Lab 1

Goals-

write, compile, and test a program that uses multiple files; upload it to flip for testing; submit it to TEACH
create and use simple makefiles and simple archiving of all files in a zip file
compile a program written in multiple source code files

You will create a program that uses two classes. Each class will be in a separate source file with an appropriate header file. The first class is Die. It will have a data member to indicate the number of sides. It will have a member function that returns the value of a single roll of a single die. The second class will be LoadedDie. It will also have a single data member that indicates the number of sides. It will have a member function that returns the value of a single roll of a single die. In this case it should return values that are higher than normal. You can't just add one to the roll. Getting a 7 on a 6-sided die stands out!

HINT: This is a programming assignment. You do not want to make the shift too subtle as that can make testing difficult.

In a fifth file you will create a program that rolls each die N times. You should print the result of each roll to the screen as well as the total of the N rolls of each die. Is your loaded die returning higher values on average?

Once your program is working and you have eliminated the bugs create a makefile to build your program. You do not need a complicated makefile. You can have a single target that just includes what you typed into the command line for testing.

Include a second target *clean* that removes extraneous files such as .o files or a.out.

Finally, you should include all 6 files into a zip archive. You must submit only the zip file in TEACH by the due date. We will only accept submissions in TEACH that are in a zip file and that include the makefile.

Modular Grading

We are using modular grading. Each lab will be divided into specific modules. Each module will be graded pass/fail. It either works properly or it does not. 10% of every lab or assignment grade is style/comments or other elements of self-documenting code and clarity. Specifically, can someone else understand what you intended for each part of your program? Each module or part of the lab will be clearly indicated and its point value. Remember the labs are worth 10 points.

Programming style- 1 point

Create the source and header file for the Die class- 2 points

Create the source and header file for the LoadedDie class- 2 points

Create and execute a program using both classes, and verify the results- 2 points

Create a makefile with targets for your program and clean - 2 points

Submit all files in zip archive- 1 point