Administrativ

Group Name / Game Name	Raupi‘s Irrgarten
Github Link	kusi0112/cgue21-raupis_irrgarten (github.com)
Students	Kuchler Sarah, 11776811 Moser Markus, 11911065
Genre	Labyrinth
Goal	Find all the food before the day ends

Game Idea and Content

Story (max. 4 sentences)	Raupi is a caterpillar living in the wild nature. His main goal is to become a beautiful butterfly and to achieve it, he must eat a lot. Help Raupi to find all the available food before the sun goes down and the day ends.
Gameplay (max. 4 sentences)	The nature is a labyrinth where Raupi must find his way through. He can only walk on the ground, high grass or big rocks are barriers for him and when Raupi falls below the start level, he dies. By achieving the goal of collecting all the food placed within the labyrinth before the timer ends, a new day (and level) can be started.
User Interaction	In Levels: ✓ WASD: Movement of Raupi ✓ Mouse: Camera view direction ✓ ESC: Quit level ✓ Space: Jump ✓ Space: Leave level when won or lost In Menu: ✓ 1: Start level 1 (in menu) ✓ 2: Start level 2 (in menu) ✓ ESC: Quit game
3D Objects	Static ✓ Labyrinth (gras, stones, …) ✓ Food (to collect by walking over it) Dynamic ✓ Raupi (3rd person main character) ✓ Dandelion particles ✓ Falling stones Trigger Shape ✓ Food

Scene lighting	Sun light (will get darker when day goes by) – directional light and point light

3D Geometry

All models, textures and animations were designed in blender and loaded with the assimp library (LearnOpenGL - Assimp). After loading all models get a matching PhysX shape (NVIDIA PhysX SDK 4.1 Documentation — NVIDIA PhysX SDK 4.1 Documentation). For example plants have static cylinder shapes, Raupi is a capsular character controller and the falling stones are dynamic shapes created with a triangle mesh.

Additionally, the grass tufts are loaded only once and replicated in the level. The positions of the grass within each level is set by CSV-Import to avoid loading the model multiple times. The CSV file is generated from blender using a phyton script to export the object positions.

Playable & Advanced Gameplay

When starting the game, the player sees the menu. Here he can navigate either to level 1 or level 2 depending on key pressed. Within the level Raupi is moved by pressing WASD and Space. The goal is to find all the food (3 pieces per level). When all food is found before the time runs out the player wins and Raupi will be happy. On the other hand, there are two lose-conditions. When the timer runs out (different time for each level) or Raupi falls below the current level height, the game is over. The information how many pieces of food are collected, the timer and if you won or lost can be found in the HUD.

Min. 60 FPS and Framerate Independence

The rendering adapts to the current rendering speed, the time spent between two game loop iterations is calculated and used in movement and speed.

Win/Lose Condition (3 Points)

Win condition = find all food before time runs out
Lose condition = time runs out or Raupi falls below the current level height

Intuitive Controls (2 Points)

The navigation within the level is like in other games: walking with WASD and jumping with Space. Quit is ESC. When other key presses are allowed, they will be described within the HUD (for example: “press 1 for level 1”).

Intuitive Camera (2 Points)

The camera is a third person camera which always has the same distance to Raupi. The camera can be rotated using the mouse.

Illumination Model (2 Points)

A directional and point light is used. The illumination color gets dimmed during the level until the day is over. The normals are loaded together with the model from blender.

Textures (2 Points)

Each object loaded has a texture which was attached in blender. We tried to use simple textures to improve the cartoon effect - especially the cel-shading.

Moving Objects (2 Points)

The main player Raupi as well as some falling stones and the particles are moving objects. Raupi can be controlled, particles have a random flight route and stones fall from large rocks using gravity.

Adjustable Parameters (1 Point)

A settings.ini is provided and read by the program to configure some parameters like fullscreen-mode. Changing those parameters will therefore affect the game. Additionally, a level.ini for each level is used to configure the particularities of each level and a menu.ini is used to configure the main menu of the game.

Additional libraries:

- ✓ Assimp (Home (assimp.org))
- ✓ PhysX (NVIDIA PhysX System Software)
- ✓ Freetype (The FreeType Project)
- ✓ irrKlang (irrKlang - audio and sound library for C++, C# and .NET (ambiera.com))

Features

Category	*Feature*	*Description (Usage)*
Optional Gameplay	Collision Detection (Basic Physics) (4 Points)	Avoid player to go through walls (grass, ground, stones) of the labyrinth using PhysX.
	Advanced Physics (6 Points)	Sometimes small stones fall to the ground and block the way. They are created as dynamic objects. Trigger shapes are used for food, to know when the food is eaten.
	Heads-Up Display (4 Points)	HUD is rendered in 2D after scene. It includes information for the player as text: ✓ How many pieces of food are collected/left ✓ Time left before day ends ✓ Some text like “day ends soon”
Effects	Lighting: Lightmap using Separate Textures (8 Points)	Baked texture exported from blender is imported and used in the program. Only found in level1.

	Animation: GPU Vertex Skinning (20 Points)	Used to animate Raupi when he walks around. The animation stops if Raupi does not move. The model of Raupi and the rigging were created by us, and the bone transformations were loaded using assimp. Link: Blender Character Animation: Walk Cycle - YouTube
	Shading: Cel Shading (4 Points)	All meshes are influenced to get a Cartoon-Effect – seen well at food shape. Is implemented with four color levels within the shader. Link: OpenGL 3D Game Tutorial 30: Cel Shading - YouTube
	Advanced Modeling: CPU Particle System (8 Points)	Dandelion particles flying around in the area where dandelion plants are. Each dandelion plant generates a particle system. The particles will disappear after a certain time. Link: Coding Challenge #78: Simple Particle System - YouTube
	Post-Processing: Contours via Edge Detection (12 Points)	In combination with cel shading to get a Cartoon-Effect; whole scene is affected. This post-processing effect was implemented using a FBO to render the scene in and a separate shader with an edge-detection kernel. Link: LearnOpenGL - Framebuffers

Other special features

Feature	*Description (Usage)*
Skybox	Around the scene a textured cube is rendered to simulate environment. Different textures per level are used.
Music	Background music in menu and level. Additional sound effects are used when eating food and the level is either won or lost.
Lighting	The color of all light sources is dimmed when playing the level. The less time is left, the darker it becomes.
Menu	We created a menu to navigate to the different levels, it includes music and animation, but Raupi and camera cannot be moved. Additional information is provided using text in hud.