution

```cpp
// www.gammal.tech

#include <iostream>
#include <queue>
#include <string>
using namespace std;

// Define the Person class
class Person {
public:
    string name;
    int age;

    // Constructor
    Person(string n, int a) {
        name = n;
        age = a;
    }
};

int main() {
    // Create a queue of Person objects
    queue<Person> q;

    // Push Person objects into the queue
    q.push(Person("John", 30));
    q.push(Person("Alice", 25));

    // Print the name and age of the front element of the queue
    cout << "Front element: " << q.front().name << ", " << q.front().age << endl;

    return 0;
}
```

---

6- Write a C++ program that demonstrates how to search for a specific number in a queue. The program should create a queue of integers and allow the user to input a number to search for in the queue. It should then output whether the number exists in the queue or not.

اكتب برنامج C++ يوضح كيفية البحث عن رقم محدد في قائمة queue . يجب أن يقوم البرنامج بإنشاء قائمة queue من الأعداد الصحيحة والسماح للمستخدم بإدخال رقم للبحث عنه في قائمة الانتظار. يجب بعد ذلك إخراج ما إذا كان الرقم موجودًا في قائمة queue أم لا.

## Input

```
Enter the number to search in the queue: 20
```

## Output

```
Number 20 exists in the queue.
```

## Solution

```cpp
// www.gammal.tech

#include <iostream>
#include <queue>
using namespace std;

int main() {
    // Create a queue of integers
    queue<int> q;

    // Push some integers into the queue
    q.push(10);
    q.push(20);
    q.push(30);

    // Ask the user to input a number to search for
    int num;
    cout << "Enter the number to search in the queue: ";
    cin >> num;

    // Flag to indicate if the number is found
    bool found = false;

    // Create a copy of the queue to iterate through
    queue<int> temp = q;

    // Iterate through the queue to search for the number
    while (!temp.empty()) {
        if (temp.front() == num) {
            found = true;
            break;
        }
        temp.pop(); // Remove the front element of the temporary queue
    }

    // Output whether the number is found or not
    if (found) {
        cout << "Number " << num << " exists in the queue." << endl;
    } else {
        cout << "Number " << num << " does not exist in the queue." << endl;
    }

    return 0;
}
```

---

7- Write a C++ program to reverse the elements of a queue using a stack. The program should define a function reverseQueue that takes a reference to a queue of integers as its parameter and reverses the order of its elements using a stack. After reversing the queue, the program should output the original and reversed queue elements.

اكتب برنامج C++ لعكس عناصر قائمة queue باستخدام stack. يجب أن يقوم البرنامج بتعريف دالة ReverseQueue التي تأخذ مرجعًا إلى قائمة queue من الأعداد الصحيحة كمعلمة لها وتعكس ترتيب عناصرها باستخدام stack. بعد عكس قائمة queue يجب على البرنامج إخراج عناصر قائمة queue الأصلية والمعكوسة.

Output

```
Original queue elements: 10 20 30 40
Reversed queue elements: 40 30 20 10
```

# Solution

```cpp
// www.gammal.tech

#include <iostream>
#include <queue>
#include <stack>
using namespace std;

// Function to reverse a queue using a stack
void reverseQueue(queue<int> &q) {
    stack<int> s;

    // Push all elements of the queue into the stack
    while (!q.empty()) {
        s.push(q.front());
        q.pop();
    }

    // Pop elements from the stack and push them back into the queue
    while (!s.empty()) {
        q.push(s.top());
        s.pop();
    }
}

int main() {
    // Create a queue of integers
    queue<int> q;
    q.push(10);
    q.push(20);
    q.push(30);
    q.push(40);

    // Output the original queue elements
    cout << "Original queue elements: ";
    while (!q.empty()) {
        cout << q.front() << " ";
        q.pop();
    }
    cout << endl;

    // Restore the queue to its original state
    q.push(10);
    q.push(20);
    q.push(30);
    q.push(40);

    // Reverse the queue elements
    reverseQueue(q);

    // Output the reversed queue elements
    cout << "Reversed queue elements: ";
    while (!q.empty()) {
        cout << q.front() << " ";
        q.pop();
    }
    cout << endl;

    return 0;
}
```

8- Write a C++ program to calculate the sum of all elements in a queue of integers. The program should iterate through the elements of the queue, calculate their sum, and output the result.

