CSC 122 001 Computer Science II
Julius Ranoa

Chapter 11 Programming Challenge 11 Prime Number Generation
Write a program that asks the user to enter an integer greater than 1 and prints the list of all prime numbers less than or equal to the number entered. Your program should use a predicate that determines whether a given integer is composite. The program should generate the list of prime numbers less or equal to X by adding all positive integers greater than 1 to a vector and then using *remove_if* function and the predicate to remove all composites from the vector.

Screenshot of runtime.

```
Please enter a number greater than 1: 101

The primes found under this number are:
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101

This number, 101, is also prime.
Process finished with exit code 0


Please enter a number greater than 1: 100

The primes found under this number are:
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97

This number, 100, is NOT prime.
Process finished with exit code 0
```

Files included: (1) IsComposite.h, (2) main.cpp

**IsComposite.h**

```
#ifndef CH11_PR11_PRIME_NUMBER_GENERATION_ISPRIME_H
#define CH11_PR11_PRIME_NUMBER_GENERATION_ISPRIME_H

class IsComposite {

public:
    bool operator()(int n) {
        for (int i = 2; i < n; i++) {
            if (n % i == 0) {
                return true;
            }
        }
        return false;
    }

};


#endif //CH11_PR11_PRIME_NUMBER_GENERATION_ISPRIME_H
```

**main.cpp**

```cpp
#include <iostream>
#include <vector>
#include "IsComposite.h"

int main() {
    int limit;
    std::cout << "Please enter a number greater than 1: ";
    std::cin >> limit;
    std::cout << "\n";

    if (limit <= 1) {
        std::cout << "Invalid number. Re-run the program \n";
        std::cout << "and pass a proper argument";
        return -1;
    }

    // Fill the vector with 2 to limit
    std::vector<int> primes;
    for (int i = 2; i <= limit; i++) {
        primes.push_back(i);
    }

    // Remove all composite numbers from vector
    auto fi = std::remove_if(primes.begin(), primes.end(), IsComposite());
    /* Lambda version of the above statement.
    auto fi = std::remove_if(primes.begin(), primes.end(),
        [ ](int n){
            for (int i = 2; i < n; i++) {
                if (n % i == 0) {
                    return true;
                }
            }
            return false;
        }
    );
    */
    primes.erase(fi, primes.end());

    std::cout << "The primes found under this number are: \n";
    for (int i : primes) {
        std::cout << i << " ";
    }
    std::cout << "\n\n";

    if (primes.back() == limit) {
        std::cout << "This number, " << limit << ", is also prime.";
    } else {
        std::cout << "This number, " << limit << ", is NOT prime.";
    }
    return 0;
}
```