Programming #5B: Prime Numbers & Functions

This programming assignment is a follow-up version of assignment #4B on prime numbers. We want to wrap up part of the C++ code we have in #4B to create separate non-void functions to deal with prime numbers. By doing so, we can call these functions to get important information regarding prime numbers and use the information to accomplish tasks related to prime numbers. Once these functions are defined, we call them wherever we need in our program, instead of rewriting the same C++ code again here and there in the main function. For example, in this particular assignment, we are going to call these functions from your main function. This makes our program conceptually clearer and more readable. After you finish this assignment #5B, compare your code to what you have for #4B. You should see the benefit of defining and calling functions in your program, instead of writing a flat piece of code in a single main function to do everything.

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

//Define the isPrime function in the following
bool isPrime(int n)
{      //Fill in your code for the isPrime function.
       //It should return either true or false based on
       //     whether n is a prime number.
}

//Define the printAllPrimesLessThan function in the following
void printAllPrimesLessThan(int n)
{      //Fill in your code for the printAllPrimesLessThan function.
       //It should print out all the prime numbers
       //     that are less than n.
}

//Define the getTheNthPrime function in the following
int getTheNthPrime(int n)
{      //Fill in your code for the getTheNthPrime function
       //It should return the nth prime number.
}

//Define the main function in the following
int main()
{      //Fill in your code for the main function
}
```

First of all, read the detailed descriptions regarding the implementation of the following three functions in your .cpp source code file

- ***bool isPrime(int n)***

- ***void printAllPrimesLessThan(int n)*** and
- ***int getTheNthPrime(int n).***

(1) Define the prime-number-testing function ***bool isPrime(int n) {…your code …}*** : The task of this function is to examine the integer value (provided each time the user calls this function) in the parameter ***n*** and return either *true* or *false* based on whether ***n*** is a prime number. To accomplish this, you can polish and revise your C++ code for part I of programming assignment #4B to determine the number of non-trivial factors of ***n***. Then you can return *true* to indicate *n* is a prime number if the number of non-trivial factors of ***n*** is 0. Otherwise, you return *true* to indicate *n* is not a prime number. **Note that this function should be a quiet function. Do not** ask for user input from *cin* in this function. **Do not** print out any message through *cout* in this function either.

(2) *Define the function* **void printAllPrimesLessThan(int n) {…your code …}** *:* The task of this function is to print out all the prime numbers less than the given positive integer value (provided when the user calls this function) stored in the parameter ***n***. To accomplish this, in your C++ code for this function you should set up a loop to examine all the integers ***i*** from 2 to ***n-1*** such that in each iteration in the loop you will call the ***isPrime(i)*** to check whether ***i*** is a prime number and will print out ***i*** if it is a prime number. *This function **should not** ask for user input from* cin *but **does** print out messages to the screen through* cout.

(3) Define the function ***int getTheNthPrime(int n) {…your code …}*** : The task of this function is to find and return the *n-th prime number* according to the integer value (provided when the user calls this function) stored in the parameter ***n***. To accomplish this task, in your C++ code for this function you should set up a loop to examine all the integers ***i*** from 2, 3, … and up until you have seen *n prime numbers.* You need to have an integer local variable *primeCount* (initialized to 0) to keep track of the number of prime numbers you have seen in the process. On each iteration of the loop, you will call the ***isPrime(i)*** to check whether ***i*** is a prime number and will increase the count in *primeCount* by 1 if it is a prime number. If *primeCount* is equal to *n*, you should immediately return the current integer value ***i*** to finish the function call. ***Note that this function should be a quiet function. Do not*** *ask for user input from* cin *in this function. **Do not** print out any message through* cout *in this function either.*

Finally write the main function to call and test the functions defined above:
Like the ATM service program you wrote for programming assignment 4A, in your main function you should set up a loop to repeatedly do the following things in each iteration of the loop until the user chooses the *quit* option:

- **display a menu** showing information about four options: (i) ***p*** for testing whether a number *n* (to be given by the user) is a prime number, (ii) ***l*** for printing a list of all prime numbers less than a number *n* (*n* to be given by the user), (iii) ***n*** for seeing the *n th* prime number (*n* to be given by the user), and (iv) ***q*** for quitting the program,
- **prompt the user to type in a character** to choose one of the options, and then prompt the user to provide a positive integer *n*, and
- **serve the user according to the user's choice**:
    - If the user's choice is ***p***, you should ask the user to enter an integer, read it and store it in a local integer variable *n*, then call ***isPrime(n)*** to determine

whether n is a prime number, and display a message according to the value returned by the call. In other words, you'll have statement like

```
if ( isPrime(n) == true)
{
}
```

- If the user's choice is ***l***, you should ask the user to enter an integer, read it and store it in a local integer variable *n*, then call the ***printAllPrimesLessThan(n)*** to let it display the all the prime numbers less than *n.*
  In other words, you'll have statement like

  ```
  printAllPrimesLessThan(n) ;
  ```

  .
- If the user's choice is ***n***, you should ask the user to enter an integer, read it and store it in a local integer variable *n*, call ***getTheNthPrime(n)*** to get the n-*th* prime number, and then display it on the screen.
  In other words, you'll have statement like

  ```
  cout << " The " << n << " prime is "
       << getTheNthPrime(n) << endl;
  ```

  .
- If the user's choice is ***q***, quit the program after displaying a thank-you message.
- If the user's choice is none of the above, display a message saying that this is not a valid option.