Assignment 1: "3D Gasket" & "Snowy World"
CSI4105: Computer Graphics
Assigned: March 21th(Tuesday), 2017
Due: March 31st (Friday), 2017 (by 11:55PM uploaded to YSCEC)

Purpose of this Assignment

- To familiarize yourself with C programming
- To familiarize yourself with OpenGL/GLUT
- To familiarize yourself with GLUT callbacks (in particular keyboard/display/idle)
- To familiarize yourself with running animation on OpenGL.

TASKS
You are to make an OpenGL/GLUT application in C/C++ that does the following:

(A) "3D Gasket"
1) Modify the 3D Gasket program covered in the class in a way that the tetrahedron is subdivided by volumes, not by surfaces. (Lecture 2, page 72)

(B) "Flying Objects"

1) Create ***N*** snowflakes (draw circles for snow) and animate it so that it looks like snow falling from the top of the screen to the bottom. Randomize the starting position and the size of each snow flake to make it look more interesting. You should also simulate the wind blowing as described below.

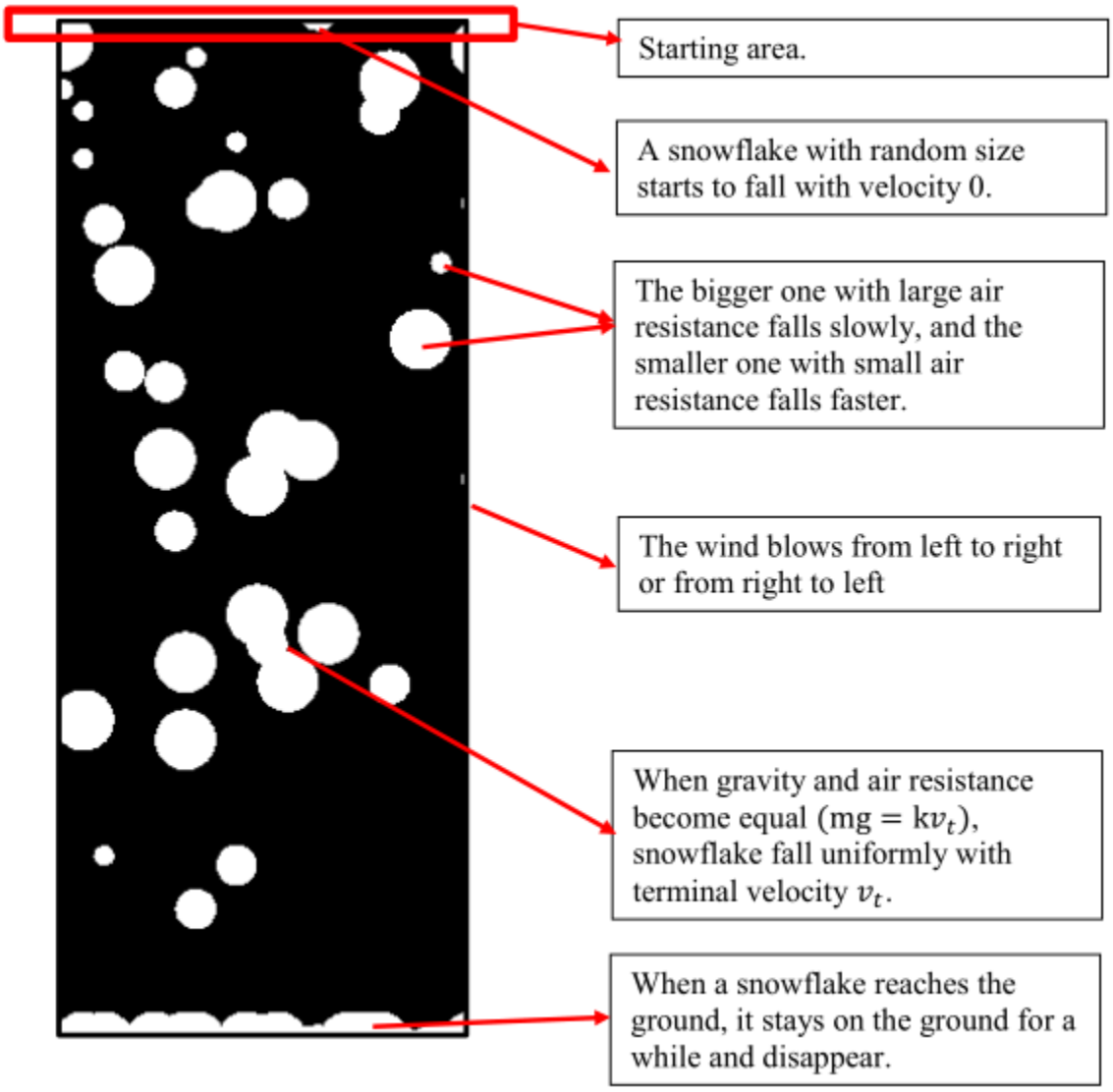
2) To simulate the snow fall, you can use the following knowledge from physics (Note that we are not simulating the snow fall which exactly follows the rule of physics as this is not a physics class).

