CS 271 and 462

PA 3 - Programming Assignment 3

See the link "Creating a Makefile" in the Resources Module.

Reminder: The gcc flag for compile only (and do not create an executable) is -c. Don't use -c and -o together in the same gcc command.

Overview of This Assignment

4 Files:

- 2 C source files
- 1 header file
- 1 makefile

You must upload the files to a CS Linux host, make, then run the executable on Linux.

Once you have finished and tested the programs, you will upload them to the PA 3 assignment in Canvas.

C Programming Skills That You Are Practicing in This Assignment

1. Writing functions in C.
2. Writing and using a makefile to expedite the compiling and linking phases of development.
3. Improving your documentation and style skills.
4. Improving the readability of your programs.
5. Increasing your debugging skills to include syntax errors in functions and function calls.

Reminder: If you submit a program that contains a syntax error, you will receive a grade of zero for that program.

Grading Rubric

The rubric for this assignment has 25% of the assignment score allocated to documentation, style, and readability. Make sure you read the rubric before you submit.

As you go...Backup Your Work

Each time you work on an assignment, make a backup of your files. Some techniques for backup:

a. copy the files onto a USB drive
b. email the files to yourself
c. upload the files to cloud storage (OneDrive, GoogleDocs, Dropbox, etc)

Warning: If you have Linux installed on your own computer and you have the GNU gcc compiler, you may try using your own computer but you should still test your files on a CS host computer before submitting.

1. Make a new directory for this lab. Place all of the files you create for PA 3 in that directory.

2. Create a makefile. There is a separate instruction sheet for this.
   - The "all" target must produce the executable **pa3**, all lowercase, no extension.
   - Required targets: all, pa3, pa3.o, pa3functions.o
   - The clean target is not required.

3. Create a header file named pa3functions.h. In this file, place the following lines:

```
#ifndef PA3FUNCTIONS
#define PA3FUNCTIONS
        here is where your function prototypes go
#endif
```

4. Create a source file named pa3functions.c In this file, place the implementations of the following three functions:
   - problem 5.26. A function named perfect that will accept an integer parameter. The function should return 1 if the parameter is a perfect number. Otherwise, it should return 0.

     An integer is said to be a perfect number if its factors, including 1 (but not the number itself), sum to the number. For example, 6 is a perfect number because 6 = 1 + 2 + 3.

   - problem 5.27. A function named prime that will accept an integer parameter. The function should return 1 if the number is prime. Otherwise, it should return 0. (See the pseudocode below)

```
function prime ( int n )
  loop from x = 2 to x = sqrt(n)
    if n is divisible by x, return 0
  end loop
  return 1
end function
```

- problem 5.28. A function named revDigits that accepts an integer parameter. The function should return the number with its digits reversed.

  Example: revDigits(7631) should return 1367

  Example: revDigits(1840) should return 481

  Example: revDigits(-945) should return -549

▸ ▸ Remember to put #include "pa3functions.h" at the top of the file.

5. Create a source file named pa3.c. In this file, place the main function as follows:

```
// header comments
#include "pa3functions.h"
#include <stdio.h>
int main (void) {
   // use the perfect function to print the positive, perfect numbers
   // less than 1000

   // use the prime function to print the first 20 positive, prime numbers
   // note:  1 is not a prime

   // repeat the following process 5 times
        // use the rand function to generate a random number between
        // 1 and 10000 (inclusive)
        // use the revDigits function to obtain the reverse of the number
        // print the reverse of the number
}
```

6. "make" pa3. If the makefile works correctly, run the executable.

   Submit 4 files:

   1) makefile
   2) pa3functions.h
   3) pa3functions.c
   4) pa3.c

Make sure that you select the correct files and that all programs are uploaded before you submit.

From the syllabus: "It is your responsibility to upload the correct files and to ensure that the uploaded files contain the work that you want to have graded."