SAE Berlin GPR
0919 Final Exam

Available time: 4 hours

Goal:
Starting from ***this repository***, clone the repository and complete all the tasks described below.

Content:
The repository contains a simple game prototype developed with Unity 2020.2.
In it you need to help *Eve* stop an invading army of Lizards by using her magical spells.
Scenes/VillageScene is a good entry point into the project.

Submission format:
Upload a zipped folder to Canvas containing the following elements:

- Git repository with per-task commits
- Working Unity project
- A final Build of the game for Windows

When committing a solved task write an extensive commit message answering the following questions:

- What was the problem?
- How did you resolve it?
- Where can I find the finished product?
- Where can I test the result?

Commit message Example:
"ADDED: Prototype0 Electric Blast
I implemented the spell using a simple projectile which deals damage upon hit. I implemented it into the SpellCastingController. It can be edited at ScriptableObjects/Spells/BasicAttack and the used prefab is called ShockProjectile. Simply click the left mouse button ingame to test it."

Notes:
You are expected to work with Git by committing into the starting repository. Work in your own branch and merge into main when you are finished. (You are not expected, or allowed, to push.) Git usage will affect the final grade.

You are allowed to freely use external resources (Internet, books). But **communicating** during the exam with other people (notably the other students taking the exam) **is not allowed** and can be considered cheating.
You are allowed to ask questions during the exam. The examiner will function as a makeshift designer and will answer design questions. Technical questions will not be answered.

Make yourself familiar with the project and codebase before writing your own code. You are expected to follow established guidelines and to reuse existing features when possible.
It is expected that both VillageScene and TestingScene will remain functional and without errors upon submission.

The exam time starts once questions about the tasks have been answered and the project is cloned and opened.
Changes committed after the deadline will be reverted.
Since this exam is taking place online the submission can be uploaded up to one hour after the deadline to account for connection issues and upload speeds. If more time is required make sure to communicate it after the exam time.

Tasks:

There are 7 tasks divided into 3 categories for a total of 90 points. 10 additional points will be distributed based on documentation, git usage and overall implementation.
Up to 10 bonus points can be awarded for completing the bonus task.

1. Bug-fixing tasks

   These represent “bugs” and are formulated the way they would be reported by designers or QA. They are not necessarily code mistakes. You are expected to fix them without changing overarching structure and design.

2. Iteration tasks

   These are tasks where you are expected to iterate on the existing systems by modifying and expanding the given code, while ensuring that previous functionalities continue to work as intended. This code is supposed to be stable and production ready, do not modify the overarching structure unless strictly necessary.

3. Prototype tasks

   These are tasks where you are asked to prototype a new feature by building on top of the existing code.They likely require more than just a straightforward implementation as you might be required to also setup new underlying systems and structures.

Bug 1: (5 Points)

“When a lizard-Enemy drops the “Lizard Soul” loot upon death the hovering text is colored in black, but it should be colored in blue as rare items are supposed to have blue coloring.”

Bug 2: (10 Points)

“When a lizard-Enemy dies the drops only spawn once the body disappears, at the same time the dead body often blocks projectiles flying above or collides with the player. The loot should drop the moment the enemy dies and once on the ground, the body should not collide with anything.”

Bug 3: (10 Points)
"The drop rates of the lizard enemies are bugged. The lizard souls items drop a lot less often than gold. Sometimes nothing is dropped at all. This does not match the loot description which says that both Gold and Souls should drop with a 50% chance, ensuring that always something is dropping."

Iterate 1: (10 Points)
Missing VFX and communication upon landing a basic attack.
Spawn the *ElectricHitEffect* prefab when a ShockProjectile (basic attack) deals damage. The object should be spawned at the point of contact and should despawn once the effect is over.

Iterate 2 (v1): (15 Points)
Picking up drops should be reflected in the UI. ("+1 Gold" from gold and "You feel strengthened by absorbing a soul" for the lizard soul) This text should be customizable on a per-drop basis in the respective scriptable object. The UI element should spawn on the top-center part of the screen and shrink in size over time while fading away, before finally despawning. If a new message is to be displayed, the old one is immediately discarded. The new one takes its place.

Iterate 2 (v2): (15 Points)
The UI should reflect when an ability is actively being used.
A spell being used should be highlighted in the UI, instead of looking "normal" until it goes on cooldown.
Add an outline to the ability UI which activates while the spell is being channeled/used. The ability UI should also grow in size while being casted and shrink back to its original size when the ability goes on cooldown. The increased size as well as the transition time should be customizable in the inspector on a per-ability basis.

Prototype 1: (20 Points)
Prototype a special ability slot. Like the basic attack, the special ability has a UI element that visualizes the cooldown. The second ability should start off by being empty. The player should then be able to "pick up" a special ability from loot drops. Upon pickup the UI should display the newly equipped ability, which should then be usable by clicking the right mouse button.

The feature should be structured in a scalable way, meaning that new abilities can be added without making any modification to the core special ability code or to other special abilities. In your document describe the steps necessary to add a new special ability to the system.
For testing purposes add the BasicAttack (ShockBlast) as the first ability that can be picked up and used as a special ability, by giving it as a 10% drop rate to the Lizards. Picking up a second special ability should replace the previously collected special ability.

Prototype 2 : (20 Points)
Building on top of the previous task, prototype the following ability. This ability should be dropped by the Lizards with a 10% drop rate.

Prototype 2 (v1): Earth wall:
"A wall grows out of the ground, blocking the enemies."

When cast, this ability spawns a wall which grows out of the ground with a linear movement. The newly created wall should be solid and count as an obstacle in the navigation of the enemies, causing them to take different paths to avoid it.
If, while growing out of the ground, an enemy is hit by the wall, the enemy should take damage. This ability should only work when cast on top of green grass; Trying to cast this ability indoors or on a path should do nothing.

- Use the *Village_Hut_Wall_front* mesh as the default mesh for the wall
- Use channel up as the casting animation
- Damage values should be editable in the scriptable object

Prototype 2 (v2): Fire trap:
"Eve places a rune on the floor which explodes when hit by an enemy, dealing damage to all enemies in an area"

First, Eve casts a rune on the ground, which stays there for a set amount of time.
If an enemy walks over the rune, it explodes, dealing damage in an area. Make sure to match the area of effect and timing of the damage to the "fireShot" effect. From a player standpoint, it should look as if the fireShot itself dealt the damage.

- Use the *FireShot* prefab for the visuals of the actual explosion
- Use the *FireHoverBall* prefab for the visuals of the placed trap
- Use the channel down animation as the casting animation

Prototype 2 (v3): Electric Chain:
"An elektric homing projectile, which bounces up to 4 times between enemies after the initial hit"

When Eve casts this ability, a projectile is spawned which locks onto the closest enemy in front of her. The projectile then flies towards that enemy, adapting its movement like a homing missile. If a first enemy is hit and there are other enemies nearby, the projectile "bounces" between them. The range as well as amount of bounces should be editable in the ScriptableObject.

- Use the *ElectricBall* prefab for the visualization
- Use the "channel forward" animation as the casting animation
- The projectile can bounce to the same target multiple times

Bonus Task:

Bug 3 Unit Tests: (BONUS 10 Points)
To ensure the correct behaviour of loot drops write EditMode Unit Tests that specifically test for the following problems from Bug 3:

- When two drops are specified with a 50-50 distribution, there should never be a case where nothing gets dropped.
- The order in the drop definition should not affect the result. Given A with a probability of 20% and B with a Probability of 80%. The loot descriptions AB and BA should result in the same drop rates.