Free falling snowflakes are affected by two forces, gravity toward the ground and the air resistance to the opposite direction. The gravity equation is $f_g = mg$, where m is the mass and g is the acceleration of the gravity, and the equation for the air resistance is $f_r = kv$, where k is the air resistance coefficient and v is the velocity. We can then calculate the acceleration of falling snowflakes as " $a = g - \frac{k}{m}v$ ".
However, for the snowflake with various size, we cannot figure out the exact value of mass m and the air resistance coefficient k. So, to simulate the snowfall, you have to decide the relationship between the size of snowflake and $\frac{k}{m}$. One option would be to set $\frac{k}{m}$ proportional to the square of the radius.

To simulate the wind blowing, you can use the same rule as explained above and setting the wind as the gravity applied to the horizontal direction.

3) Keyboard control
Press Escape key to exit the application
Press Space key to stop/resume motion
Press 'a' key to have the wind blow from right to left.
Press 'd' key to have the wind blow from left to right.

See following diagram for pictorial description:



Hints

- Timer is not necessary for this particular project.
- It may be helpful to construct a "structure" that contains object's current position and directional velocity
- Use the same "reshape" callback function as Box1.cpp
- To draw a circle, draw a polygon with infinity number of sides (or large enough number of sides to make it look like a circle). Can be done easily with trigonometrical functions (sine, cosine).
- To move snowflakes, first, calculate acceleration, then calculate the velocity, and finally update the next position to move to.

☼ **Rubric**

To get an "average" grade, the minimum you are required to do is to accomplish tasks (A), (B) 1-3. If you cannot do this, your grade will be below average.

- +30 pts: Implementation of the tetrahedron subdivided by volumes in Task (A)
- +30 pts: Implementation of various sized snowflakes falling down from the top to bottom starting in randomized positions in Task (B)
- +10 pts: Implementation of plausible physics applied to speed of snowflakes depending on their various sizes, in Task (B)
- +10 pts: Implementation of simulating the wind blowing in Task (B)
- +10 pts: Implementation of keyboard control in Task (B)
- -5*n pts: 5 points are lost every time the instructions for the submission format is not followed (More importantly, filename convention is worth 10 points!)

☼ **Extra Credit (up to 10 pts)**

To get an "above average" grade, you need to perform additional interesting "Bells and Whistles" beyond the basic requirement. Suggestions include:

- up to 2 pts: Try different shapes/colors of the snowflake
- up to 1 pts: Try timer
- up to 1 pts: Try more wind direction
- up to 3 pts: Snow piling up
- and so on… Be creative!

Assessment

- You will be assessed individually in the lab sessions
- You'll only have approximately 4-5 minute Q&A session with either the Professor or the TA's.
- Your uploaded file will be downloaded directly from YSCEC and compiled in the lab
  - You cannot use a different C file
  - Code must compile and run.
  - You can be asked anything about your code (every single line)
  - Be prepared to answer the questions, I forgot isn't a good answer!

Submission (READ CAREFULLY)
Please follow the instructions on the filename convention, it is important!

You must implement your assignment in a **single** C/C++ file. Even if you are an expert in C and know how to split the program into multiple headers/source files, it is required that the entire program is within a **single** C file; this is a hard constraint, no exceptions! This is necessary to expedite compiling for our in-class assessment. This rule will apply to all assignments. *Also, it is very important that you use the proper filename convention.*

NAMING CONVENTION
File name:
{ID}_{surname}_{firstnames}_A1.cpp
Example
2015147000_Kim_SeonJoo_A1.cpp
Please upload your file to YSCEC.
A letter grade lower every day turned in late. Marks deducted for incorrect filename.