COP 3014: Spring 2022
Homework 2

Total Points: 100
Due: Monday, 02/14/2022 11:59:00 PM EST through Canvas

1 Objective

The purpose of this assignment is to

- Practice using CLion to create, edit, compile, debug, and run a c++ program.
- Use the `iostream` library to perform console input and output.
- Use variables and arithmetic operators in C++ to do basic math.
- Familiarize yourself with selection statements (if . . . else and switch).
- Use fundamental algorithmic error checking to check the validity of user input.
- Familiarize yourself with simple looping constructs.

This assignment requires you to submit 2 files on Canvas. Please do so in a SINGLE submission. Canvas allows you to turn in multiple flies in one submission. Once you have uploaded your first file, click "Attach Another File" to upload your second file.

Turn in your 2 files `cakes.cpp` and `sharks.cpp` to Canvas.

2 Cake Slicing

Patrick is very impressed by the success of Spongebob's catering venture. He has convinced Spongebob to start selling cakes alongside the Pretty Patties. The cakes come in 3 sizes - wubmo (small), mumbo (medium) and jumbo (large). The Pretty Patty Store only sells completely spherical cakes. He will get the radii of the 3 cake sizes sold, not necessarily in order, and the number of slices in each size, and rank them in ascending order of volume per slice. While Patrick is good at taking advantage of situations ad-hoc, he is not that great at math or remembering things. So, he enlists your help.

Write a C++ program to get cake radii and number of slices per cake for 3 sizes of cake as input and rank the cakes in decreasing order of volume per slice.

Call this program `cakes.cpp`

Specifications

1. Prompt the user and read in 3 cake radii. Store them in 3 variables. These need not be in the wumbo - mumbo - jumbo order. (5 points)
2. Prompt the user and read in the number of slices for each of the 3 cake sizes entered. The number of slices will correspond to the cake size, ie, the first number of slices is for the first cake size. (5 points)

3. If the user entered values become the same 3 volumes per slice, print an error message. (10 points)
4. Use `if ...else` statements, in whatever combination required, to print the cake numbers and their volumes per slice out in decreasing order. (25 points)
5. Add comments wherever appropriate to explain your logic. (5 points)
6. You can assume that the user will only enter the proper type of data for the values. Positive floating-point numbers for the radii and positive integers for the number of slices. No characters or text.
7. You may use 3.14 as the value for Π
8. The formula for the volume of a sphere is $\frac{4}{3}\Pi radius^3$
9. You may assume that all slices of a cake will be automatically of equal volume.
10. Print the values accurate to 3 digits after the decimal point.

Sample Run 1

```
Enter the 3 radii values: 3.1
4.5
1.2
Enter the number of slices for each: 2
6
1
cake 2: 63.585, cake1: 62.362, cake3: 7.235
```

Sample Run 2

Enter the 3 radii values: 16.8 1.42 91.34 Enter the number of slices for each: 45 1 12 cake 3: 265870.505, cake1: 441.147, cake2: 11.988

Sample Run 3

```
Enter the 3 radii values: 30
20
10
Enter the number of slices for each: 27
8
1
Error. You entered the same volume per slice 3 times.
```

Problem 2 - Estimating the Damage of a Sharknado

Sharknados can be very destructive. As a research grunt for a popular internet celebrity that tries to estimate the theoretical amount of damage caused by movies, you have been asked to assess the damage caused by different Sharknados. Your boss has determined that the damage caused by a Sharknado is calculated by using the formula:

```
damage = windspeed * sharkWeight * duration
```

The user will enter the windspeed as a number, in miles per hour; the duration of the tornado, in seconds, and the type of shark as a character. The formula requires the windspeed in meters per second. You will have to convert the windspeed into meters per second. 1 mile per hour is 0.447 meters per second. The following table will help you match the shark (and hence its average weight) to the user input character, which we will call the reference letter.

