**PLATFORM GAME ASSIGNMENT**

**iSAS code: *BP\_Platform***

**Maximum Grade 6.5**

**Game Rules**

Purple: I don’t understand it?  
Green: Done

1. Create a playing field using a plane. Add some floors, obstacles and endless pits to make it an interesting level. Use primitive shapes only.
2. Some floors in the level should be of a different physics material (with different friction and/or bounciness). Make sure these floors also have a different colour.
3. Create a player character. The character consists of a body, two hands and two feet. Use a capsule for the body, spheres for the hands and the boxes for the feet. You may scale the shapes and apply colours to your liking.
4. The character should be able to move forward, backward and rotate.
5. The character should be able to jump while standing on the ground and be affected by gravity.
6. Create some enemy characters that move toward the player until they are at a certain target distance from the player. This target distance should be the same for all enemies and should be stored in a Scriptable object. Whenever an enemy reached this target distance, it has acquired a target lock on the player.
7. Use the character model for your enemies but change one or more details (like the colour). Don’t try to make the enemies too intelligent, they don’t have to move around obstacles or avoid endless pits.
8. The level must contain at least two moving platforms capable of transporting players and enemies. One platform should move in horizontal direction (global x- and/or z-axis) and one in the vertical direction (global y-axis).
9. The player should be able to shoot rotating cubes that disappear after a couple of seconds. There can by only three rotating cubes in the game at any one time (to prevent the player creating an endless stream of cubes).
10. When a cube hits an enemy, the enemy should be pushed back by the impact of the collision and this cube should be destroyed.
11. After being hit by three cubes, the enemy should be destroyed.
12. As soon as an enemy acquires a target lock it stops moving towards the player and creates a gravity inversion field at the current position of the character. Use a capsule collider to represent this inversion field.
13. As long as a character is inside this volume, its gravity is inverted. You may choose whether you want the enemies to be affected by the inversion field or not.
14. After a couple of seconds the gravity inversion field disappears and the enemy that created this field starts moving to the player again.

**Other Requirements/Remarks**

* Use a RigidBody for player and enemy movement. You can use forces and/or MovePosition and MoveRotation to move and rotate your player and enemies.
* Do not use a CharacterController for player and enemy movement
* Use a ScriptableObject to define the target distance to the player and the duration of the special attack.
* Make sure to use Update and FixedUpdate correctly.
* Your Game Object Hierarchies should not have more than one RigidBody (probably on the top most parent object).
* Static collider (GameObjects with a collider and without a RigidBody) should not move.
* Even though your characters might have game object hierarchies, use only one, simple collider for each character.

**Maximum Grade 7.5**

**Additional Game Rules**

These game rules replace the game rules 12 through 14 regarding the casting of the gravity inversion field.

1. As soon as an enemy acquires a target lock, it should publish an event, stop moving towards the player and start jumping up and down at the current position. The target lock remains in place even if the player moves out of the original target distance.
2. After a couple (three?) seconds the enemy loses the target lock on the player. When this happens, the enemy should publish a different event, stop jumping and start moving towards the player again.
3. Whenever three (or more) enemies have a target lock on the character, they will create a gravity inversion field at the current position of the character.
4. As long as the character is inside this volume, its gravity is inverted. Use a capsule collider to represent this gravity inversion field.

**Additional Requirements/Remarks**

* Use coroutines for the enemy behaviour. Try to completely avoid the Update function for the enemy behaviour script(s).
* Publish an event whenever an enemy acquires a target lock and whenever an enemy loses its target lock. Implement the special gravity inversion attack by listening to these events.

**Maximum Grade 8.5**

In addition to the requirements mentioned above, you should also implement a feature that enhances the game play and uses a system we did not cover in class (for example audio, UI, lighting, materials, animation, etc).

If you’re aiming for this grade, discuss your plans with your teacher before handing in your work.

**Maximum Grade 10.0**

Implement all rules and requirements for the maximum grade of 8.5 and make the game really playable. So put effort in the design of your level, the handling of the player, the placement of the enemies and the visual design. Also make your game a complete game with a start, and end screen.

If you’re aiming for this grade, discuss your plans with your teacher before handing in your work.