24-780 Engineering Computation Problem Set 07

You need to create a ZIP file (It may appear as a compressed folder in Windows) and submit the ZIP file via the 24-780 Canvas course. The file name of the ZIP file must be:

PS07-YourAndrewID.zip

For example, if your Andrew account is *hummingbird@andrew.cmu.edu*, the file name must be:

PS07-hummingbird.zip

If your ZIP file does not comply with this naming rule, you will automatically lose 5% credit from this assignment. If we are not able to identify who submitted the file, you will lose another 5% credit. If we finally are not able to connect you and the submitted ZIP file, you will receive 0 point for this assignment. Therefore, please make sure you strictly adhere to this naming rule before submitting a file.

The ZIP file needs to be submitted to the 24-780 Canvas course. If you find a mistake in the previous submission, you can re-submit the ZIP file with no penalty as long as it is before the submission deadline.

Notice that the grade will be given to the final submission only. If you submit multiple files, the earlier version will be discarded. Therefore, if you re-submit a ZIP file, the ZIP file MUST include all the required files. Also, if your final version is submitted after the submission deadline, late-submission policy will be applied no matter how early your earlier version was submitted.

Make sure you upload your Zip file to the correct location. If you did not upload your assignment to the correct location, you will lose 5%.

The ZIP file needs to include:

- C++ source file of your program (ps7.cpp)

Submission Due: Please see Canvas

START EARLY!

Unless you are already a good programmer, there is no way to finish the assignment overnight.

PS7 Rewrite PS4-2 solution using a C++ class (ps7.cpp) [100 points]

1. **Download ProblemSet4-2 solution from the course Canvas**
2. **Rename ps4-2-solution.cpp to ps7.cpp**
3. **Add four classes, Artillery, CannonBall, Obstacle, and Target**
4. **Then, re-write the rest of the code accordingly.**

For the full credit,

a. **Artillery class has member variables that fully describe the state of the artillery**
b. **CannonBall class has member variables that fully describe the state of the cannon ball**
c. **Obstacle class has member variables that fully describe the state of the obstacle**
d. **Target class has member variables that fully describe the state of the target**
e. **CheckHitTarget function needs to be replaced with Target::CheckHitByCannonBall(double bx,double by) where mx and my are the cannon-ball coordinate. (Let's make Target and Obstacle classes be responsible for collision check this time)**
f. **Add Obstacle::CheckHitByCannonBall(double bx,double by) where mx and my are the cannon-ball coordinate, and use it in the main function.**
g. **DrawArtillery function needs to be replaced with Artillery::Draw**
h. **DrawCannonBall function needs to be replaced with CannonBall::Draw**
i. **MoveCannon function needs to be replaced with CannonBall::Move**
j. **FireCannon function needs to be replaced with CannonBall::Fire**
k. **GenerateObstacle needs to be replaced with Obstacle::Generate (* In ps4-2-solution.cpp, GenerateObstacle function generates all obstacles. Obstacle::Generate needs to be for one obstacle, and main function should run a for-loop to generate all the obstacles.)**
l. **DrawObstacle needs to be replaced with Obstacle::Draw (* In ps4-2-solution.cpp, DrawObstacle function draws all obstacles. Obstacle::Draw needs to draw only one obstacle, and the main function should run a for-loop to draw all the obstacles.)**
m. **DrawTarget needs to be replaced with Target::Draw**
n. **MoveTarget needs to be replaced with Target::Move**
o. **Add a member function Target::Initialize, and let the main function call this function for initializing the object instead of main function directly setting the coordinate and size of the target. (Initialize member functions of other classes are optional for this assignment).**

Extra 3 points: Allow the user shoot up to five cannon balls simultaneously. Trajectories must be drawn for all balls in the air.

Extra 3 points (Requires previous 3 extra points): Add a class called Explosion and modify the program so that the explosion visual effect is visible when the ball hits an obstacle, not just the target.

Multiple explosions can be rendered simultaneously for this point. You don't get this points if it the motion of the target and/or the artillery stop while the explosion is visible.

Good luck!

Make sure your program can be compiled with no error in one of the compiler servers. Don't wait until the last minute. Compiler servers may get very busy minutes before the submission deadline!