ENEE 140 Lab 1

Lab instructions

This handout includes instructions for the recitation sessions on Wednesday and Friday. Follow these instructions to familiarize yourself with tools we will use in ENEE 140. There is no homework due this week. To prepare for the next lecture, complete the reading assignment and try to solve the weekly challenge.

1 Learn to use the lab computers

1. The laboratory has computers running the Windows operating systems. It is ok if you prefer to bring your own laptop and use it during the recitation. However, we recommend you to learn how to use the lab computers just in case.

2. To access the lab computers, you need an ECE account and password. Ask your UTF/GTA or contact the help desk if you have problems with this.

3. Log in the computer in the lab with either your ECE account or the temporary one from your UTF/GTA.

4. Open a web browser on the lab computer and visit OIT webpage (`http://www.helpdesk.umd.edu/`) for information about your account—in particular, how to set a secure password.

2 Familiarize yourself with the class resources

Class Web page. Open `http://ter.ps/enee140` in your browser. Click on the "Syllabus" tab. Here you can find the tentative schedule for the rest of the semester, as well as all the materials and assignments.

Elms. Open `https://elms.umd.edu` in your browser.

1. Click on "Login to ELMS" and log in the ELMS system with your university ID and choose the course ENEE140.

2. Familiarize yourselves with the structure of the Elms system

   - Grades: all scores (quizzes, homeworks, projects, exams)
   - Quizzes: all quizzes and surveys (note that some are due this week!)
   - Panopto Recordings: videos posted occasionally, to supplement missed lectures.

Piazza. Sign up for the class message board at `https://piazza.com/umd/spring2025/enee140/home`.

- The instructors will use Piazza for important announcements. Check the message board at least twice a week for new posts.
- You are also encouraged to use the message board extensively, to ask for help or to answer questions posted by your classmates.

3 Install and configure the CLion IDE on your laptop

Download CLion from `https://www.jetbrains.com/clion/` and install it on your computer. If you have a Windows PC, you also need to install Cygwin (`https://www.cygwin.com/`). Make sure you install at least the following Cygwin packages: `gcc-g++, gcc-core, gdb, make, man-pages-posix`. You can find more information about configuring CLion on Windows at `https://www.jetbrains.com/help/clion/quick-tutorial-on-configuring-clion-on-windows.html`.

The first time when you run CLion, you will have to configure some settings. You can watch the "Introducing CLion" video from `https://www.jetbrains.com/clion/documentation/` to see what these settings should look like. Also, take a look at the "Quick Start" guide at `https://www.jetbrains.com/clion/help/quick-start-guide.html`.

CLion is commercial software, but students can get free licenses by applying here: `https://www.jetbrains.com/student/`.

4 Learn to create a new project in CLion

After you are done with the configuration, create a new project. These instructions will show you how to create a C project called `hello_world`, which includes one C source file called `hello_world.c`

Go to the "File" menu and click on "New Project ..." In the dialog, select "C executable" (this is important—other selections won't work) and specify the path where you want CLion to save your project and the name of the project. Let's name this project `hello_world`.

CLion will create a C source file called `main.c`. Right click on this file and select "Refactor" → "Rename...". Rename the file to `hello_world.c`.

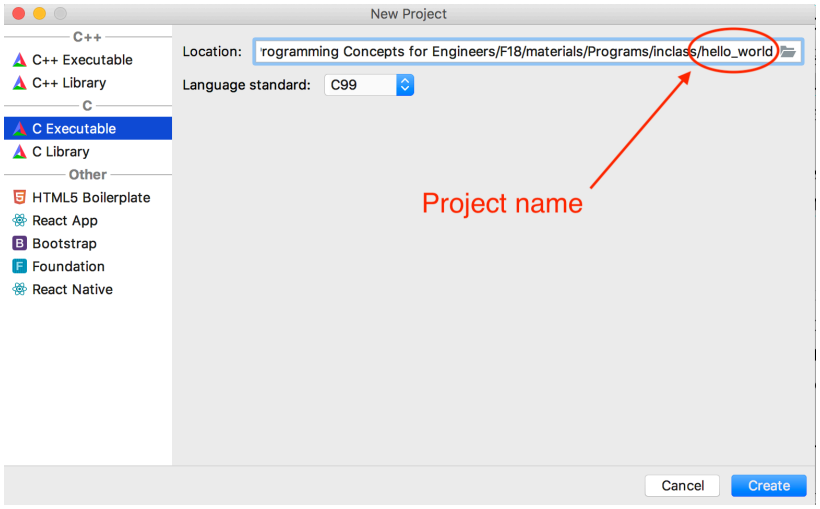
Click on "Reload changes" to update the project. To check that this worked, open the `CMakeLists.txt` file and make sure the `add_executable` directive looks like this:

