Clicker 1

Due: Friday, March 1 @ 11:59pm (Scala Only)

Updates and Pitfalls

-

Objective

Gain experience working with inheritance and polymorphism

Description

You will write the core code for a basic clicker game. A GUI has been provided for the game to help you visualize what you are building. This game will have the following properties

- The goal is to collect as much gold as you can
  - The player starts with 0 gold
- The player can click a gold button and earn 1 gold per click
- The player can purchase 3 different types of equipment to increase the rate of gold collected
- Some equipment automatically collects gold without requiring clicking
- The state of the game (gold, buildings purchased, time) can be saved as a JSON string
- The saved state can be loaded and all offline gold will be collected
  - Ie. When the game is loaded all gold collected by the equipment in real-time should be added even when the game was not running

Project Structure

1. Create a new project in IntelliJ
2. Pull the Scala Examples repo and copy the clicker package into the src folder in your new project
3. In the src folder create a package named tests

Testing Objectives (45 points)

Testing Objective 1

This game will contain 3 different types of equipment each with different properties and a class determining the behavior of all equipment of a particular type. Each of these types will extend an Equipment class that defines 3 abstract methods:

- goldPerSecond returning the amount of gold per second collected by all equipment of the type
- goldPerClick returning the amount of gold collected by all equipment of the type whenever the gold button is clicked by the user
- costOfNextPurchase returning the amount of gold needed to buy another one of this type of equipment

In the Equipment abstract class:

- Implement the buy method to increase the quantityOwned by 1
- Make the other 3 methods abstract by deleting their bodies

In the Shovels Class implement the 3 inherited abstract methods such that:

- Each shovel adds 1 gold per click (remember to multiply by quantityOwned for each of these methods)
- Shovels do not collect any gold per second
- The first shovel costs 10 gold and the price increases by 5% after each purchase

In the Excavators Class implement the 3 inherited abstract methods such that:

- Each excavator adds 20 gold per click
- Each excavator adds 10 gold per second
- The first excavator costs 200 gold and the price increases by 10% after each purchase

In the GoldMines Class implement the 3 inherited abstract methods such that:

- Gold mines do not collect gold on clicks
- Each gold mine adds 100 gold per second
- The first gold mine costs 1000 gold and the price increases by 10% after each purchase

In the tests package write a test suite named TestEquipment that tests all the proper state and behavior of these classes.

Testing Objective 2

In the Game class implement all methods needed to play the game.

- goldPerSecond returns the total gold earned each second from all equipment
- goldPerClick returns the amount of gold earned per click from all equipment, plus 1 from the initial click amount
- clickGold that adds goldPerClick to the gold total
- buyEquipment that takes one of the three equipment key strings [“shovel”, “excavator”, “mine”] and purchases one of that type of equipment if the player has enough gold to afford it
- Update that takes the current epoch time in nanoseconds and updates the amount of gold the player has based on the goldPerSecond and the amount of time passed since the last time this method was called

In the tests package write a test suite named TestUpdate that tests all this functionality

Testing Objective 3

In the Game class implement toJSON and fromJSON to store and load the state of the game. The saved/loaded JSON should be in the format:

```
{
 "gold": 3410.1482007745067,
 "lastUpdateTime": 1465620069483139,
 "equipment": {
  "shovel": {
   "numberOwned": 10,
   "name": "Shovel"
  },
  "excavator": {
   "numberOwned": 9,
   "name": "Excavator"
  },
  "mine": {
   "numberOwned": 7,
   "name": "Gold Mine"
  }
 }
}
```

And all three state variables must be set when fromJSON is called with a string in this format

In the tests package write a test suite named TestJSON that tests this functionality.

Primary Objective (35 points)

The entire game works according to the specs defined in this document. There is no additional functionality for this objective. If all three testing objectives are correct this objective should also be met.