|  |  |
| --- | --- |
| **Practicum Case** |  |
| COMP6122 | COMP6122001  Framework Layer Architecture |
| **Computer Science** | **O221-COMP6122-CT01-05** |
| ***Valid on*** *Odd Semester Year 2021/2022* | **Revision 00** |

**Learning Outcome**

* LO1 – describe use of design pattern in java
* LO2 – apply design pattern in java

**Topic**

* Session 5 – Structural Design Pattern I

## Sub Topics

* Adapter

1. **Adapter Pattern**

Adapter is a structural design pattern that allows objects with incompatible interfaces to collaborate. You can create an adapter. This is a special object that converts the interface of one object so that another object can understand it.

An adapter wraps one of the objects to hide the complexity of conversion happening behind the scenes. The wrapped object isn’t even aware of the adapter. For example, you can wrap an object that operates in meters and kilometers with an adapter that converts all of the data to imperial units such as feet and miles.

## Soal

*Case*

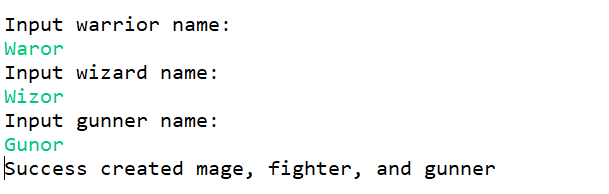
**Angel Adventure**

Angel Adventure is a very simple RPG (Role-Playing Game). Currently, the game only has one character, and the **only one compatible** with the game is **Warrior** (as a fighter). Angel as the senior game engineer of Angel Adventure, she just creates two types of hero, **Gunner** and **Black Mage**. You as the junior game engineer. she asks you to help her to make the Gunner and Black Mage are **compatible** **with the game**, so she can quickly launch the new update for better experiences for the gamers. Below are the following details of the program that has been built:

* **Start Menu**

At first, the program will:

* Ask the user to input the **heroes’ name** for the **Warrior, Black Mage,** and **Gunner**.
* After that, **create** the warrior, black mage, and gunner.
* After created the heroes, the program will **redirect** to the **main menu**.

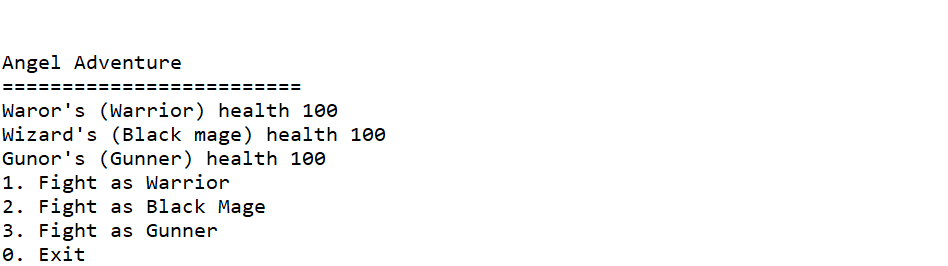


**Figure 1. Start Menu**

* **Main Menu**

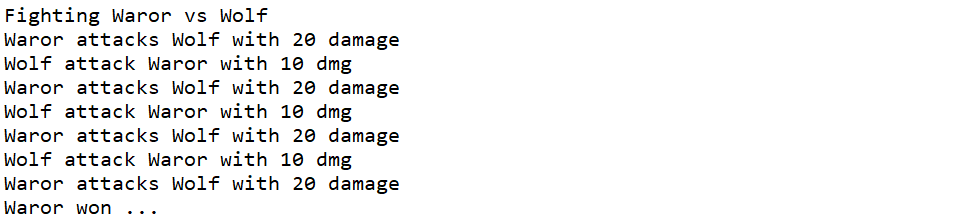
In this menu, the program will:

* Show the **heroes’ health**.
* Ask the user to input the menu **between 0 and 3 (inclusive)**.



**Figure 2. Main Menu**

* If the user **chooses 1** **(Warrior)**, then:
  + If the **warrior’s health** is **above 0**, then:
    - The program will start playing the game **automatically** and fight as a **Warrior** with the monster until he/she wins/losses.



**Figure 3. Fight as Warrior**

* + **Otherwise**, the program will print the **warrior’s dead message**.

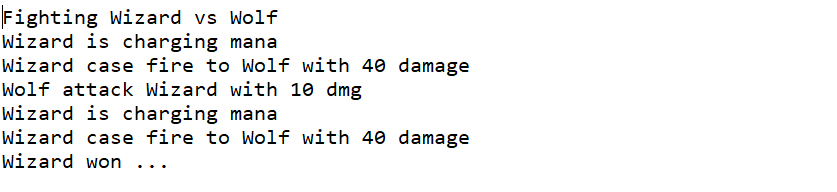


**Figure 4. The Warrior is Dead Message**

* If the user **chooses 0** **(Exit)**, then the program will be **terminated**.

For now, everything that has been mentioned above has already been created by the senior developer before. You don’t need to think about the logic of the game. Because the logic of the game is already created. So right now, things that you must do are **convert the Wizard and Archer to support the game**, below are the criteria you must do:

* If the user **chooses 2 (Black Mage)**, then:
  + If the **mage’s health** is **above 0**, then:
    - The program will start playing the game **automatically** and fight as a **Black Mage** with the monster until he/she wins/losses.
    - As a wizard, when he/she want to attack the monster, he/she will **charge mana first** before he/she attacks.
  + **Otherwise**, the program will print the **black** **mage’s dead message**.



**Figure 5. Fight as Black Mage**



**Figure 6. Black Mage is Dead Message.**

* If the user **chooses 3** (Gunner), then:
  + If the **gunner’s health** is **above 0**, then:
    - The program will start playing the game **automatically** and fight as a **Gunner** with the monster until he/she wins/losses.
  + **Otherwise**, the program will print the **gunner’s dead message**.

Graphical user interface

Description automatically generated with low confidence

**Figure 7. Fight as Gunner**



**Figure 8. Gunner is Dead Message**

**Notes:**

* **Do not change the logic** of the game.
* Please **edit** the **code** that **has** been **provided**.
* You are only allowed to edit the **menu2()** and **menu3()** methods.
* You are allowed to add new classes to create **adapter classes** for the Archer and Wizard.