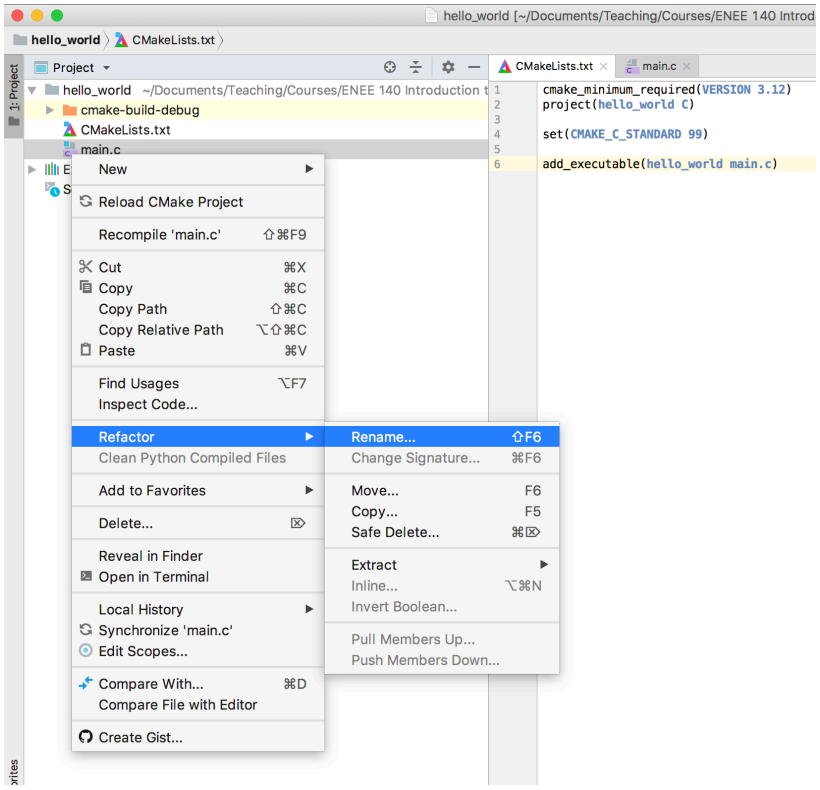
```
add_executable(hello_world hello_world.c)
```

Congratulations! You are now ready to start writing your first C program.

5 Write your first program

1. Create a new C project, called `hello_world`.
2. Within this project, create a new source file named `hello_world.c`.
3. Type in the "Hello World" program from the top of page 7 of the textbook.





4. Compile and run the program. You should see the message "hello, world" in the Console pane of CLion.

5. Try to understand how this program works.

6 Checklist

Make sure that you get the following done before the next lab:

- ☐ Get your ECE account so you can access the computers in the lab.
- ☐ Get the textbook.
- ☐ Set up your IDE (CLion) and write your first program.
- ☐ Sign up for the Piazza message board and browse the Elms system.
- ☐ Visit `http://www.eskimo.com/~scs/cclass/krnotes/top.html` and learn where you can find information about the textbook chapters that are unclear.
- ☐ Remember the name and email of your TA.
- ☐ Take a look at the class Web page and find the office hours (either Dr. Dumitraș's or of any TAs) that best fit your schedule. If you cannot make any of them, contact Dr. Dumitraș.

7 Familiarize yourself with the GRACE system

1. Accessing GRACE UNIX from your PC or Laptop

   (a) If you plan to work on any computer other than those in the lab, you need to make sure that the computer has the software needed to connect to the GRACE system. Download and install such software, if needed; most of these applications are available for free.

   - SSH client and terminal
     - **Windows**: Install **PuTTY**, a free application that enables you to log in to your GRACE UNIX account from your PC or laptop using the ssh protocol.
     - **Mac**: The **Terminal** application (found under "Applications/Utilities") provides similar functionality. You will have to invoke `ssh` on the command line.
   - SCP/SFTP client
     - **Windows**: **WinSCP**, a free application for secure file transfer between your GRACE UNIX account and your PC/Laptop.
     - **Mac**: You may use the `scp` or `sftp` commands from the **Terminal**. Alternatively, you may install **CyberDuck**, a free Mac application that provides functionality similar to WinSCP.

   (b) Access the GRACE UNIX system from the lab computer. You will need your GRACE account and password to do it (we don't have temporary accounts for you). You can find more information about this here: `http://www.grace.umd.edu/help.html`.

   (c) If you are not sure whether you have a GRACE (now known as TerpConnect) account, check the "account information" at OIT webpage (`http://www.helpdesk.umd.edu/`). Ask your TA if you need help.

2. What is UNIX?

(a) UNIX is an operating system, like Windows. An operating system manages the activities and resources of the computer. Most operating systems are written in C.

(b) UNIX was designed and developed in the late 60s and 70s at Bell Labs. UC Berkeley has also contributed a lot, through the Berkeley Software Distribution (BSD).

(c) Modern operating systems that are derived from UNIX include Linux and Mac OS X.

(d) UNIX allows multiple applications running at the same time (multitasking); it allows multiple users to use the same system at the same time (multiuser); it provides shells that allow the user to type in commands and to launch applications.

(e) In UNIX, each user has a separate working space, called an account. The system identifies users by their usernames. Also called login names or account names.

3. UNIX commands for managing user accounts

(a) `whoami`: display the effective current username.

(b) `who`: display the users who are currently logged in on the system, it shows their username, from where they are working, etc.

