Task 1 Design Rationale

Oskar Hosken - 31450628

Creating a new abstract class called Map and inheriting all the maps (as a list of strings) from it:
In order to move between maps and have the name of each map show in the display of the game, I thought it was necessary to create an abstraction that allows all game maps to have a menu description per se. I have named the abstract method as menuName(). This allows for the future Teleport Action to display as “Mario teleports to ‘menuName()’”.
This also cleans up some of the code in the Application class and enhances the Single Responsibility Principle in our code as the Application class is no longer used to initialise the maps. We instead have a new abstraction to take care of that.

Creating LavaZone as a new map:
The LavaZone map is a smaller, harder map. I chose to make the map roughly half the size of the original as it saves some time for the player as they don’t have to move as far to get to Bowser for instance.
I added a lava lake in the centre with a bridge over the top purely for looks and randomly placed the Lava on the map. In the top left corner is a little arena for Bowser and Peach, with the bottom right dedicated to a safe zone for the player to get away from Bowser as he cannot walk on the Floor() ground type.
Additionally, the top left is a little room for the warp pipe.

Creating a WarpPipe ground:
Firstly, this extends the Ground class because it can be jumped onto and entered, it cannot be picked up or consumed.
The WarpPipe was difficult to implement. It takes the map it is on, the map it leads to, the location it is at and the location it leads to as parameters when being instantiated. It uses these parameters for the teleport action associated with it.
I used the tick() method to make sure that the Piranha Plants don’t spawn on the first round, but the second. This is simply done by using a counter and when it is incremented after the first turn, the Piranha Plant is placed on the Warp Pipe.
Next, if the Warp Pipe doesn’t have a Piranha plant on it, but has an actor on the location and the actor hasn’t been on that location before, we add a new teleport action that goes to the Lava Zone (if it is on the original map and vice versa). I then set a flag to true to indicate that we do not need to add another teleport action here if the actor decides to not go down the pipe.

Additionally, if the warp pipe doesn't contain any actors (piranha plant or player) we add a new jump action available for the player to use.
And finally, to reset the warp pipe we simply add the piranha plant back on top of the pipe. Note that I have created one instance of a piranha plant per warp pipe. This avoids us creating multiple instances of piranha plants for each warp pipe.

Creating a Teleport Action:
The teleport action is necessary for the player to go between maps and to enhance the experience of going down a warp pipe.
This action simply takes the same parameters as a warp pipe and uses the gameMap.moveActor method to move the actor between maps as if they are teleporting. If the player is teleporting to the Lava Zone, I create a warp pipe at the top left corner (where they teleport to) that teleports the player back to the original warp pipe they came from. This does create multiple instances on top of one another on the map but it was the only way I was able to make sure the player teleports back to the same warp pipe they came from.

Creating the Piranha Plant:
As for the Piranha Plant. It acts very similarly to every other enemy, except it cannot move.
I overwrote the getIntrinsicWeapon method, giving it 90 damage with the verb 'chomps'.
When it is reset, we simply increase its max HP by 50 and then set its HP to that number.
Each turn we check whether the player is nearby and if so, it will attack. Otherwise, it will do nothing and stay on the warp pipe.