Administrative	
Group Name / Game Name	Treasure-Hunt
GitHub Link	https://github.com/Skrebov/cgue22-TreasureHunt
Students	Enikő Török, 12023999 David Skrebic, 11801320
Genre	Platformer/Treasure Collect
Goal	Evade obstacles and try to collect the treasure in the castle.
Graphics API	OpenGL
Gameplay	
3D-Geometry	Everything is being imported from Blender obj-files. The objects were created by us. For this we are using the Assimp library. For the import we used the tutorial in [9] and [10].
Playable/Advanced Gameplay	The player can move around and collect diamonds. He can die when he touches spikes or falls inside of a pit. He can also travel on moving platforms. If he collects a certain number of diamonds, he wins the game.
Min. 60 FPS and Frame-Independence	The FPS are being set to 60 by using glfwSetFrames(). The deltaTime is being calculated every frame and the relevant calculations all include the deltaTime to achieve Frame-independency.
Win/Lose Condition	You can win the game by collecting the diamonds. You lose the game when you touch deadly objects, such as spikes. You also lose when you fall inside of a pit.
Intuitive Controls	As in many PC-games the player is controlled using the WASD-keys(we use polling for that) and the mouse to look around. You can also jump using space.
Intuitive Camera	Look at the section “Intuitive Controls”, as we have a first-person game. The player and the camera are at the same position. There is also a “free” or debug-camera when, where you can look around freely. Press left_alt to access it. Be careful, as the collision detection is still on, even when in the free mode.
Illumination Model	As we import all the objects from obj files from Blender, we also import the normals. Furthermore, all the objects that are imported have textures. We have multiple light sources. One directional light source. Multiple point lights and a flashlight, which can be toggled using the F button. We partly used [11] for the theory of this.
Textures	As we import all of the models, all of them have textures.
Moving Objects	There are moving platforms and spikes. Some bricks can be moved by pushing them away.
Documentation	This document.
Adjustable Parameters	We included a settings file, where the settings of the game can be set. The file is to be found in the assets folder under “settings.txt”.
Optional Gameplay	
Collision Detection und Advanced Physics	Those aspects were implemented using the PhysX library. The Collision detection is implemented through the Character Controller, which we added, and which functions as our player. For the Advanced Physics we have included multiple dynamic objects, for example the dynamic boxes,

	which are used to build a wall and they are being pushed away when the character collides with them. Furthermore, we implemented kinematic platforms and spikes, which are being moved using the PhysX engine. We defined a ControllerBehaviorCallback to define how the game should act when the controller collides with another object. We also implemented a SimulationCallback to use TriggerObjects, such as the diamonds that can be collected and are then deleted from the scene.
HUD	The HUD was implemented using the FreeType library. At the current stage only, the collected diamonds are being displayed (at the left corner of the screen). At loss or at win a text is being shown as well. The HUD can be toggled using the Q button. You can also show a menu with the controls by pressing M. We implemented the HUD according to the tutorial. [8].
Effects	
Environment Map	We created a space themed skybox and applied environment mapping on the statue at the beginning of the level. This was achieved with the help of this tutorial. [12]
CelShading	The level itself is celshaded. We used your presentation for guidance. [13]
Vertex Shader Animation	The lava in the second pit (below the spikes) is animated using vertex shader animation, we used your tutorial as guidance. [14] For the recalculation of the normals we used this reference. [15]
Bloom	The diamonds and the lamps on the walls have a bloom effect on them. The implementation was according to this tutorial. [16]
CPU Particle System	In the second room particles can be seen falling. It was done, using the tutorial in [17].
Shadow Map with PCF	Shadow Map for the directional light, the shadows can be seen within the level. Done with the help of [18].
Input	
WASD	Move the player
SPACE	Jump
ESC	Quit
F1	Toggle Wireframe-Mode
F2	Toggle Backface-Culling
L_CTRL	Duck
L_ALT	Enter the free(debug) camera, where you can move around freely
F	Toggle the flashlight
B	Toogle bloom_effect
C	Toogle cellshading
Q	Toggle the HUD
M	Show controls
+(numpad)	Increase brightness of the scene.
-(numpad)	Decrease brightness of the scene.
Mouse movement	Look around
Additional Information	
Used libraries	NVIDIA PhysX SDK 4.1 [1] Assimp [2]

	Freetype [3] GLFW [4] GLEW [5] GLM [6] stb_image [7]
“Features”/How to play	You enter the game and explore the level. Try to collect all the diamonds to win the game. But do not touch the spikes or you will lose. Be careful around the pits as well! Use Q to toggle the HUD, use F to toggle the flashlight! Enter the level by pushing away the wall at your right hand side.

Sources

[1] https://github.com/NVIDIAGameWorks/PhysX
[2] https://github.com/assimp/assimp
[3] https://freetype.org/
[4] https://www.glfw.org/
[5] http://glew.sourceforge.net/
[6] https://github.com/g-truc/glm
[7] https://github.com/nothings/stb
[8] https://learnopengl.com/In-Practice/Text-Rendering
[9] https://learnopengl.com/Model-Loading/Mesh
[10] https://learnopengl.com/Model-Loading/Model
[11] https://learnopengl.com/Lighting/Multiple-lights
[12] https://learnopengl.com/Advanced-OpenGL/Cubemaps
[13] https://tuwel.tuwien.ac.at/pluginfile.php/2415211/mod_page/content/32/CelShading_SS19.pdf
[14] https://tuwel.tuwien.ac.at/pluginfile.php/2415211/mod_page/content /32/Animation_SS18.pdf?time=1619523068902
[15] https://catlikecoding.com/unity/tutorials/flow/waves/
[16] https://learnopengl.com/Advanced-Lighting/Bloom
[17] http://www.opengl-tutorial.org/intermediate-tutorials/billboards-particles/particles-instancing/
[18] https://learnopengl.com/Advanced-Lighting/Shadows/Shadow-Mapping