# Adventure Quest: A Simple Turn-Based RPG with Java and Database (Optional GUI & Difficulty)

## Overview

This assignment involves developing a turn-based RPG game called "Adventure Quest" using Java. Players create a hero, navigate a dungeon, battle monsters, collect items, and earn high scores. The core functionalities will be implemented using fundamental Java concepts like classes, methods, loops, and conditionals. The game can be either text-based or feature a basic GUI using JavaFX or Swing (optional). Furthermore, the game will interact with a database using JDBC to store player progress and high scores. This assignment also provides the opportunity to explore optional features like a difficulty system, advanced game mechanics, and a well-defined scoring system.

## Who is this assignment for?

This assignment serves as a sneak peek into Game Development. If you're interested in that, this is a great way of learning the ins and outs of the concept of Game Development. Who knows, you might even create a game worthy of topping Shin Megami Tensei or even Persona.

Some of you may be dissuaded from taking this assignment due to its complexity. Trust me, this won't be easy, but nothing in life is easy. Take the hardest task and you come out a more skilled programmer. Your future self will thank you for that so that you don't get questioned by employers down the line :p

*"Confusion is part of programming."* - Felienne Hermans

# Basic Features (8 marks)

| Feature | Description | Related Concepts |
|---|---|---|
| Character Creation (1 mark) | Players create a hero with a name (via text input or GUI) and basic stats (HP, Attack Power). The `Hero` class encapsulates hero data and methods like `attack()`, `takeDamage()`, and `displayStats()`. | T8 (Classes), T6 (Java Methods) |
| Dungeon Exploration (2 marks) | A linear dungeon represented by a 1D array containing rooms, monster encounters, and items. Players navigate using commands ("North", "South") or GUI elements (buttons/visual map). Dungeon layout is randomly generated. | T1 (Problem Solving), T5 (Arrays) |
| Monster Battles (2 marks) | Turn-based battles triggered when a player encounters a monster. Players choose to "Attack" or "Flee" (text-based) or select actions via GUI. The battle continues until either the player or monster's HP reaches zero. Battle status is displayed in the console or GUI. | T3 (Selection), T4 (Repetition), T6 (Java Methods) |
| Item Collection (1 marks) | Players find items in the dungeon and store them in an inventory array. The `Item` class represents item properties (name, effect). Methods manage adding and removing items from the inventory, displayed in the console or GUI. | T5 (Arrays), T6 (Java Methods) |
| Game Over/Victory (1 mark) | When the player's HP reaches zero, the game ends with a "Game Over" message. Reaching the end of the dungeon results in | T3 (Selection), T6 (Java Methods) |

| | | |
|---|---|---|
| | victory and a score, displayed with a victory message/screen. The score is calculated based on factors like enemies defeated, time spent, difficulty and optional item-based bonuses. | |
| High Scores (JDBC) (1 mark) | Player names and scores are stored in a database using JDBC. The top 5 scores are retrieved and displayed. | T1 (Problem Solving), T5 (Arrays), T6 (Java Methods), T7 (File Input/Output) |

## Additional Notes for Basic Features

Simple turn-based battle system output:

