CMSC 216 Quiz 1 Worksheet

The first quiz for the course will be on Fri, Jun 7. The following list provides additional information about the quiz:

- The quiz will be a written quiz (no computer).
- Closed book, closed notes quiz.
- Answers must be neat and legible.
- Quiz instructions can be found at http://www.cs.umd.edu/~nelson/classes/utilities/examRules.html
- Make sure you know your section number and your TA's name.

The following exercises cover the material to be included in this quiz. Solutions to these exercises will not be provided, but you are welcome to discuss your solutions with the TA or instructor during office hours. If is recommended that you try this exercises on paper first (without using the computer). Qu

Exercises

1. Read the document available at:

   http://www.cs.umd.edu/~nelson/documents/SuggestionsForWritingComputerPrograms.htm

2. Name at least one difference between a #include and an import statement in Java.

3. Name and briefly explain the compilation stages associated with a C program.

4. What is the difference between static storage and automatic storage?

5. Write a Unix command that will copy all C files present in the directory /tmp to your home directory, assuming your current directory can be any directory.

6. Named two flags we use with gcc for compilation.

7. Write the Unix command that will allow us to compile the program myProg.c, and the leave the executable in a file named myProg.x.

8. Write a complete C program that reads two integer values and prints the powers of two of values in the specified range. You can assume the first value is less than or equal to the second. For example, if the user enters 3 and 4 we expect to see 8 16.

9. Write a C function that determines whether a positive sequence of integer values provided by the user represent an increasing sequence. For example, 3, 6, 10 represents an increasing sequence. The function will return true if the sequence is increasing and false otherwise. You can assume a negative value will mark the end of the sequence. You may not use arrays for this problem and your function must work for any number of values (not just 3).

10. Write a C function that reads two positive floating point values from the user. The function must keep asking the user for values as long as invalid values (e.g., negative or character data) is provided by the user. The function will return the number of attempts needed to read the values. Use the message "Invalid data" to report any data problems.

11. Write a recursive C function that computes the n^{th} number in the Fibonacci sequence.

12. Expand the previous C function so you can print the number of times the function was called.