(c) `finger tdumitra`: display the information of user with username tdumitra.

(d) `passwd`: change your password, must know the old/current one.

(e) `renew`: enter your password again to renew your authentication ticket.

(f) `logout`, `exit`, `bye`: exit from a login session.

(g) Sometimes, the UNIX commands you issue can become very long. To make the commands easier to understand, you can type a `\` and continue typing the command on the next line, for example:

```
finger \
  tdumitra
```

(h) Remember to log out every time before you leave your computer, and do not share your password with others.

Question 1: log in your GRACE UNIX account, type in the following commands and read the output on the screen (you can also write down the output if you want):

1) `whoami`

2) `who`

3) `finger xxx` (where xxx is some username from the output of the `who` command)

4) `finger yyy` (where yyy is your own login ID or account name)

4. The UNIX file system

(a) The UNIX file system consists of files and directories. A file is a collection of data. A directory is a "file folder" that stores one or multiple files.

(b) The UNIX file system is hierarchical, in the sense that a directory can have sub-directories. (Tip: using directories and sub-directories to organize your files.)

(c) root: is the top directory for all users (/) and for each individual user ( ).

(d) . and .. directories: . (single dot) is the current working directory; .. (double dots) is the parent directory of the current working directory.

(e) `mkdir`: create a new directory

(f) `cd`: change directory.

(g) `pwd`: print the current working directory

(h) `rmdir`: delete a directory, be careful about this. There is no "Recycle Bin" or "Trash" in UNIX for you to retrieve the deleted items conveniently. You may lose the directory permanently.

(i) `mv`: rename a directory (short for move).

(j) `ls`, list the contents in a directory or directories.

Question 2: After logging into your GRACE account, create the following directory structure:

1) a directory called ENEE140 under your home directory
2) 3 directories under ENEE140: Lab, Project
3) 3 directories under Project: Proj1, Proj2, Proj3
4) 2 directories under Lab: Week01, Week02

Question 3: Assuming that you are currently at directory `ENEE140/Lab/Week01` (you can get there from your home directory by the following command: `cd ENEE140/Lab/Week01`), which directory each of the following command will take you and how to get back to `ENEE140/Lab/Week01` from that directory? The answer for part 1) is given as an example. Try these on the computer to confirm your answer.

1) `cd ../../Project/Proj1`

   Answer: the new directory will be `ENEE140/Project/Proj1`.
   This information can be verified by command: `pwd`
   To get back, you can use `cd ../../Lab/Week01`

2) `cd ..`
3) `cd ~`
4) `cd ../Week02`
5) `cd ~/ENEE140`
6) `cd ../Project/Proj2`
7) `cd ../Week08`
8) `cd .`

5. UNIX files

   (a) We will work on two types of files in UNIX: plain-text, such as C programs edited in CLion or Eclipse, and executable files, which result from compiling a program.

(b) Executable files can be invoked from the command line to run the compiled program.

(c) Some useful UNIX file management commands:

i. `cp file1 file2`: make a copy of file1 and name it file2.
ii. `mv file1 file2`: rename file1 to file2.
iii. `rm file1`: delete file1
iv. `diff file1 file2`: compare and show the difference of the two files.
v. `wc file1`: count the number of lines, words, and characters in file1.
vi. `chmod`: change the read, write, and execute permission of a file.
vii. `more, less, head, tail`: view a (plain text) file or part of it.
viii. `grep key file1`: search text "key" in file1 and show the line(s) that contain "key"

8 Practice programming

1. Open your CLion IDE
2. Create a C project called `circle`
3. Create a C program called `circle.c` with the following content:

```c
/* This program asks user to enter the radius of a circle
         and then print out the area and circumference of the circle.
         It has the following features:
         1. variable declaration.
         2. constant variable declaration.
         3. read in input from screen (scanf() function).
         4. print out the value of variables by printf()  function.
         5. define all the variables in a function before any other statement.
         6. illustration of a few best programming practices:
             * tell the user what you are expecting
             * check whether you are reading in the right thing or not after
             * each scanf() by simply printing out the values

                         Gang Qu, January 27, 2010
                         Tudor Dumitras, January 24, 2014
 */
#include <stdio.h>
#define PI 3.14159

int
main()
{
    float r, A, C;
    printf("Enter the radius of the circle (positive number):");
    scanf("%f", &r);
    printf("Radius = %f \n", r);

    /*
        we should check whether r is positive or not,
        which we will learn pretty soon.
     */
    A = PI * r * r;
    C = 2 * r * PI;
    printf("Area = %f, Circumference = %f\n", A, C); return 0;
}
```

The purpose of this exercise is to help you practice using the IDE. Don't worry if you don't understand some of the statements in this C program; we will discuss them in detail in class. If you are impatient, you are also welcome to look them up in the textbook or to ask the TAs.

Question 4: Copy the `circle.c` program to GRACE, under the `ENEE/Lab/Week01/` directory, using WinSCP.

Homework

No homework due this week.

Reading assignment

K&R Introduction and Chapters 1.1, 1.2, 1.4.

Take a look at K&R Exercise 1.4. Can you solve it by applying what you have learned from the reading?