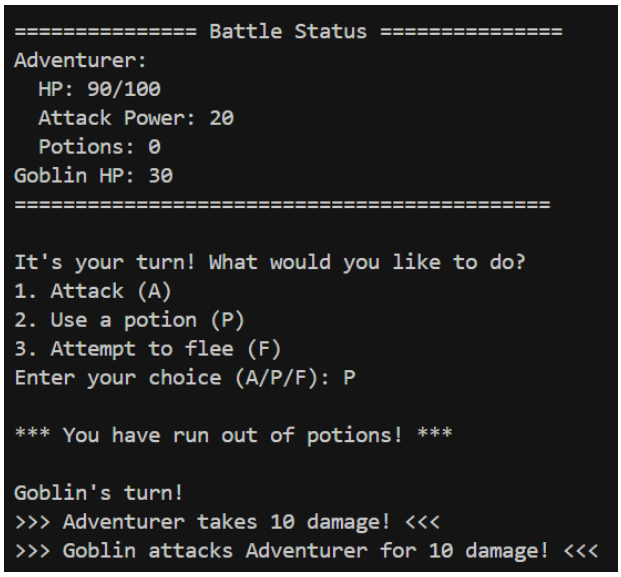


*Figure 1: Sample turn-based battle system*

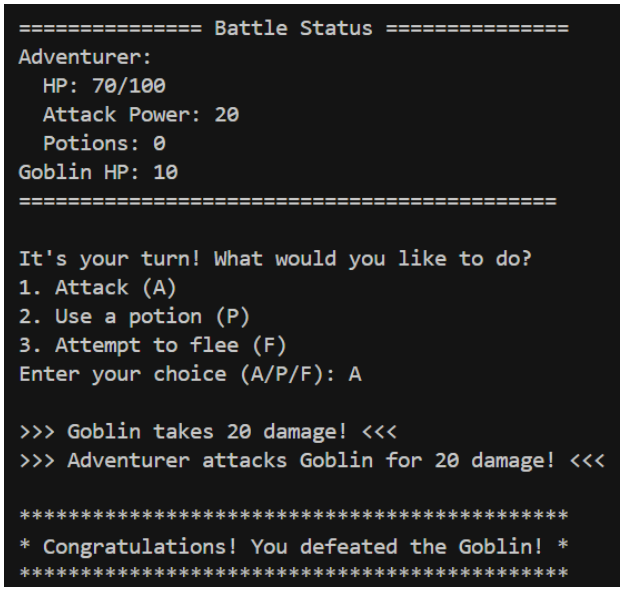


*Figure 2: Sample turn-based battle system*

Simple Character-Creation output:

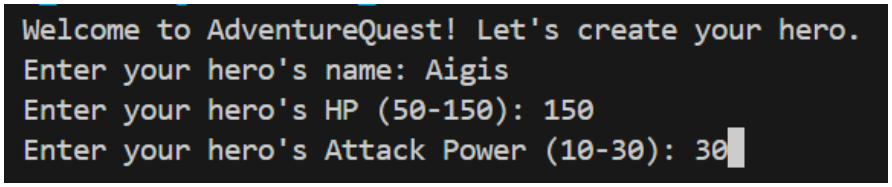


*Figure 3: Sample character creation system*

Simple Dungeon Exploration output:

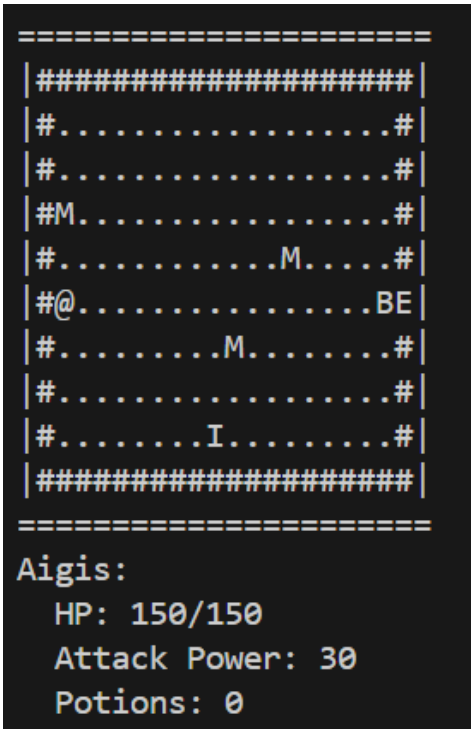


*Figure 4: Sample dungeon exploration. '@' is where the player is, 'M' is where the monster is, 'I' is where the item is, 'B' is where the boss is, 'E' is the exit*

# Database Requirements for Basic Feature “High Scores (JDBC)”

HighScores Table

| Column Name | Data Type | Description |
|---|---|---|
| PlayerName | `VARCHAR(255)` | Stores the player's name. |
| Score | `INT` | Stores the player's score. |

**Score Calculation and Tracking**

The player's score is awarded upon successfully completing the dungeon and is calculated based on the following factors:

- Enemies Defeated: The number of monsters the player defeats contributes significantly to the score.
- Time Spent in Dungeon: A measure of how quickly the player completes the dungeon. Shorter times yield higher scores.
- Difficulty Level: A higher difficulty level leads to a higher score multiplier.
- Items Collected: Potentially, finding and utilizing specific items could grant bonus points.

