Programming Assignment #4B

Prime Number Testing: A prime number *n* is a natural number (i.e. a non-negative integer) which is greater than *1* and is not divisible by any other natural number other than *1* and *n*. In other words, *n* has no non-trivial factors other than *1* and *n*. So for example, 2 is the smallest prime number. To test whether a natural number *n* is a prime number or not, you can set up a loop to test every natural number between *1* and *n* (i.e. natural numbers from *2* to *n-1*) one by one to see whether *n* is divisible by any of them and count the number of non-trivial factors for *n* you have seen in the whole process. If *n* is not divisible by any of them, the number of non-trivial factors for *n* is 0 and *n* is a prime number; otherwise n is not a prime number.

Programming Specification: Your main function should have the following three parts of code one after another:

Part I: Prompt the user to input a natural number *n*, determine and display all the factors of *n* between *1* and *n* in an increasing order, and then report how many factors *n* has in total and whether *n* is a prime number or not.

Hint: Declare a variable to count the number of non-trivial factors of *n* between 1 and *n* (i.e. natural numbers from 2 *to* n-1*)*. This count should be set to 0 before the following loop starts. Set up a loop to determine the number of natural numbers between *1* and *n (i.e*. natural numbers from 2 to *n-1)* that can divide n. On each iteration of the loop, you should check one of these candidates to see whether it can divide *n.* If the number does divide *n,* you should display a message to say it is a factor of n, and you will also increase the count of factors of *n* seen so far by 1. After finishing the loop, you can write statements to tell the user the total number of non-trivial factors you have found for *n*, and then report whether n is a prime number based on whether the number of non-trivial factors is 0 or not.

Part II: Find and display every prime number less than or equal to 100 and determine in total how many prime numbers are less than or equal to 100.

Hint: You need to have one loop nested inside another loop. *The outer loop is responsible to pick numbers from 2 to 100 one by one, one for each iteration. The inner loop is responsible for determining whether the number picked by the outer loop is a prime number*. Note that the count of the number of divisors should be reset to 0 each time before the **inner** loop is started. Therefore the setting the inner loop here will be very similar to what you have in (1) above in order to test whether the number picked is a prime number, except that you don't want to display the factors of that number. If the inner loop determines that the number examined is a prime number, display a message about this newly found prime number, and increase the count of prime numbers found by one.

Part III: Find and display the first 30 Prime numbers.

Hint: It is very similar to (2) above. Again, you need to have one loop nested inside another loop. The outer loop picks 2, 3, 4, and so forth with one number for each iteration. Note that the count of the number of divisors should be reset to 0 each time before the **inner** loop is started. The outer loop should continue as long as you haven't seen 30 primer numbers. The inner loop is like what you have in (2) above to test whether the number picked is a prime number. In the inner loop when you determine that the number picked by the outer loop is a prime number, display a message about this newly found prime number, and increase the count of prime numbers found by one.