Dungeon 3: The Party Planner's request (Fluid Simulation, **33g of total 100 g**). 1g may be exchanged for a human percentage point in COMP5823M.

As part of the celebration of Lord K's birthday, Lady Partypooper is responsible for a magnificent show which involves water falling from the sky to a big pool. However, after several tests and a few drowned minions, she decides to try her show in a simulator first.

You, of course, do not mind providing your assistance. You will build a fluid simulator based on Smoothed Particle Hydrodynamics (SPH). It should come with a GUI where you can specify the initial state of the water and the environment. It should be able to enable/disable different forces.

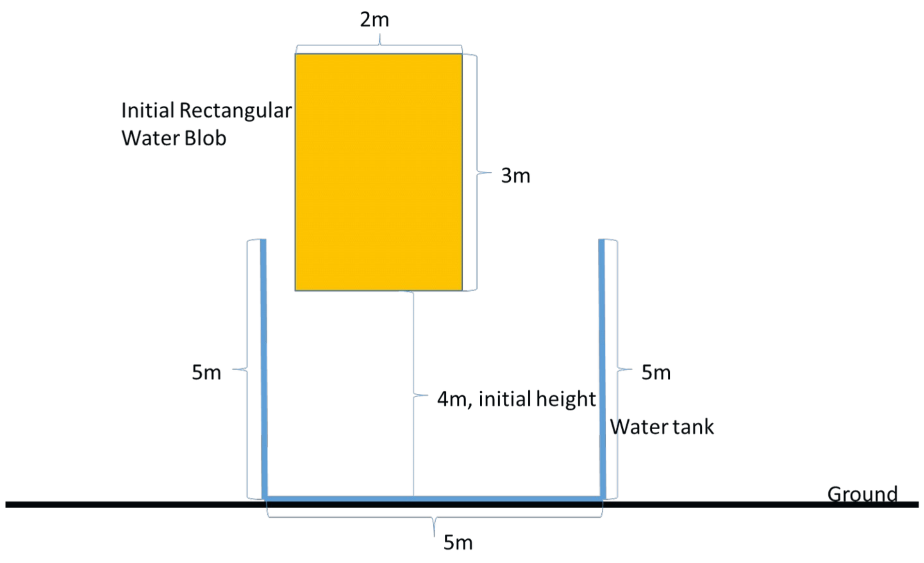
Lady Partypooper has graciously agreed that the first demo can be simple which only involves a couple of simple scenarios.

For your hard work, the Lady Partypooper will reward you for:

1. A GUI to set up the simulation environment, run the simulation and render the results. (**7g**)
2. A 2D SPH simulation with a water tank and a water blob floating in the air at first then free-falling under gravity (see Hint 1). The only forces considered here are gravity and internal pressure. The simulation should also consider the collisions between the particles and the water tank (but not between particles). The integration scheme should be the Leapfrog scheme. The kernel should be the poly6 kernel we introduced in the class. (**11g**)
3. The same simulation as 2, but replacing the kernel with Debrun's spiky kernel for pressure computation (only for pressure). (**5g**)
4. The same simulation as 3, plus the viscosity and the viscosity kernel introduced in the class. (**5g**)
5. The same simulation as 4, plus the surface tension with the poly6 kernel. (**5g**)

The total reward is **7g + 11g + 5g + 5g + 5g** = **33g**.

Hint 1: the initial state of the simulation:



Hint 2: the rendering can be performed via one of the following approaches:

- drawing each particle as a small disc, or
- using textures to fill each grid cell with transparencies controlled by the local particle density, or
- marching cubes to draw the free surfaces

Hint 3: the details are in the slides as well as https://matthias-research.github.io/pages/publications/sca03.pdf

Hint 4: There are several hand-tuned parameters, such as the viscosity coefficient, the time step, the radius of the kernel, etc. Don't panic if your first implementation doesn't work and don't assume it is necessarily caused by some bugs in your code. It might be just a parameter tuning issue.

Hint 5: You can also dampen the collisions between the metaballs and the water tank by multiplying the speed (a scalar) with a number between 0 and 1 after collision response.

Hint 6: Is the poly6 kernel really suitable for 2D simulation? Why/Why not? If not, what polynomial kernel is more suitable for 2D simulation and why?

The whole submission needs to be implemented in Visual Studio C++. It should be your own work except for any third-party libraries. Please list all the external libraries and code you used in your ReadMe file. Otherwise, it might risk plagiarism. Also, for the part you used the external code, the marks will reflect it. Make sure that you submit a zip file containing the *whole* Visual Studio solution (just the source files, not compiled binaries please). You are free to use third-party libraries for GUI, rendering, triangulation, video-capture and linear algebra operations, etc. But IT SHOULD BE STANDALONE SO THAT I CAN RUN IT ON MY WINDOWS COMPUTER.