Note: Implementing only one of these features will mark you as eligible for getting 1 mark for the Basic Feature “High Scores (JDBC)”. Ex: You implement only enemies defeated, hence you are eligible for the 1 mark this basic feature awards.

**Storing Score in JDBC**

After calculating the total score, it should be stored in the HighScores table along with the player's name using a PreparedStatement. This involves creating a SQL query to insert the data into the table.

**Considerations**

- Balance: Ensure the score calculation is balanced to prevent scores from becoming excessively high or low.
- Flexibility: Design the scoring system to be adaptable to changes in game difficulty and challenge levels.
- **Do note that you are not limited to only using JDBC, it is merely an example that is very beginner friendly. If you feel like you want to do MySQL or whatever, please go ahead if you’re up for that task.**

# Advanced Features (4 marks)

## 1. GUI Elements (1 mark)

If implementing a GUI, consider incorporating the following elements:

- **Character Creation:** A form to enter the character's name with potentially visual elements (e.g., hero type image).
- **Dungeon Navigation:** A visual representation of the dungeon with interactive elements (buttons, clickable areas) for movement.
- **Battle UI:** Visually display monster and hero HP, battle actions with buttons, and potential visual effects for attacks.
- **Inventory:** A graphical inventory display with elements like tooltips or item images.
- **Game Over/Victory:** Custom screens to visually represent game states, including the final score.
- **High Scores:** An attractive leaderboard displaying high scores.

## 2. Difficulty System (1 mark)

Enhance the game with a difficulty selection feature:

a. Difficulty Selection: Allow players to choose from Easy, Medium, and Hard difficulties (console input or GUI menu).
   i. Monster Stat Adjustments: Modify monster HP and attack power based on the selected difficulty:
   ii. Easy: Lower HP and attack power.
   iii. Medium: Default HP and attack power.
   iv. Hard: Higher HP and attack power.

b. (Optional) Encounter Frequency: Adjust the frequency of random monster encounters in the dungeon to further differentiate difficulty levels.

## 3. Additional Mechanics (1 mark)

Consider implementing these optional features for extra credit. Choose either of the three below to receive 1 additional mark. This means if you've implemented a basic menu system, as well as multiple monster types, you are only eligible to receive 1 mark.

1. **Multiple Monster Types:** Create different monster types with unique abilities (e.g., slime, goblin, spider). If using GUI, consider unique images/animations for each monster type.

2. **Inventory System with Effects:** Items can affect player stats (e.g., potions heal HP, swords increase attack). Visual representations and effects can be used in the GUI.
3. **Basic Menu System:** Develop a menu system for actions like starting the game, viewing high scores, and exiting. This can be implemented with a menu bar or button panel in a GUI.

## 4. Literally Anything Else (1 mark)

Be as creative as possible, enjoy the process and make sure to highlight CLEARLY that it is an additional feature that is not listed down in this assignment.

# Contact Me

Please contact me at u2000421@siswa.um.edu.my, title your email as "[FOP QUESTION] <Insert your Question here>". Example email title:

[FOP QUESTION] Inquiry regarding difficulty system

If you follow the format of the title above, it will assist me to answer your question ASAP.

For those who have my phone number, please do not attempt to ask questions on my WhatsApp as I receive a lot of messages on a daily basis. If you email me, I can promise you'll get a faster response.

Contact Details:

1) Name: Faris
2) Email: u2000421@siswa.um.edu.my

# Advice

For the love of God, please learn how to use GitHub and Git. Watch this video, your future self will thank you for it. Using GitHub desktop is already passable man, it's not that hard to use.

Here's a neat website if you want to learn about SQL to make your life easier when handling JDBC.

Don't ever compare your output with other students. Learn how to overcome impostor syndrome and you're good to go. A lot of the students in the faculty face impostor syndrome because everyone is literally so smart because you're in Universiti Malaya, the number one university in Malaysia, they like to compare each other. Don't forget that there's nothing wrong with going at a slower pace compared to your peers. Enjoy the learning process as much as you can, because life is all about learning and not who gets to the finish line early. Also don't be surprised if you hear some of your batchmates complete the assignment by Week 6, it happens here in FCSIT.