Lab 0: An Introduction to C

CSE/IT 113L

NMT Department of Computer Science and Engineering

"A common mistake that people make when trying to design something completely foolproof is to underestimate the ingenuity of complete fools."

— Douglas Adams

"The most exciting phrase to hear in science, the one that heralds new discoveries, is not 'Eureka!' but 'That's funny...'"

— Isaac Asimov

"If Java had true garbage collection, most programs would delete themselves upon execution."

— Robert Sewell

Contents

1 Introduction

In this lab you will be introduced to the C programming environment, buddy programming, and making your own program.

2 Buddy Programming

We will be using Buddy Programming for all of the labs in this class. Buddy programming differs from Pair Programming, what we do in the prelabs, in that there is no `driver`, `navigator`, or `keeper`. You will work on your own code and develop your own program. You do, however, have a buddy in the form of the person (or people if you were in a group of three) you were paired with in the prelab. You will need to sit together for this lab, just like the prelabs, and you are allowed to discuss problems you are having and possible solutions. However, you are not allowed to look at each others code until both of you are finished with your own code. At this point you can review each others code.

We would like to make it clear though that even though you have a buddy you should not be sharing code with them, either written on a paper, typed out in an email or other messaging service, or through smoke signals (weirder methods probably exist). We do run all the labs through a program that **will** inform us of any cheating. Yes, this has happened in the past. Be good, do your own work, and be a supportive buddy!

3 Requirements

3.1 Your First C Program

Make sure you are in your lab0 directory. Open Gedit or Atom from the terminal

```
$ gedit&
or
$ atom&
```

Gedit is a popular GUI based text editor. However if you are a Computer Science Major, it is recommended to look into the classic UNIX text editors Vim and/or Emacs. However, there is a learning curve to using either one.

Exercise 1 (hello.c, lab0.script).
In your favorite editor, type in the following "Hello world!" program.

```c
#include <stdio.h>

int main (void)
{
        printf ("Hello, world!\n");
        return 0;
}
```

Save the file (in your lab0 directory) as hello.c

Return to your terminal and compile and run your code.

Compile:

```
$ gcc -g -Wall hello.c -o hello
```

Run:

Since you have now saved your executable to a file named `'hello'`, you can run it using the following command.

```
$ ./hello
```

Make a script called `lab0.script` that shows `hello.c` compiling and running.

3.2 ASCII Art

This week, you will be creating something fun. ASCII Art!

Exercise 2 (name.c, name.script, ascii.c, ascii.script).
Open up Gedit or some other editor of your choice and save a blank document as name.c

Add the basics for a C file:

```c
#include <stdio.h>

int main(void)
{

```

```
  return 0;
}
```

Now you can use `printf` statements to make "art" using letters, numbers, and characters.

First, you are going to write your name in ascii art (this will go in name.c). Uppercase letters must minimally be 10 lines tall and lowercase characters must minimally be 5 lines tall. Fill the screen with as many characters of your name as possible. If you have a 2 or 3 letter name make the characters taller. Remember to put newline characters at the end of each line!

For example, here is the letter "K" in ascii art.

```
#include <stdio.h>

int main(void)
{
        printf("k        k\n");
        printf("k       k\n");
        printf("k      k\n");
        printf("k     k\n");
        printf("k    k\n");
        printf("k   k\n");
        printf("k  k\n");
        printf("k k\n");
        printf("kk\n");
        printf("k k\n");
        printf("k  k\n");
        printf("k   k\n");
        printf("k    k\n");
        printf("k     k\n");
        printf("k      k\n");
        printf("k       k\n");
        printf("k        k\n");

        return 0;
}
```

Next, make whatever you would like, name this file `ascii.c`. Your designs will be graded on effort so get creative!

If you need some inspiration see `http://asciiart.website`.

Also, feel free to make more than one design. You have no limitations for this assignment.

After you have completed your designs, make a script of your ASCII art. You will create two script files here, name.script and ascii.script.

```
$ script ascii.script
$ gcc -g -Wall ascii.c -o ascii
$ ./ascii
$ exit
```

3.3 Extra Credit

The CS Department has a server on Discord which is used by multiple classes for communication and dissemination of information seeing as a decent amount of students have a Discord account and probably an even higher amount do not pay attention to their Canvas notifications.

Exercise 3 (discord.EXT).
You can get **10% extra credit** for this lab by joining the department server and the 113 channels.

Use `https://discord.gg/BvDT4DC` to join the server and type `!cse113-s` or `!cse113-b` in the general chat to join the 113 channels.

Steps needed to get extra credit after joining the server:

1. Post a message in the `C Programming` general chat introducing yourself:
   - Name
   - Preferred Pronouns
   - If you had to eat one meal everyday for the rest of your life what would it be? Why?
2. Take a screenshot of your post, rename it `discord.EXT` where EXT is whatever the original file extension of your screenshot was.

Submitting

You should submit your code as a tarball file that contains all the exercise files for this lab. The submitted file should be named (**note the lowercase**)

`cse113_firstname_lastname_lab0.tar.gz`

Upload your `.tar.gz` **file to Canvas.**

List of Files to Submit

Exercises start on page 1.