Name:…………………… Q9

CS2123 Data Structures

Suppose you want to implement a simple library called **numbers** that will export a lot of functions to find out about numbers (e.g., positive, negative, even, odd etc.)

For the time being, suppose your library exports only two functions `int even(int x)` and `int positive(int x)`. These functions will return 1 if the given number is even and positive, respectively; otherwise return 0. Define an interface named **`numbers.h`** exporting the above two functions (give the boilerplate lines (`#ifndef, #define, #endif`)) and implement **numbers.c**

```
/****** numbers.h ******/
```

What is the purpose of the boilerplate lines in the above file?

```
/****** numbers.c ******/
```

Now suppose you want to implement an application that asks user to first enter a number. It then prints " this is a positive/negative even/odd number" on the screen. Implement **myapp.c** using the functions from the above library.

```
/****** myapp.c ******/
```

```
/****** myapp.c cont'd ******/
```

Suppose now you have `numbers.h`, `numbers.c`, and `myapp.c`. What commands will you use to compile your application as well as the above library on Linux.

```
Linux terminal

hostname:~>
```

Clearly you will add a lot of functions to your library and application in the future. Since you need to compile them again and again, you realize that it will be better to use Makefile and make. Give a simple Makefile for compiling your library and your application.

```
# Makefile
```