Documentation

Group/Game Name: **Gardening Simulator**

Brief description of implementation: A C++ OpenGL game. A gardening simulator, where you grow and take care of plants. You have to take care of a certain number of plants and try not to lose too much time, because then you have to dismiss some of them which will cost you.

Additional libraries:

- **ASSIMP-vc143-mtd**: for loading .obj files
- **stb_image:** to load .png files containing an alpha channel.

Gameplay:

Mandatory:

- **3D Geometry** (multiple external custom 3D assets)
- **Playable** (complete playable gameloop)
- **Advanced Gameplay** (Gameloop with different gameStates and different interaction possibilities)
- **Min 60 FPS** and **Framerate Independence** (yes, tested on three different GPUs)
- **Win/Lose Condition** (Integrated as Gameloop, player is informed through command line outputs.)
- **Intuitive controls** (WASD movement, for detailed Keybindings see keybinding section)
- **Intuitive Camera** (toggleable between scene camera and free floating ArcBall cam)
- **Illumination model** (PBR shaded as well as NPB stylised shading model. Both using directional light. PBR also uses multiple Point lights.)
- **Textures** (multiple custom made PBR Textures, including diffuse, roughness, metalness, AO and normal maps)
- **Moving Objects** (player, as well as potted plant if carried and water simulation)
- **Documentation**

- **Adjustable Parameters** (initial screen size, full screen mode, pointAmount, Lifes, ShadowMap resolution)

Optional:

- **Advanced Gameplay** (We implemented a dedicated playable game loop where the player has to interact with multiple game objects in the right sequence while being under time-pressure. The game state is tracked using the Plant/pot objects in the scene.)
- **Collision Detection** (Basic Physics) (Axis Aligned Bounding Boxes are calculated for the objects in the scene. For precision in interactions, a point in front of the player is calculated out of the player bounding box. Collisions happen when the player box and a box in the environment overlap, be it with something on the map (checkCollision) or if the player wants to exit the playable game world (isInside). In either case the player cannot move into (or out of) the thing. Interactions are reported when the point overlaps with an interaction box in the game world.)
- **Heads-up Display**: (Dedicated HUD-manager class, rendering all sprites on top of the 3D viewport if enabled. Sprites consist of simple .png textures with an alpha channel that are loaded during startup and then rendered in front of the viewport using an orthographic transformation matrix)

Effects:

Lighting:

- **Shadow Map with PCF**: (A shadow map is rendered each frame as a first pass, using a simplified shader. The information is then used by NPB and PBR shaders to create shadows inside the scene during the main rendering passes, Shadow Acne as well as Peter panning was countered by introducing a variable shadow bias and by enabling frontface culling during the shadow map rendering.)

Advanced Modelling:

- **GPU Particle System** using Compute Shader: (A custom set of shaders, including a compute and a geometry shader, was developed to enable a simple physics based particle simulation. The Simulation is used to simulate a water sprinkler, to animate the watering of plants. The whole functionality is packaged into a dedicated component, The Watering Manager class.)

Animation:

- **Hierarchical Animation:** (Hierarchical Animation is used to make the player hold a plant during walking around. The player Object has a specific slot for a plant, that is rendered relative to the player's modelMatrix, so that the player can carry around plants).

Texturing:

- **Specular Map:** (As part of the PBR shader mode, specular maps and metallic maps are used to increase the different materials of certain objects, such as the water pipe or the watering table.)

Shading:

- **Simple Normal Mapping:** (As part of the PBR shader mode, normal maps are used to further increase realism)
- **Physically Based Shading:** (The game has a PBR shaded mode, which uses a custom made set of PBR shaders, that take different material properties via texture maps into account and can handle multiple light sources)

Post Processing:

- **Contours via Edge Detection:** (During the NPB shader mode, a two-phase render pass is enabled, the first pass produces a simple diffuse, normal and shadow map. The second pass is used as image post-processing for the generated maps and introduces a simple sobel edge detection as well as a blur filter to generate nice, stylized outlines over hard edges.)

Other special features:

The F1 key allows the player to switch between a realistic PBR mode, and a stylized NPB shading style.

The F2 allows switching between the scene camera and a special debug camera that can be used with the mouse to turn the scene around while playing.

The F3 key is used for toggling the HUD display on or off.

Walk-through:

When the program starts, the scene camera is switched on and the player can freely move inside of the floor bounding box with ‘WASD’ keys and jump on the watering station and the

dirtpile with the ‘SPACE’ key. For playing, the player has first to go to the counter on the left, where after some time pots are continuously generated, there are 4 slots for pots.
When the player takes a pot with the ‘E’ key from one of the slots, he then has to go to the dirt pile on the right side of the room, where he has to interact again to get some soil.
After that, the player has to go to the watering station and interact with one of the 4 slots there to start the watering process, which causes the plant to be taken from the player and put on the watering station. After watering is finished, the player has to interact again with the now grown plant and put it back on a slot on the counter on the left.
If the player lost too much time and the counter is full of empty pots, he either has to dismiss the plant with the ‘R’ key, losing a life in the process, or put the plant back on the watering station (which can be used as temporary storage for empty pots and finished plants) and then proceed to go to the counter and take a new pot (or dismiss it, losing a life).
When all lives (4 in the submission ‘gameSettings.ini’ config file setting, adjustable) are lost, the game is over.
When the player manages to bring enough (3 in config file, adjustable) finished plants back to the counter, the game is won and a new game can be started by resetting with the ‘ENTER’ key. There is also a pause menu, when pressing ‘P’, which can be exited with ‘P’ again or with pressing ‘ENTER’, which will reset the game.

There is also a heads up display, which shows the remaining lifes, as well as an “E”, when the player is in an interaction box and can interact with it.

All interactions, the gathered points, as well as the beginning and end of the game are accompanied by console outputs.
The game can be quit by pressing the ‘ESC’ key.

The screen size and full screen mode can be changed in the ‘window.ini’ file.

Usable keys:
W - Move 'up'
A - Move 'left'
S - Move 'down'
D - Move 'right'
E - interact
R - dismiss plant
P - pause

F1 - Shading style Toggle
F2 - toggle debug camera
F3 - toggle hud
ESC - Quit game
Enter - reset game if the game is paused or ended
Space - jump