Call this file `sharks.cpp`

Shark	**Weight (kg)**	**Reference Letter**
Basking Shark	14500	B
Great White Shark	2270	G
Hammerhead Shark	230	H
Whale Shark	21000	W
Bull shark	130	U
Lemon Shark	183	L

1. Print a menu for the user. In the menu, display the sharks and their reference letters. (6 points).
2. Prompt the user the enter the shark in question. Read in the reference letter. (3 points).
3. Prompt the user to enter the windspeed (in miles per hour) and read in the value (3 points).
4. If the speed entered were negative, loop until the user enters a positive value or 0. (4 points)
5. Prompt the user to enter the duration of the tornado (in seconds) and read in the value (3 points).
6. If the duration entered were negative or 0, loop until the use enters a positive value. (4 points)
7. Use a `switch` statement on the reference letter to figure out the weight. (15 points).
8. Calculate the damage done by the Sharknado and print it, accurate to 2 digits after the decimal point. (4 points).
9. If the reference letter entered doesn't match any shark, print an error message (3 points)
10. Add comments wherever appropriate to explain your logic. (5 points)
11. You can assume that the user will only enter the proper type of data, an uppercase character for the reference letter and a floating point number for the others.

Sample Runs

The underlined text is the user input.

Sample Run 1

```
Basking Shark - B
Great White Shark - G
Hammerhead Shark - H
Whale Shark - W
Bull Shark - U
Lemon Shark - L
Enter the Shark:  G
Enter the windspeed (in miles per hour):  -15.7
Windspeed should be 0 o positive.  Enter the windspeed (in miles per hour):  -91
Windspeed should be 0 o positive.  Enter the windspeed (in miles per hour):  153.2
Enter the duration (in seconds):  56
Estimated Sharknado Damage:  $ 8705228.45
```

Sample Run 2

```
Basking Shark - B
Great White Shark - G
Hammerhead Shark - H
Whale Shark - W
Bull Shark - U
Lemon Shark - L
Enter the Shark:  X
Enter the windspeed (in miles per hour):  119.76
Enter the duration (in seconds):  0
Duration should be positive.  Enter the duration (in seconds):  -23.5
Duration should be positive.  Enter the duration (in seconds):  102
Error:  Invalid Reference Letter for Shark
```

3 General Requirements

1. Include the header comment with your name and other information on the top of your files.
2. Please make sure you're only using the concepts already discussed in class. Please restrict yourself to variables, operators,selection statements and loops. Using functions, arrays, C++ 11 and up features or anything more advanced will result in a loss of 10 points.
3. Each program is worth 50 points.
4. If we have listed a specification and allocated point for it, you will lose points if that particular item is missing from your code, even if it is trivial.
5. **This is individual work. You may NOT collaborate with other students in the course, former students, hire tutors to "help", copy solutions off the internet, or use pay-for solution websites, including but not limited to Chegg, CourseHero, WiseAnt, Bartelby, tutor.com, assorted Social Media groups, whatever GroupMe students might have created, etc.) This includes posting the problem statements on these websites even if you do not use the answer obtained. Doing so is a violation of the Academic Honor Code. Violation of the Honor Code will result in a 0 grade on the program, a reduced letter grade in the course and potentially more serious consequences.**
6. No global variables (variables outside of main())
7. No `goto` statements or the `auto` keyword.
8. All input and output must be done with streams, using the library `iostream`
9. You may only use the `iostream` and `iomanip` libraries (you do not need any others for these tasks)
10. NO C style printing is permitted. (Aka, don't use printf). Use cout if you need to print to the screen.
11. When you write source code, it should be readable and well-documented (comments).

12. Make sure you either develop with or test with JetBrains CLion (to be sure it reports no compile errors or warnings) before you submit the program.

13. Testing your program thoroughly is a part of writing good code. We give you sample runs to make sure you match our output requirements and to get a general idea of how we would test your code. Matching your outputs for JUST the sample runs is not a guarantee of a 100. We have several extensive test cases.

14. Please make sure you've compiled and run your program before you turn it in. Compilation errors can be quite costly. We take 5 points off per compiler error for the first 9 errors. The 10th compiler error will result in a grade of 0.

15. Only a file turned in through Canvas counts as a submission. A file on your computer, even if it hasn't been edited after the deadline, does not count.

16. The student is responsible for making sure they have turned in the right file(s). We will not accept any excuses about inadvertently modifying or deleting files, or turning in the wrong files.

17. **Program submissions** should be done through the Canvas class page, under the assignments tab (if it's not there yet I'll create it soon.) Do not send program submissions through e-mail – e-mail attachments will not be accepted as valid submissions.

18. The ONLY files you will submit via Canvas are `cakes.cpp` and `sharks.cpp`

19. **General Advice** - always keep an untouched copy of your finished homework files in your email. These files will have a time-stamp which will show when they were last worked on and will serve as a backup in case you ever have legitimate problems with submitting files through Canvas. Do this for ALL programs.