## QA Puzzle

## Nordeus - Job Fair 2023

Hey there, future QA superheroes! Are you ready to dive into the exciting world of testing, debugging and many more? Get ready to flex those testing muscles and put your bug-busting skills to the test, because we’ve got a thrilling task lined up for you.

## Let’s get started…

Get your strategy hat on and prepare for an exhilarating battle of wits and cards. Here’s guide to navigating this exciting game:

**Player Duels:** It’s a two-player showdown! Each player starts with 20 points of health. Your mission? Outsmart your opponent and make him lose all his health!

**Deck Power:** Each player gets its own deck of 25 cards.

**Starting Hand:** Both players draw 6 cards from the deck at the beginning of the game.

**Draw Power:** On the start of the player’s turn the player draws 1 card from the deck.

**End of Turn:** Turn finishes when the player chooses to end it, when the player’s hand is empty, or when the player attacks his opponent (player can only attack once per turn with one Attack Card).

**Farewell, Card:** Every card vanishes from the player’s hand after it’s used.

**Victory or Defeat:** There are only two ways this game can end. Either one player loses all its health points, or someone finds themselves with no cards left to play which makes him lose the game(no cards in both deck and hand).

**Types of cards in this game:**

1. **Protect Card (index 1) :** When a player is under attack, the player can use this card at the beginning of its turn to dodge the bullet and keep its health intact. This card can be played at any time during the turn, but its effect will only be activated when the player is attacked.
2. **Boost Attack Card (index 2) :** When played it adds 3 additional attack damage during turn in which is played (this is only boost for Attack Card, player can not attack with this)
3. **Attack Card (index 3-7) :** Attacks a player’s opponent by the amount of damage equal to the index of the card. **Special ability of Attack Card :** If a player is attacked by the amount of damage that is equal to one of its own Attack Cards, player can play one of those cards at the beginning of its turn to dodge the bullet and keep its health intact (Note: After using special ability of attack card player gets to play his own turn. Opponent can not be attacked with the same card which special ability is used for deflecting the attack).

Fork Card Game Project Repository to your GitHub

## What is your goal?

Like nothing is perfect, every piece of code is more or less likely to have some bugs (like one you will be provided with… It has been written in a bit of a hurry before summer vacation. Sorry!). Responsibility of every developer and especially QA engineers is to minimize the probability that some code has bugs. It is known that the later the phase of development is, the more expensive fixing a bug becomes. It would be perfect if we could find all bugs before any execution testing. So let’s try to do that!

## About the challenge

1. Write thorough TDS for this card game. What is TDS? **Test Design Specification** (TDS) is a document that outlines the detailed design of test cases and testing procedures. This document typically describes the specific inputs, expected outputs, and steps for executing tests. It helps ensure that testing is systematic and thorough. Examples: *Player takes damage by 6 -> Player’s health decreases by 6. Player draws a card -> Number of cards in deck decreases.* Be sure to cover all edge cases and all possible scenarios! Also, if there are some open questions you have, write them down and make reasonable assumptions. Write test cases in the provided TDS document.

Note: This step is probably the most important one! Bad TDS = bugs in production

1. Now that you are finished with making the best possible TDS it’s time to write some **Unit Tests**! If your TDS is detailed enough it should be easy for you to cover the code with Unit Test in order to make sure there are no bugs in this game. Make sure to cover *Player.java* and classes in *cards* package with Unit Tests (if you have time you can then cover *Game.java* class with Unit Test as well). One example of one Unit Test can be found in class *TestPlayer.java* in the package *test* (it tests method *takeDamage* from class *Player*).
2. Now that you covered your code with Unit Test some might have failed. Oh well, as we expected, our code has some bugs. Give your best and find all the bugs and fix them! Write down each bug you found in the Bugs document and describe each bug - what is the bug and how you fixed it.

***Note:*** *Be aware! Not all bugs can be detected with Unit Tests.*

***Pro Tip****:* Sometimes original methods and functionalities are too time-demanding while testing and searching for bugs. So, you can write some fake methods (we call them *test commands*) in order to make your testing go faster! If you decide to use them make sure to leave them in code (start the name of that function with prefix *testCommand*) so we can see all the great test commands you came up with. 😀

## Set-up

* [Install Java 17](https://www.oracle.com/java/technologies/downloads/)
* [Fork Repository to your GitHub](https://github.com/LukaVuksic/qa-job-fair) ([How to fork a repo?](https://docs.github.com/en/get-started/quickstart/fork-a-repo))
* [Download IntelliJ IDEA](https://www.jetbrains.com/idea/download/) (If you want you can use IDE of your choice (Visual Studio Code for example))

#### **Submission format**

Please note that each challenge can be completed in less than 2 weeks (e.g. a few days) but we wanted to be mindful of your faculty obligations.

Submission should be sent to email [jobfair@nordeus.com](mailto:lukav@nordeus.com) with a link to the forked Github project with attached TDS and Bugs files. Email subject should be: **QA Puzzle.**

### Challenge is open until November 20, 2023 end of day. Good luck!

*The future of our card game kingdom rests in your hands. Are you up for the challenge? Let’s embark on this epic quest together and make testing and debugging a heroic endeavor!*

*Adventure awaits – let’s play, test, and conquer together!*