National University of Singapore
School of Computing
CS1010X: Programming Methodology
Semester II, 2019/2020

Mission 16
C Madness

Release date: 02 May 2020
Due: 31 May 2020, 23:59

Required Files

- mission16.zip

Information

This mission will give you a taste of how programs are written in C and hopefully also an appreciation of the differences between C and higher-level languages like Python. There are 3 distinct tasks, each with one or more tasks to complete. When you are done, copy only the function(s)/struct(s) you were asked to modify or implement into Coursemology.

Getting Started

You will first need access to a C compiler and source editor. The easiest way is to use an online coding site (e.g. `http://www.tutorialspoint.com/compile_c_online.php`). However, for those who are planning on exploring C programming a little more in depth, it is best that you install and learn how to use the GNU Compiler Collection (GCC):

- **Mac OS X:** Install Apple Xcode from the App Store (it's free), then open the Terminal app and type "`sudo gcc`". You should be prompted to enter your password, followed by an installation process. If you see "`gcc: fatal error: no input files`" or something similar after entering your password, GCC is already installed.
- **Windows:** Cygwin (`https://cygwin.com/install.html`) is a shell that gives you access to many Linux command-line tools, including GCC. Run the installer and pick "http://download.nus.edu.sg" as the download site. When asked to select packages, expand the Devel branch and make sure "gcc-core" is selected.

  The Cygwin Terminal window will give you access to gcc — just place the mission files in the "C:\Cygwin\home\UserName" directory to access them in the terminal.
- **Others:** Do let us know if you're on some other operating system and we can help you out. Though if you're on a Linux distribution, chances are GCC is already installed by default.

C programs must first be *compiled* before you can run them. This means your code is translated into binary instructions for your processor to directly evaluate, which makes C programs extremely fast. To compile the first task with GCC, type the following in the terminal:

```
$ gcc -Wall -o grades grades.c
```

Note that you will first need to use the 'cd' (change directory) command to navigate to the folder containing the question files. The `-Wall` option tells GCC to output any and all compilation warnings, while the `-o grades` option tells GCC to name the executable file "`grades`". If successful, you should be able to run the program by doing:

```
$ ./grades
```

You may edit the `.c` and `.h` files with any text editor (or a specialized C code editor) to do the tasks. It will be up to you to find an editor that you are comfortable with.

If none of these instructions make sense to you, please come to a remedial session to get it sorted out. :)

Task 1: Student Grading (5 marks)

Files: `grades.c`
Items: 3

You are to complete a simple program for printing the level, grade and remarks given to a student from his/her EXP gained. The details are provided in the template file.

Make sure you test your program thoroughly. At the very least, you should get the same output as the sample executions provided. This applies to the other tasks as well.

If you encounter the error "`undefined reference to 'pow'`" during compilation, compile using the -lm option like this:

```
$ gcc -Wall -lm -o grades grade.c
```

Task 2: Structured Geometry (5 marks)

Files: `geom.c`
Items: 2

This task deals with structures and arrays of structures. The first subtask requires you to complete a function for computing the smallest rectangle that will enclose all given rectangles.

For the second subtask, you are to create an appropriate struct to return multiple values from the "`make_circle`" function. Be sure to uncomment the lines in the "`print_circle_in`" function and study what they are expecting from the structure.

Task 3: Odd Pointers (5 marks)

Files: `oddity.c`
Items: 1

This task focuses on pointers. You are to complete the "`copy_odd`" function for copying all *odd* characters from the source string to the destination string. The twist is that **you are not allowed to define any additional variables** — only use the pointer variables provided.