In doing this project, you will learn how to use the OpenCL API and its kernel language OpenCL C to animate your particles.
In addition, you will display them using OpenGL, which will allow you to consolidate your knowledge of this API.
This topic is also an opportunity to start thinking about optimization. Don't be afraid to look for and test different approaches. In fact... you will have to do so. So do it, but it will be easier if you are happy to do it.

You will need to use OpenGL 4.0 minimum (with shaders of course) as well as OpenCL 1.2 minimum.
- Particles should never be allocated to RAM. Never. Even at startup. Never. If you are wondering if at such and such a time this is possible, the answer is: "Never". Everything will be allocated on video memory.
- In terms of performance, you must run at least one million particles at 60 FPS and three million (minimum) at 20 FPS. Make sure that the number of particles can be changed easily.
- You need to make interoperability and for cleanliness, you need to synchronise memory between the different APIs. The acquisition and release functions are your friends.
- You are free to use the language you want.
- You can use the graphics library of your choice (SDL2, SFML, GLFW, MLX, Glut...).
- A Makefile or similar is required. Only what is on your repository will be evaluated.

You'll have to implement a particle system. These particles will be initialized in a sphere or a cube and it will be possible to switch from one to the other with inputs. They can be attracted by a center of gravity that can be activated and deactivated by a key. This can either be static (placed under the mouse at the time of the input) or follow the mouse according to the input.
They will have to be allocated on the GPU memory (VRAM). Moreover, it will be necessary to respect some performance constraints despite the large number of particles: one million particles at 60 fps and three million at at least 20 (and there you will understand why the GPU is good).
You will also have to display the number of FPS in the window (in the title for example), this will make performance testing easier.
For the pleasure of the retina it will be necessary to put colors. They will have to depend at least on the distance of the particles from the cursor.
Remember that the cleanliness of the code for interoperability is really important. If you don't know what we're talking about, reread the general instructions.

When you are sure that everything is working and that your performance is good, here are some bonuses you might want to add:
- A camera that can move in particles, controllable with WASD and the mouse. Because that's class.
- Particle generating emitters with a limited lifetime. There will be points dedicated to these bonuses and others for your creativity.

!For bonuses, the performances requested in the constraints do not apply... don't abuse them either. 10,000 particles at 20 fps is abuse.