CS 210 - Fundamentals of Programming I
Spring 2019 - In-class Exercise 1 for 01/15/2019

Detach the answer sheet that accompanies this exercise. Answer the questions during or after the lecture and turn in it in along with your introduction sheet.

Setting up CodeBlocks

Launch CodeBlocks. The first time you use CodeBlocks on a machine it will ask you a few questions:

- In the Compiler auto-detection dialog, select GNU GCC Compiler and click OK.
- For File associations, choose 'Yes, associate Code::Blocks with C/C++ file types' and click OK.

You should maximize the window if it is not full size.

You may need (or want) to configure some of the settings under the Settings menu. For this course, you must configure the compiler to "Enable all compiler warnings...[-Wall]". This is done by going to the Settings menu, selecting Compiler, checking the appropriate box under the Compiler Flags tab, and clicking OK.

To start writing a new program in CodeBlocks, you do the following steps:

1. To create a new project, just click on Create a new project (or go to the File menu and choose New, then Project.) This gives you the dialog box to create a new project. Click on Console Application, then Go. This will bring up the Console application wizard. Click Next. Choose C for the language, then click Next.

2. For today's project, give it the title "tutorial". Browse to a folder to create the project in. It is suggested that you create a folder on your network drive for this course, e.g., "CS 210". Select your UE network drive (CodeBlocks requires you to use a drive letter, e.g. O:), then click Make New Folder, give the new folder a name, and click OK. Then click Next. (After the first time, just browse to this folder.)

3. The Compiler option should be set to GNU GCC Compiler. Click Finish.

4. CodeBlock creates a "workspace" that contains one or more CodeBlock projects. This is depicted in the Management window on the left as a hierarchy. Usually there will be only one project, but if there are more than one, the **bolded** project is the current project ("tutorial" in this case, since it is the only one). Clicking on the + next to Sources shows the main program file called "main.c". Double-click on main.c to bring it up in an edit window. The file contains the simple "Hello World" program.

Writing a program

The following program is a typical example of a problem statement and implementation that includes the analysis and design for the program as part of the comments. It will be used as part of the lecture demonstration.

Problem Statement

Write a program to read in an integer and print out the integer, its square, and its cube.

Implementation

Here is a C program implementing this program. Note that the comments contain the analysis and design of this program.

```
/* File: main.c
   Read in an integer and computes its square and cube.

   Input:   An integer
   Output:  Its square and cube
   ----------------------------------------------------------------
   Class: CS 210                          Instructor: Dr. Deborah Hwang
   Assignment: In-class Exercise 1        Date : August 27, 2015
   Programmer: <fill in your name>
*/

#include <stdio.h>   // scanf, printf

int main ()
{
    // Analysis: The data used or computed by this program
    int n,           // input from console
        n_squared,   // square of input
        n_cubed;     // cube of input

    // Design: The steps to solve the problem
    // 1. Read input from the console
    printf ("Please enter an integer: ");
    scanf ("%d", &n);

    // 2. Compute square and cube of input
    n_squared = n * n;
    n_cubed = n * n_squared;

    // 3. Display results
    printf ("\nThe input integer is %d.\n", n);
    printf ("Its square is %d, and its cube is %d.\n", n_squared, n_cubed);

    return 0;
}  // end main
```

Name: _____

CS 210 - Fundamentals of Programming I
Spring 2019 - In-class Exercise 1 for 01/15/2019

(10 points) Answer the following questions regarding the program written for this class. Turn in this sheet and your introduction sheet at the end of class.

1. What are the (names of the) variables in this program?

2. What is the type of the variables?

3. What kind of values can the variables hold?

4. What is the input statement in this program?

5. How many output statements are there in this program? Give the first one in the program.

6. What is the purpose of the '`%d`' in the `printf` statement?

7. What does an assignment statement do? Give the first one in the program.

8. What arithmetic operator is used in this program? What operation does it perform?