Grading Scheme for Programming Assignment 1

Instructor: Maxwell Young

Out of 100 points total.

Some important points:

— If the code fails during execution, please try 2 more times with random ports (remember to change in both client and server call) as unclosed ports (from a previous run using another student's code) have caused problems in the past.

— I've included the standard makefile. If the students have included their own, go ahead and use that. Otherwise, use the one I provided (either Java or C++).

— I've provided a file.txt file for testing. This is almost identical to the one the students used to debug their code, but slightly changed to make sure nobody just hardcoded the output.

Part 1: Following submission instructions [8 points total]

1. Single tar or zip file submitted to MyCourses -- **2 points**

2. Used the correct names: -- **4 points (all or nothing)**

`client.cpp` and `server.cpp`
or
`client.java` and `server.java`

3. No .o files or other stuff submitted -- **2 points**

Part 2: Compilation and Execution [92 points total]

I've attached a sample execution, please take a look.

4. Typing "make" compiles and produces executables on Pluto -- **25 points**

5. Code executes without crashing -- **25 points**

If the code crashes/segmentation faults (**hanging is okay**), then the student gets 0 on this portion. The student may still get points going forward.

If this is C++, then test:

```
./server 6008
./client localhost 6008 file.txt
```

In the case of Java, test:

```
java server 6008
java client localhost 6008 file.txt
```

For java code, if you cannot run it on Pluto due to an memory/thread error, please run it on your machine locally. Pluto's resources are capped and some students ran into this error, but it doesn't mean their code is wrong. Flag any outstanding cases and let me know.

6. Correct output to screens in the following sense -- **20 points**

6i - First, random port must be printed to screen as "Random port chosen: <x>" where <x> is the port value. (10 points).

6ii - Second, the ACK messages must be correct and printed on the client side (see attached output for this part) (10 points)

7. Client and server both terminate after execution; that is, neither hangs -- **5 points, all or nothing**.

8. The `upload.txt` file has correct text according to the following criteria -- **17 points**

8i - If the file has the correct content, award full 17 points.
8ii - Else, if the file has errors but has at least ***some*** of the correct text (that is, it's not all gibberish), award 14 points.
8iii - Else, if the file is ***all*** gibberish, but the file exists (that is, the file was successfully created) award 10 points

For the above, subtract 5 points if `upload.txt` is not used as file name.

If the file wasn't even created, award 0 points.

8iv - If students have hardcoded a file transfer then they receive 0 points on this portion of the grade and subtract another 20 points. Students should not be doing this!