1- Write a C++ program that prompts the user to enter four integers. The program should use a stack to store these integers and print only the odd elements of the stack.

اكتب برنامج C++ يطلب من المستخدم إدخال أربعة أعداد صحيحة. يجب أن يستخدم البرنامج stack لتخزين هذه الأعداد الصحيحة وطباعة العناصر الفردية فقط stack.

Input

```
Enter four integers, pressing Enter after each:
Enter element 1: 2
Enter element 2: 3
Enter element 3: 4
Enter element 4: 5
```

Output

```
Odd elements of the stack: 5 3
```

Solution

```cpp
// www.gammal.tech

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

int main() {
    // Create a stack named myStack
    stack<int> myStack;
    int num;

    // Prompt the user to enter four integers
    cout << "Enter four integers, pressing Enter after each:" << endl;
    for (int i = 0; i < 4; ++i) {
        cout << "Enter element " << i + 1 << ": ";
        cin >> num;
        myStack.push(num);
    }

    // Print odd elements of the stack by popping and displaying them
    cout << "Odd elements of the stack: ";
    while (!myStack.empty()) {
        if (myStack.top() % 2 != 0)
            cout << myStack.top() << " ";
        myStack.pop();
    }

    return 0;
}
```

2- Write a program C++ program to additionally print only the prime numbers among the entered integers using a stack.

اكتب برنامجًا C++ لطباعة الأعداد الأولية فقط بين الأعداد الصحيحة المدخلة باستخدام stack.

Input

```
Enter four integers, pressing Enter after each:
Enter element 1: 1
Enter element 2: 2
Enter element 3: 3
Enter element 4: 5
```

Output

```
Prime elements of the stack: 5 3 2
```

Solution

```cpp
// www.gammal.tech

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

// Function to check if a number is prime
bool isPrime(int n) {
    if (n <= 1)
        return false;
    for (int i = 2; i <= sqrt(n); ++i) {
        if (n % i == 0)
            return false;
    }
    return true;
}

int main() {
    // Create a stack named myStack
    stack<int> myStack;
    int num;

    // Prompt the user to enter four integers
    cout << "Enter four integers, pressing Enter after each:" << endl;
    for (int i = 0; i < 4; ++i) {
        cout << "Enter element " << i + 1 << ": ";
        cin >> num;
        myStack.push(num);
    }

    // Print prime elements of the stack by popping and displaying them
    cout << "Prime elements of the stack: ";
    while (!myStack.empty()) {
        if (isPrime(myStack.top()))
            cout << myStack.top() << " ";
        myStack.pop();
    }

    return 0;
}
```

3- Write a program to count and print the number of even integers entered by the user using a stack.

اكتب برنامجًا لحساب وطباعة عدد الأعداد الصحيحة الزوجية التي أدخلها المستخدم باستخدام stack.

Input

```
Enter four integers, pressing Enter after each:
Enter element 1: 2
Enter element 2: 3
Enter element 3: 4
Enter element 4: 5
```

Output

```
Number of even elements entered: 2
```

Solution

```cpp
// www.gammal.tech

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

int main() {
    // Create a stack named myStack
    stack<int> myStack;
    int num;
    int evenCount = 0;

    // Prompt the user to enter four integers
    cout << "Enter four integers, pressing Enter after each:" << endl;
    for (int i = 0; i < 4; ++i) {
        cout << "Enter element " << i + 1 << ": ";
        cin >> num;
        myStack.push(num);

        // Count even integers
        if (num % 2 == 0) {
            evenCount++;
        }
    }

    // Print the count of even elements
    cout << "Number of even elements entered: " << evenCount << endl;

    return 0;
}
```

4- Write a program to print only the negative integers entered by the user using a stack.

اكتب برنامجًا لطباعة الأعداد الصحيحة السالبة التي أدخلها المستخدم باستخدام stack فقط.

