Questions of Advanced-Programing course at Shahid-Beheshti-University

Exception handler

by

Farbod Fooladi

Legal triangle

We have a class called IllegalTriangleException, which is thrown when we have an illegal triangle. A triangle is illegal when its sides do not apply to the condition of the existence of a triangle. Now we want you to write a program that takes 3 sides of a triangle (integer numbers). And if it was not possible to make a triangle with its sides, it will throw an exception of the mentioned type with the message "You can't make triangle!" written in it.

Advanced calculator

Mehrshad needs a calculator for his math class. Considering that the calculator shops are closed, it asks you to:

Write a calculator program that has addition, subtraction, division, multiplication and division operations.

In addition, for each operator, there should be a method, for example, a method to do addition, etc.

At the beginning, the program should ask the user which of the operations he wants to perform and then, based on that, call the appropriate method and take a number from the user.

tip

Pay attention that we know that dividing by zero and the square root of a negative number does not make sense in the range of real numbers, so handle the relevant exception in the method that performs these operations.

- Use your default java class to divide by zero
- Create an exception class for your own negative root,
- Also, if the user enters a non-numeric string instead of numbers, you should throw the corresponding exception.
- Your program must work with integers.
- Correct this question manually.

Sudoku

You must be familiar with Sudoku game. In this question, we are going to implement a Sudoku with the help of Exceptions.

At first, the game table is empty. At each step, get a coordinate + the desired number of the user from the input. Also, if the showChart command was entered, display the table.

Define (and use!) three Exceptions for when the input number already exists in the row, column, and square of the selected house.

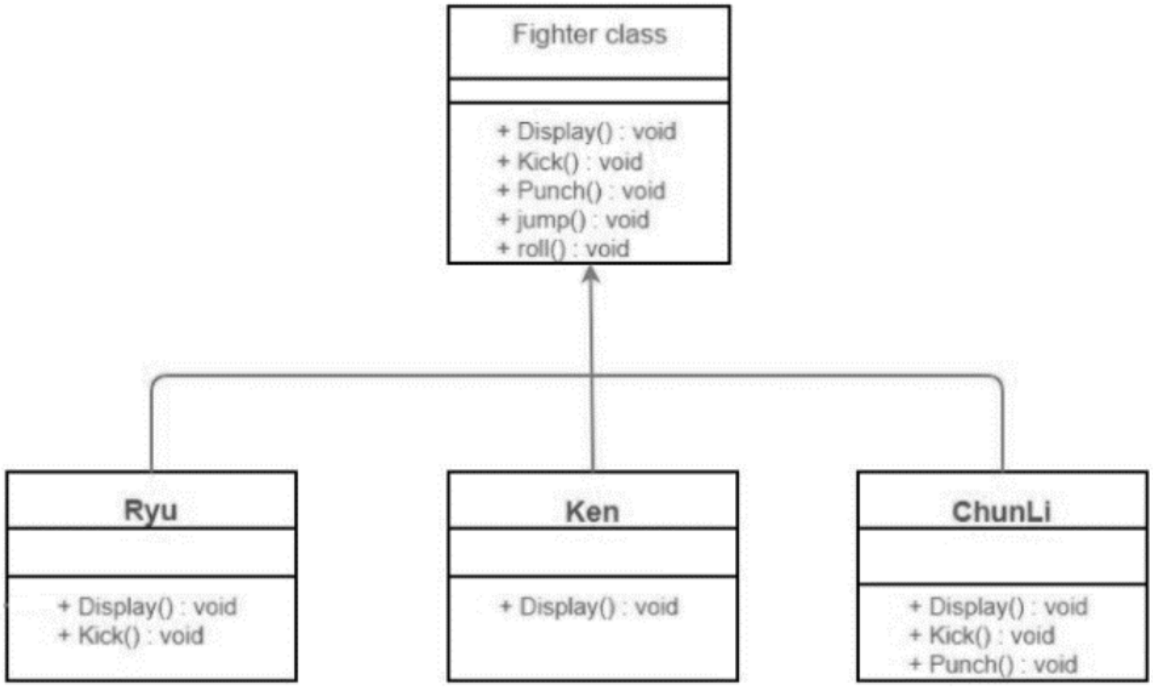
Also, manage other exceptions in other situations (such as number range, table range, wrong input, etc.) with relevant exceptions.

When the game is over, exit the program by showing the appropriate message.

Design pattern

To make better use of her quarantine days, Tara is making a game called Black Warrior. In the initial version of the game, a fighter has four moves: kick, punch, jump, and roll. Tara first uses inheritance and has a Fighter class that other characters inherit from.

Each move can be overridden in the class of one of the replayable characters and have a more specific function.



After a while, Tara realizes that some characters can't jump high, or some don't have the ability to roll, and there are three modes for kick: weak, medium and strong. These changes produce a lot of duplicate code. Therefore, he has to completely change his code and use a suitable design pattern (strategy) for it.

Here's what you need to do to help Tara:

Redesign the main Fighter class and implement its properties or methods in interfaces or other classes in such a way that the following problems are solved:

- As mentioned for kick, there are three modes.
- A character can either jump high or short.
- All characters include display and punch methods.
- A character may or may not have the ability to roll.

Note that all the methods you define must be void, and their body should only include the command to print the class name of that method; For example, if the Roll interface is implemented by two classes CantRoll and CanRoll, we have for CantRoll class:

```
void roll(){

   System.out.println("cantRoll");

}
```

Note that there is no need to implement characters.