اكتب برنامج C++ لحساب مجموع كل العناصر في قائمة queue الأعداد الصحيحة. يجب أن يقوم البرنامج بالتكرار عبر عناصر قائمة queue وحساب مجموعها، وإخراج النتيجة.

## Output

```
Sum of queue elements: 100
```

## Solution

```cpp
// www.gammal.tech

#include <iostream>
#include <queue>
using namespace std;

int main() {
    // Create a queue of integers
    queue<int> q;
    q.push(10);
    q.push(20);
    q.push(30);
    q.push(40);

    // Initialize sum variable
    int sum = 0;

    // Calculate sum of queue elements
    queue<int> temp = q;
    while (!temp.empty()) {
        sum += temp.front();
        temp.pop();
    }

    // Output the sum of queue elements
    cout << "Sum of queue elements: " << sum << endl;

    return 0;
}
```

---

9- Write a C++ program to calculate the average of all elements in a queue of integers. The program should iterate through the elements of the queue, calculate their sum, divide the sum by the total number of elements, and output the average.

اكتب برنامج C++ لحساب متوسط جميع العناصر في قائمة queue من الأعداد الصحيحة. يجب أن يقوم البرنامج بالتكرار عبر عناصر قائمة queue وحساب مجموعها، وتقسيم المجموع على إجمالي عدد العناصر، وإخراج المتوسط.

## Output

```
Average of queue elements: 25
```

## Solution

```cpp
// www.gammal.tech

#include <iostream>
#include <queue>
using namespace std;

int main() {
    // Create a queue of integers
    queue<int> q;
    q.push(10);
    q.push(20);
    q.push(30);
    q.push(40);

    // Initialize variables for sum and count
    double sum = 0;
    int count = 0;

    // Calculate sum of queue elements and count the number of elements
    queue<int> temp = q;
    while (!temp.empty()) {
        sum += temp.front();
        temp.pop();
        count++;
    }

    // Calculate the average
    double average = sum / count;

    // Output the average of queue elements
    cout << "Average of queue elements: " << average << endl;

    return 0;
}
```

10- Write a C++ program that counts the occurrences of a specific number in a queue of integers. The program should take a queue and a target number as input, iterate through the elements of the queue, count the occurrences of the target number, and output the count.

اكتب برنامج C++ يقوم بعد تكرارات رقم معين في قائمة queue من الأعداد الصحيحة. يجب أن يأخذ البرنامج قائمة queue ورقمًا مستهدفًا كمدخلات، ويكرر من خلال عناصر قائمة queue ويحسب تكرارات الرقم المستهدف، ويخرج العدد.

## Output

```
Occurrence of 10 in the queue: 3
```

## Solution

```cpp
// www.gammal.tech

#include <iostream>
#include <queue>
using namespace std;

// Function to count the occurrences of a number in a queue
int countOccurrence(queue<int> q, int num) {
    int count = 0;
    // Create a temporary queue to traverse the original queue
    queue<int> temp = q;
    while (!temp.empty()) {
        // If the front element of the queue matches the target number, increment count
        if (temp.front() == num) {
            count++;
        }
        temp.pop(); // Remove the front element
    }
    return count; // Return the count of occurrences
}

int main() {
    // Create a queue of integers
    queue<int> q;
    q.push(10);
    q.push(20);
    q.push(10);
    q.push(30);
    q.push(10);

    // Define the target number to count occurrences
    int num = 10;
    // Count the occurrences of the target number in the queue
    int occurrence = countOccurrence(q, num);
    // Output the result
    cout << "Occurrence of " << num << " in the queue: " << occurrence << endl;

    return 0;
}
```