Minimum requirements:

At least 10 objects in the scene, at least 8 of them should be more complex (loaded from an external file, geometric primitives will not be accepted). Objects can be repeated throughout the scene, but there have to be at least 3 unique objects (for example chairs, tables, wardrobes, etc.). Other objects can be generated procedurally or saved in the code (cube, sphere, teapot, etc.).

At least 8 objects have to be textured, there should be at least 3 unique textures in the scene (for example 2 chestnut tables, 5 oak chairs, textured carpet). Other objects don't need to have a texture (for example silverware, utensils, etc.).

At least one object has to be (visibly) animated. It can change color, position, intensity of color and/or lighting and similar.

Scene has to contain at least 2 lights, at least 8 objects have to be lit and some of these objects should be lit by multiple lights.

These are the minimum requirements. Additionally, each project has to represent something meaningful (see below), a scene with randomly placed objects will not be accepted.

Project has to contain at least 4 of the following techniques:

- Non-trivial text (story narration, GUI, text allowing you to turn animations on/off, or change parameters of a scene, etc.)
- Environmental mapping with skybox
- Toon shading applied to entire scene (except for elements where it makes no sense, such as GUI)
- Bump/normal/parallax (occlusion) mapping and other texture mappings
- Blending
- Cone lights
- Fog
- Light attunement, ambient lighting and background lights
- Larger number - at least 10 - of (main) lights, for example streetlights next to a pavement. It is also possible for some of the lights to only affect lighting of certain objects in the scene
- Simple physics (such as gravitational force) with some sort of acceleration, linear movement will not be accepted
- Projection textures (for example a projector)
- Procedural textures
- Procedural geometry (generated terrain based on heightmap, object deformation, etc.)
- Plane mirror (mirror effect, water reflections, etc.)
- View frustum culling / portal culling, but only with larger scenes containing more than 50 objects
- Audio (sounds of explosions, item pickups, etc.)

Note that if you decide to pick any of the techniques, you have to fulfil all the points (for example, you have to implement both skybox AND environmental mapping if you decide to, merely adding skybox to the scene will not be accepted). If there is anything else, you'd like to implement let us know and we will tell you if it's enough to be accepted.

Text in brackets ( ) is considered a suggestion, unless stated as a requirement, you are free to come up with your own ideas as to what to implement**.**

Meaningful application:

Simple game: This kind of application can have multiple identical objects that repeat throughout the scene, same textures, multiple of simple animations, simple lighting and fixed camera position. The important part is for the game to be playable, meaning the player will have a task to finish or a goal to achieve. The game usually has a simple physics system (collision detection, velocity, gravitation, etc.) so that the player can't reach places they are not supposed to. The game also MUST contain a simple GUI, in which player sees things such as the goal/task of the game, controls, amount of acquired points, current difficulty, time, whether they succeeded or not, etc.

Examples: Snake, Labyrinth, Sokoban, Hanoi towers

Nice scene: This kind of application can have very few animations (fish in an aquarium, ticking clocks, etc.) and simple controls, but the focus is placed on the scene looking nice. It is supposed to have many objects, many different textures, moving camera you can use to zoom in and out and walk around, multiple lights and more complex shaders.

Example (this is all one scene): A room with a desktop and several chairs, a bed, a lamp, carpets, a wardrobe, bookshelves, paintings on the walls, clocks showing current time. The scene should contain point and cone lights with attunement.

Nice effect: This kind of application has a relatively small number of objects, but it has a complex shader that shows off a certain effect, for example a burning torch, a fountain, or firing a rocket launcher followed by an explosion. This application contains more complex animations, which can create and/or destroy objects (flames of a torch, water drop in fountain, other particle effects), it contains many animated/changing textures, such as lighting changing based on the intensity of a flame. Effect should be easily controllable, for example shooting a rocket launcher, or moving a torch around. Other objects in the scene can be simple and are mostly there to support the effect (for example to show how lighting changes when we come closer with a lit torch).

More information on lighting, techniques, etc.:

Objects and used lighting have to fit the scene and make sense. For example, we will not accept a table that emits light for no reason. We also don't accept a teapot with sky reflection placed in the middle of a room with a roof above it, nor normal mapping that doesn't react properly to lighting. Burning table can emit light, but it should be accompanied by flame animation. Food that players can interact with can also emit light to notify players.

Objects that are not supposed to be lit (flames, lightbulbs, UI), should not be lit.

Extra points:

If you implement an application with a care, be it an exceptional visual part (particle effects, with physics to them for example), or the code part (endless, procedurally generated maze), you can get extra points towards exam. Maximum is 5 points and they only count if you get at least an E from the written exam.

Deadlines and format:

Presentations of the projects will be held during the exam period. It is not necessary to defend your project before the written exam, but you will not be given a grade until you do (unless you get F from the exam). Actual dates for project presentation and exams will be put into IS, the format of presentation will be discussed on the seminars (feel free to ask your teacher anything you are not certain about)

FAQ:

What is a complex object, how complex should it be, is a table made from 5 cubes a complex object?

We consider objects loaded from external files (.obj files, etc.) as complex objects. This requirement is meant to stop students from creating projects built only from cubes stacked atop one another. Please restrain yourself from exporting cubes into .obj files and loading them to the project this way. This will not be accepted and frankly is just a waste of your time. We would like to see something more interesting than generic examples.

Can the game be ugly?

Maybe. It has to be a game first and foremost. Each type of the application focuses on different aspects of graphics programming (game focuses on interactivity, nice scene on shaders, effect on animations). For this reason, the game doesn't need to contain perfect lighting or great animations. But people (even those who didn't program the game) have to be able to play it.

Can we use third-party libraries?

Yes, but. You can use libraries that were used at the seminars (and this should be mostly enough for you to create the final project). You can also use different libraries for loading special formats of images, geometries, etc. but only if these libraries focus on the "C++" part and leave OpenGL functionality on you. Simply put, if you get an array of data as the result from the library it's OK, if you get an OpenGL buffer/texture, object with draw method and similar, then you can't use it.

Can we use different programming languages?

Yes, but. OpenGL is ported to many languages. If you choose to use anything other than C++ (with exception of Java), we will not be able to help you out (Python, Prolog, Haskell, …). You will be left out to help yourself or rely on information from the internet. This might not be the best thing shortly before the deadline. However, in the end this is your choice, you will have to present the project to use personally and we will be asking questions about it, so as long as it fulfils the requirements and you can convince us you know what you did, we will be fine.

Can it be a 2D project?

No, 2 is too few dimensions, but we will accept 2.5D projects (3D scene with orthographic camera). OpenGL is mainly for 3D graphics and we want you to use 3D objects in your project. This means you will have to work with 3D transformations, too, which is what we would like to see.