Input

```
Enter four integers, pressing Enter after each:
Enter element 1: 1
Enter element 2: -9
Enter element 3: 5
Enter element 4: -8
```

Output

```
Negative elements of the stack: -8 -9
```

Solution

```cpp
// www.gammal.tech

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

int main() {
    // Create a stack named myStack
    stack<int> myStack;
    int num;

    // Prompt the user to enter four integers
    cout << "Enter four integers, pressing Enter after each:" << endl;
    for (int i = 0; i < 4; ++i) {
        cout << "Enter element " << i + 1 << ": ";
        cin >> num;
        myStack.push(num);
    }

    // Print only negative elements
    cout << "Negative elements of the stack: ";
    while (!myStack.empty()) {
        if (myStack.top() < 0)
            cout << myStack.top() << " ";
        myStack.pop();
    }

    return 0;
}
```

5- Write a C++ program that takes four integers from the user, pushes them onto a stack, and then counts and prints the number of negative elements in the stack.

اكتب برنامج C++ يأخذ أربعة أعداد صحيحة من المستخدم، ويدفعها إلى stack ثم يحسب ويطبع عدد العناصر السالبة في stack .

Input

```
Enter four integers, pressing Enter after each:
Enter element 1: 1
Enter element 2: -1
Enter element 3: -5
Enter element 4: -4
```

Output

```
Number of negative elements: 3
```

Solution

```cpp
// www.gammal.tech

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

int main() {
    // Create a stack named myStack
    stack<int> myStack;
    int num;

    // Prompt the user to enter four integers
    cout << "Enter four integers, pressing Enter after each:" << endl;
    for (int i = 0; i < 4; ++i) {
        cout << "Enter element " << i + 1 << ": ";
        cin >> num;
        myStack.push(num);
    }

    // Count and print the number of negative elements
    int negativeCount = 0;
    while (!myStack.empty()) {
        if (myStack.top() < 0)
            negativeCount++;
        myStack.pop();
    }

    cout << "Number of negative elements: " << negativeCount << endl;

    return 0;
}
```

6- Write a program that takes four integers from the user, pushes them onto a stack, and then prints all the elements between -2 and 8 (inclusive) in the stack.

اكتب برنامجًا يأخذ أربعة أعداد صحيحة من المستخدم، ويدفعها إلى stack ثم يطبع جميع العناصر بين -2 و8 (شاملة) في stack .

Input

```
Enter four integers, pressing Enter after each:
Enter element 1: -1
Enter element 2: 5
Enter element 3: 9
Enter element 4: 11
```

Output

```
Elements between -2 and 8 in the stack: 5 -1
```

Solution

```cpp
// www.gammal.tech

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

int main() {
    // Create a stack named myStack
    stack<int> myStack;
    int num;

    // Prompt the user to enter four integers
    cout << "Enter four integers, pressing Enter after each:" << endl;
    for (int i = 0; i < 4; ++i) {
        cout << "Enter element " << i + 1 << ": ";
        cin >> num;
        myStack.push(num);
    }

    // Print elements between -2 and 8 (inclusive)
    cout << "Elements between -2 and 8 in the stack: ";
    while (!myStack.empty()) {
        int currentElement = myStack.top();
        if (currentElement >= -2 && currentElement <= 8)
            cout << currentElement << " ";
        myStack.pop();
    }

    return 0;
}
```

7- Write a C++ program that takes four integers from the user, pushes them onto a stack, and then checks if the number 5 is found in the stack. Output whether the number 5 is found or not.

اكتب برنامج C++ يأخذ أربعة أعداد صحيحة من المستخدم، ويدفعها إلى stack ثم يتحقق مما إذا كان الرقم 5 موجودًا في stack . إخراج ما إذا تم العثور على الرقم 5 أم لا.

Input

```
Enter four integers, pressing Enter after each:
Enter element 1: 1
Enter element 2: 2
Enter element 3: 5
Enter element 4: 6
```

Output

```
The number 5 is found in the stack.
```

Solution

```cpp
// www.gammal.tech

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

int main() {
    // Create a stack named myStack
    stack<int> myStack;
    int num;

    // Prompt the user to enter four integers
    cout << "Enter four integers, pressing Enter after each:" << endl;
    for (int i = 0; i < 4; ++i) {
        cout << "Enter element " << i + 1 << ": ";
        cin >> num;
        myStack.push(num);
    }

    // Check if the number 5 is found in the stack
    bool found = false;
    while (!myStack.empty()) {
        if (myStack.top() == 5) {
            found = true;
            break;
        }
        myStack.pop();
    }

    // Output whether the number 5 is found or not
    if (found) {
        cout << "The number 5 is found in the stack." << endl;
    } else {
        cout << "The number 5 is not found in the stack." << endl;
    }

    return 0;
}
```

8- Write a program in C++ that demonstrates the usage of a queue. Follow the instructions below:

Create a queue named x.
Push three integer values (5, 6, and 7) into the queue x.
Process all elements in the queue by printing them.
Count the total number of elements processed.
Output the total number of elements processed.
Output the size of the queue after processing.

