Programming Project #1

CIS 2353 - Prof. John P. Baugh – Oakland Community College – OR
Fall 2018

Points: _____ / 100
Due: October 19, 2018 at 11:59 p.m.

Objectives

- To apply knowledge of the `Comparable` interface to a problem
- To understand and override the `toString` and `equals` methods

Instructions

For this assignment, you will create a `GamePiece` class, to be used in a variation of the classic "Rock, Paper, Scissors" game. This variation is called "Rock, Paper, Scissors, Lizard, Spock". A description about how each piece fairs against others is later in this document.

The `GamePiece` class will contain the following fields:

- `classification`
  - The classification of the game piece
  - Valid values are based on an enumerated type, `Piece`:
    - `Rock`
    - `Paper`
    - `Scissors`
    - `Lizard`
    - `Spock`

- `name`
  - The String value of the name, corresponding to the enumerated type
  - "Rock", "Paper", "Scissors", "Lizard", or "Spock" are the valid values.

- `playerName`
  - The name of the person playing (of type String)
  - Can be any name ***under 20 characters long***

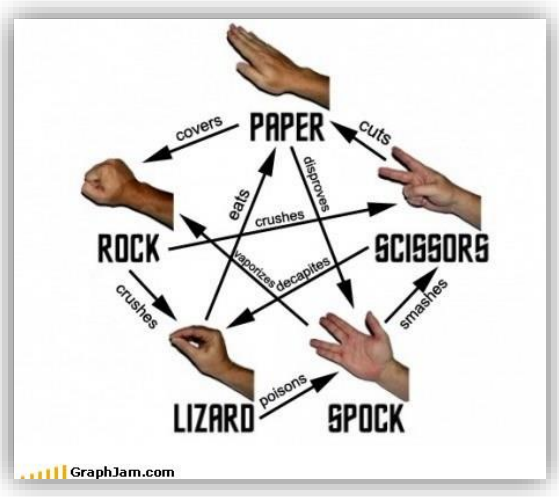
The class will contain the following methods:

- `GamePiece( )`
  - Constructor – initializes the fields to their default values
    - Default classification is set to Rock
    - Default name is set to "Rock"
- `GamePiece (Piece classification, String name, String playerName)`
  - Constructor – initialize the fields to the values passed in by the client
- Getter and setter methods for each of the field names
- `toString( )`

- overridden from the Object class
- This method should return a string containing a reference to the value in the following, depending on the internal Piece value:
  - Rock: “I’m hard and have sharp edges.”
  - Paper: “I’m made of trees and can cover a rock easily.. and disprove Spock!”
  - Scissors: “I’m extra sharp to cut right through paper… and lizards!”
  - Lizard: “Hssssss.”
  - Spock: “It is only logical that you do your CIS 2353 homework.”

- `equals( )`
  - overridden from the Object class
  - A `GamePiece` is “equal” to another `GamePiece` if their classifications match
- `compareTo(GamePiece otherPiece)`
  - implemented from the interface `Comparable`
  - Specifically, `Comparable<GamePiece>` should be used for the interface
  - The method returns the following:
    - -1 is the current game piece object’s classification is less than `otherPiece’s` classification
    - 0 if the game pieces have the same classification
    - 1 if the current game piece object’s classification is greater than `otherPiece’s` classification

Winning/Losing



Piece A	Piece B	Phrase to denote winner	Who wins?
Rock	Paper	Paper covers rock	Paper
Rock	Scissors	Rock crushes scissors	Rock
Rock	Lizard	Rock crushes lizard	Rock
Rock	Spock	Spock vaporizes rock	Spock
Paper	Scissors	Scissors cuts paper	Scissors
Paper	Lizard	Lizard eats paper	Lizard
Paper	Spock	Paper disproves Spock	Paper
Scissors	Lizard	Scissors decapitates lizard	Scissors
Scissors	Spock	Spock smashes scissors	Spock
Spock	Lizard	Lizard poisons Spock	Lizard

Obviously, if the two pieces are the same, it's a tie!

Deliverables

Turn in a **zipped up folder**, containing the Java file(s) required for this project. This includes all relevant classes that are created, as well as the client used to test the class(es) – i.e., the class containing the main method. Upload them to D2L under Assignments, to the appropriate assignment directory.