CS 382E: Lab 1
Date: September 6, 2022

Philippos Mordohai
Department of Computer Science
Stevens Institute of Technology

Objective

The goal of this lab is to gently introduce the C programming language and command-line compilation tools.

1 Compile, Link, and Execute

Your C program should be stored as a source file, with extension `.c`. Open the terminal, and `cd` to the directory where the file with the source code is. The first step is to use `gcc` to compile your code:

```
gcc hello.c -c
```

This command tells `gcc` to compile our source code `hello.c`, and generate an object file `hello.o`.

The next step is to link all the object files. Since in this example we only have one object file, we can simply use

```
gcc hello.o
```

which will generate an executable file, whose name is `a.out` by default, if there's no any error in the code. If there are multiple object files generated from different source files, say `hello.o`, `test.o`, and `demo.o`, the command line would be

```
gcc hello.o test.o demo.o
```

Then we can run the executable:

```
./a.out
```

Note that `./` in the beginning is necessary.

If we want to generate an executable with another name instead of the default `a.out`, we need to use options. For example, to name our executable `cute`, we should do:

```
gcc hello.o -o cute
```

To run this, type the following in the terminal:

```
./cute
```

Note that the output file name (`cute` in this example) has to follow `-o`, but `-o output_name` can be anywhere after the command `gcc`. Keep in mind that an option and its argument always go together as a pair or tuple.

Usually the following steps can be simplified to just one command:

```
gcc hello.c
```

which will compile (without saving the object file), link, and generate an executable. For the rest of the course, feel free to use this command to speed up your workflow, but for this lab, you have to know how to use the options to do the compilation and linking separately. This will help you later when we write assembly programs.

2 Printing in C

Printing to the terminal in C is done using `printf()`. The syntax is as follows:

```
int printf(const char *format, var1, var2,...);

printf("The variables are \%c, \%d, \%f\n", var_0, var_1, var_2);
// each specifier (starting with \%) will be replaced by the
// corresponding variable.
// This will print "The variables are A, -1, 2.560000"
```

See https://www.programiz.com/c-programming/c-input-output for more details. In CS 382, the following format specifiers will be sufficient:

- `%d` integer
- `%f` float
- `%s` string (null terminated)

- `%c` single character
- `n` newline (do **not** use in `scanf()`)

3 Types

C, like Java, is a strongly typed language. Considering the types of variables and literals (constants) try to predict the output of the following program.

```c
#include <stdio.h>
main() {
   int  number = 1;
   char character = 'k'; /*ASCII value is 107 */
   int sum;
   sum = number + character;
   printf("Value of sum : %d\n", sum );
}
```

Now, try the following examples, keeping operator precedence in mind. Make sure that the correct format specifier is used to print the results of each operation. Can you come up with other tricky cases?

```c
int x = 5/3;
float x = 5/3;
int x = 5.0/3;
float x = 5/3.0;

float x = 4+5/2;

float x = 5*1.0+2;

float x = 1/3+1/3+1/3;
float x = (1+1+1)/3;
```