This homework is very substantially shorter and simpler than those that will follow. Its main purpose is to ensure that everyone is familiar and comfortable with the packaging and submission requirements. Responses to this homework are by electronic submission only. All **electronic submissions** must be made using the Web interface at `http://chaw.eip10.org/u/` only. Do not use email or any other means for submission. If your submission is successful, you will be presented with a Web page indicating so, along with a timestamp and MD5 checksum; you should save that information for your records as a backup. You should submit a single file named using the template

`cos451-hw01-`*`lastname`*`-`*`firstname`*`-`*`pqrs`*`.tgz`

where *`lastname`* and *`firstname`* are replaced by the obvious and *`pqrs`* is replaced by a 4-digit number of your choosing. (Please check the details carefully; attention to details is important in this course in particular, and Computer Science in general.) You should design your submission so that the command

`tar zxf cos451-hw01-`*`lastname`*`-`*`firstname`*`-`*`pqrs`*`.tgz`

(with the same replacements as above) results in the *creation of a directory*

`cos451-hw01-`*`lastname`*`-`*`firstname`*`-`*`pqrs`*

(again with the same replacements) as a subdirectory of the working directory (i.e., informally within the directory in which the *tar* command is executed).

In the above directory (created by the earlier *tar* command with your submitted package) should be all the **source** code for your submission (organized in further sub-directories if you like) as well as a **README** file and a **Makefile** with the usual semantics. Do not submit any kind of non-source files (executables, object files, class files, etc.). Be especially careful if you use IDEs such as Eclipse for your work because the packages created by them often contain compiled, not source files.

Ask for clarifications in class or on the newsgroup if you have any doubts regarding the submission format and procedure. (It's an essential part of the homework.)

The README file should be a *plain text* file (not PDF, .docx, etc.) that includes, at a minimum, the following:

- Class information (University of Maine, COS 451, Spring 2022).
- Author information (your name)
- Primary (`@maine.edu`) email address.
- Date in an unambiguous format
- A brief summary of the contents of the submitted package (file-wise).
- A brief description of what the submitted code does.
- Instructions for compiling the code. (Ideally, just typing 'make' should work, but any special requirements or wrinkles should be noted here.)
- Instructions for running and testing the code.
- Known bugs, limitations, etc.
- Any other information that will help someone understand the submitted material.

The *answers to the non-programming questions* should all be in a single PDF file named `ans.pdf`. Please answer the questions in numerical/listed order in this file. The file may be

produced either purely electronically or by first writing the answers by hand on paper and then scanning the resulting pages into PDF. In either case, everything should be very easy to read and comprehend.

For *each programming question*, the *make* command noted above should result (with just one invocation of the command total, not one per question) in the creation of an executable file `qmn` where *mn* is replaced by the appropriate 0-padded question number. (For instance, for question 5, the filename would be `q05`.) Running that executable program (`q05` in the example) should result in the required input-output behavior. Unless stated otherwise, all such programs should read from *standard input* and write to *standard output* (and optionally *standard error*). Please be sure to understand what these terms mean (they do not mean "terminal") and to ensure that your programs do not make additional assumptions (such as interactive input/ouput at a terminal).

You are welcome to use any inanimate resources (e.g., books, Web sites, publicly available code) to help you with your work. However, **all such help must be clearly noted** in your submissions. Further, no matter what you use, *you must be able to explain, in detail, how it works*. (You may be called upon to explain your homework in person.) Refer to the class policy for details, and ask for clarifications if you are unsure if something is allowed.

1. (35 pts.) This question does not need an explicit answer; it is a placeholder for points allotted to proper submission and packaging as described above.

2. (5 pts.) There is at least one error in this homework; find it and identify it in your submission.

3. (15 pts.) Write a program that ignores its standard input and writes exactly two lines (terminated by the newline character) to its standard output. The first line is "Hello, World!" and the second is "Goodbye, cruel world!"). It should also write two lines to standard error. The first is "All is..." and the second is "well!"

4. (15 pts.) Given the sets $A = \{1, 2, 42\}$, $B = \{24, 7, 1, 2\}$, provide a brief description (prose) *and* list the elements (explicitly) of:

   (a) $A \times B$.

   (b) $\{a \times b \mid a \in A,\ b \in B\}$.

   (c) $\mathcal{P}(A \setminus B) \times B$.

5. (15 pts.) Provide formal definitions of the following graphs (with n being a parameter for each definition):

   (a) An *n-cycle*, i.e., a graph with n vertices and edges forming a single cycle that connects all vertices.

   (b) A *complete* graph with n vertices: there is an edge for every (unordered) pair of vertices.

   (c) An *n-star*: edges connecting a single vertex (the *center*) with every other vertex.

6. (15 pts.) Write a program that reads its standard input (fully, until end of input) and writes exactly two lines (terminated by the newline character) to its standard output. The first line is "Hello, x" with x replaced by the contents of the first line of the standard input. The second line of the output is "Goodbye, y"), with y replaced by the contents of the last line of the input. In case the input has less than one line, the string "stranger" should replace both x and y. This program should write any reasonable tracing/debugging information to standard error. (Writing nothing to standard error is a valid choice.)