Linköping University
Department of Science and Technology/ITN
Aida Nordman

Introduction to Programming (C++)

TND012

Laboration 4

Course goals

- To write programs using functions.
- To structure a program with several sub-tasks corresponding to different functions.

Preparation

You must perform the tasks listed below before the start of the lab session on week 39. In each lab session, your lab assistant has the possibility to discuss about three questions with each group. To make the best use of your time during the lab session, it is important that you read the description of all exercises in this lab and do the indicated preparation steps.

1. Review the concepts and the examples presented in Fö 7. See for instance the example program **`example.cpp`** (available in code folder for Fö 7).
2. Review also the examples presented in lesson 3.
3. Prepare step 0 and step 1 of the exercise described below.

Your source files should be placed in the folder **`TND012\Labs\Lab4`**.

Presenting solutions and deadline

The exercise in this lab is compulsory and you should demonstrate your solution during your lab session on **week 39**. After week 39, if your solution for lab 4 has not been approved then it is considered a late lab. This lab must be presented latest on your lab session of week 42.

We remind you that your code for the lab exercises cannot be sent by email to the staff and that at most two labs can be presented in a lab session

Before presenting your code make sure that it is readable, well-indented, and that the compiler issues no warnings[1]. Moreover, **you are not allowed to use global variables**, i.e. all variables must be either declared inside the **`main`** or inside a function. Otherwise, your lab won't be approved.

If you have any specific question about the exercise, then send us an e-mail. Be short and concrete, otherwise you won't get a quick answer. You can write your e-mail in Swedish. Add the course code and your study programme to the e-mail's subject, e.g. "**`TND012/ED:`** …".

[1] Follow the instructions given in the appendix of lab 1.

Exercise

Write a menu driven program to handle prime numbers as the running example in the last page shows. An integer **N > 1** is a prime number if the only divisors of **N** are **1** and **N** (i.e. **1** and **N** are the only integers that divide **N** with a reminder equal to zero). For instance, **2** and **7** are prime numbers but **21** is not a prime number.

File **uppgift1.cpp** contains already the basic structure of your program. It can be compiled and executed (no output is produced). Try to understand the code first. Then follow all the steps given below.

Step 0

Start by adding to the **main** function in the file **uppgift1.cpp** the necessary code to display the program's menu with the following options:

1. Test prime
2. Next prime
3. Previous prime
4. Display primes
5. Exit

Then, add the code for requesting and reading the user option. Finally, compile and execute your program. Correct any errors you detect at this point, before proceeding with the next steps.

Step 1

Add skeleton code[2] for handling the user option. This skeleton code[3] should be completed when going through the next steps. Nevertheless, write the code of your **main** function in such way that you can compile and run the program. Recall that there are five possible user options.

Step 2

Write a function **test_prime()** that tests whether a given integer **N ≥ 1** (input parameter) is a prime number. If **N** is prime then the function returns **true**. Otherwise, the function returns **false**. Note that **true** and **false** are values of type **bool**[4]. Then add a function call to **test_prime()** in the **main** function. Find the suitable place in **main** to add this function call.

Compile and run your program. Correct any errors you detect at this point, before proceeding with the next steps.

Step 3

Write a function **next_prime()** that given an integer **N ≥ 1** (input parameter), computes and returns the smallest prime **P ≥ N**. Note that function **test_prime()** can be useful to compute **P**. Then, add a function call to **next_prime()** in the **main** function. Find the suitable place in **main** to add this function call.

Compile and run your program. Correct any errors you detect at this point, before proceeding with the next steps.

---

[2] Skeleton code contains only dummy code and some high-level program structures, but it compiles.
[3] **Hint**: use a **switch**-statement.
[4] Boolean variables were introduced in Fö 4, see example on slide 7.

Step 4

Write a function **`prev_prime()`** that given an integer **`N ≥ 1`** (input parameter), computes and returns the largest prime **`P ≤ N`**. Note that function **`test_prime()`** can be useful to compute **`P`**. Then, add a function call to **`prev_prime()`** in the **`main`** function. Find the suitable place in **`main`** to add this function call.

Compile and run your program. Correct any errors you detect at this point, before proceeding with the next steps.

Step 5

Write a function **`display_primes()`** writes all prime numbers in the interval **`[1,N]`** given an integer **`N ≥ 1`** (input parameter). Note that function **`test_prime()`** can be useful for this task, too. Then add a function call to **`display_primes()`** in the **`main`** function. Find the suitable place in **`main`** to add this function call.

Compile and run your program. Correct any errors.

```
===============
1. Test prime
2. Next Prime
3. Previous Prime
4. Display Primes
5. Exit
===============

Your choice: 1
Test prime?  21

21 is not a prime.

===============
1. Test prime
2. Next Prime
3. Previous Prime
4. Display Primes
5. Exit
===============

Your choice: 2
Next prime?  15

Next prime of 15 is 17.

===============
1. Test prime
2. Next Prime
3. Previous Prime
4. Display Primes
5. Exit
===============

Your choice: 3
Test prime? 15

Previous prime of 15 is 13.
```

```
===============
1. Test prime
2. Next Prime
3. Previous Prime
4. Display Primes
5. Exit
===============

Your choice: 4
N? 15

Primes in[1,15]

2 3 5 7 11 13

===============
1. Test prime
2. Next Prime
3. Previous Prime
4. Display Primes
5. Exit
===============

Your choice: 10

Wrong choice!!!

===============
1. Test prime
2. Next Prime
3. Previous Prime
4. Display Primes
5. Exit
===============

Your choice: 5

Exiting …
```

Lycka till ☺