اكتب برنامجًا بلغة C++ يوضح استخدام قائمة queue. اتبع التعليمات التالية:

قم بإنشاء قائمة انتظار باسم x.
ادفع ثلاث قيم صحيحة (5، 6، و7) إلى قائمة queue x.
معالجة جميع العناصر الموجودة في قائمة queue عن طريق طباعتها.
حساب العدد الإجمالي للعناصر التي تمت معالجتها.
إخراج العدد الإجمالي للعناصر التي تمت معالجتها.
إخراج حجم قائمة queue بعد المعالجة.

Output

```
5 6 7
Total elements processed: 3
Size of the queue after processing: 0
```

Solution

```cpp
// www.gammal.tech

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

int main() {
    // Create a queue named x
    queue<int> x;
    int count = 0;

    // Push integer values 5, 6, and 7 into the queue x
    x.push(5);
    x.push(6);
    x.push(7);

    // Process all elements in the queue and print them
    while (!x.empty()) {
        count++; // Increment the count for each element processed
        cout << x.front() << " "; // Print the front element of the queue
        x.pop(); // Remove the front element from the queue
    }

    // Output the total number of elements processed
    cout << endl << "Total elements processed: " << count << endl;

    // Output the size of the queue after processing
    cout << "Size of the queue after processing: " << x.size() << endl;

    return 0;
}
```

9- Write a C++ program that demonstrates the usage of a queue. Follow the instructions below:

Create a queue named x.
Push three integer values (5, 6, and 7) into the queue x.
Print each element in the queue by incrementing its value by 1.

اكتب برنامج C++ يوضح استخدام قائمة الانتظار. اتبع التعليمات التالية:

قم بإنشاء قائمة queue باسم x.
ادفع ثلاث قيم صحيحة (5، 6، و7) إلى قائمة queue x.
اطبع كل عنصر في قائمة queue بزيادة قيمته بمقدار 1.

Output

```
6 7 8
```

Solution

```cpp
// www.gammal.tech

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

int main() {
    // Create a queue named x
    queue<int> x;

    // Push integer values 5, 6, and 7 into the queue x
    x.push(5);
    x.push(6);
    x.push(7);

    // Process all elements in the queue and print them after incrementing by 1
    while (!x.empty()) {
        // Print the front element of the queue after incrementing by 1
        cout << x.front() + 1 << " ";
        x.pop(); // Remove the front element from the queue
    }

    return 0;
}
```

10- Write a C++ program that demonstrates the usage of two queues. Follow the instructions below:

Create two queues named x and y.
Push three integer values (5, 6, and 7) into the queue x.
Transfer all elements from queue x to queue y.
Print all elements in queue y.

اكتب برنامج C++ يوضح استخدام قائمتين من queue . اتبع التعليمات التالية:

قم بإنشاء قائمتين بالاسم x و y.
ادفع ثلاث قيم صحيحة (5، 6، و7) إلى قائمة queue x.
نقل كافة العناصر من قائمة الانتظار x إلى قائمة queue y.
طباعة جميع العناصر في قائمة الانتظار y.

Output

```
5 6 7
```

Solution

```cpp
// www.gammal.tech

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

int main() {
    // Create two queues named x and y
    queue<int> x;
    queue<int> y;

    // Push integer values 5, 6, and 7 into the queue x
    x.push(5);
    x.push(6);
    x.push(7);

    // Transfer all elements from queue x to queue y
    while (!x.empty()) {
        y.push(x.front()); // Push the front element of queue x into queue y
        x.pop();           // Remove the front element from queue x
    }

    // Print all elements in queue y
    while (!y.empty()) {
        cout << y.front() << " "; // Print the front element of queue y
        y.pop();                  // Remove the front element from queue y
    }

    return 0;
}
```
