Name: ________________________________________

**Shell Script - Points: 56 (10% of 30%)**

Write code for an executable shell script that will do the following actions, in the exact order given below. (Approximately 40 lines of executable code.)

Summary and Purpose:

> The script expects a single optional C++ program source file name on the command line. It will compare the number of lines in this source file with the number of lines in another file, and it will compile whichever program has more lines. It backs up the output file first.

Comments Required:

> Write brief block comments containing the **step number** in front of each step.

---

1. *[Points: 4]* Start your script with all the parts of a correct DAT2330 script header; however, you do *not* need to write down the Purpose or Assignment Label.
2. *[Points: 7]* Issue a good error message (follow the DAT2330 guidelines) and exit the script with status 1 if there is more than one command line argument.
3. *[Points: 5]* If there is a program name on the command line, put that program name into a variable named **`prog`**; otherwise, the script must get the program source file name from the user. Put the program name entered by the user into variable **`prog`**. Quick-exit the script with status 2 if the user signals EOF to the script. (No error message is needed.)
4. *[Points: 6]* Make sure the pathname in variable **`prog`** is not an empty string; otherwise, issue a good error message and exit the script with status 3.
5. *[Points: 6]* Make sure the pathname in variable **`prog`** is a plain file; otherwise, issue a good error message and exit the script with status 4.
6. *[Points: 6]* Make sure the file whose name is in variable **`prog`** is not an empty file (has a size larger than zero); otherwise, issue a good error message and exit the script with status 5.
7. *[Points: 6]* If the file whose name is in variable **`prog`** is not readable, add read permissions for user and group (not for others). Quick-exit the script with status 6 if changing permissions fails. (No output or error messages are needed.)
8. *[Points: 3]* Put only the count of lines from the file named by the **`prog`** variable into a variable named **`plines`**.
9. *[Points: 3]* Put only the count of lines from the file named **`foo.cpp`** into a variable named **`flines`**. (You may assume this file exists; no code is needed.)
10. *[Points: 4]* If something with the name of **`happy`** already exists in the current directory, rename it to be **`happy.bak`**.
11. *[Points: 6]* Compile whichever of the two files that has the most lines into output file **`happy`**. Quick-exit the script with status 8 if the compile fails.

− FIN −