Programming on Purpose (CSC 120)
Winter 2024

Project 4 – A computer player for Igel Ärgern

In this project, you will implement a computer player for the "Igel Ärgern" game.

1 Objectives

- More practice with Java, as well as OOP and other good software development practices.
- Practice working with interfaces.

2 Due Dates

Milestone 1: Monday, 3/11, before class A working implementation of your "Igel Ärgern" player

Milestone 2: Tuesday, 3/12, end of day The final version of your code

3 Your Mission

You will implement a class that implements the `igel.game.Player` interface and which provides a computer player for the Igel Ärgern game.

4 Starter code

Once you unzip the starter code, you will have a folder called `igel` which contains the subfolders `game`, `humanplayer`, and `kristinaplayer`. The `igel` folder also contains the files `Driver.java` and `README`.

- `game`: This folder contains the game logic for the Igel Ärgern game. This is what you have been implementing in Projects 1–3. Files in this class that you should look at are `Player.java`, `ReadOnlyCell.java`, and `Move.java`.
- `humanplayer`: This folder contains the class `HumanPlayer`, which is an implementation of the `Player` interface and creates an Igel Ärgern player that allows human players to play the game by providing input.
- `kristinaplayer`: This is a dummy implementation of a computer player for the Igel Ärgern game. This implementation does not work as a player, but it serves as an example of how you should set up your own package. See the `README` file for some additional instructions.
- `Driver.java`: This class contains a main method. Run it. It allows you to play an Igel Ärgern game with four human players. Read the comments in this file to see how to replace (some of) the human players with computer players.

5 Requirements

1. You must submit a Java package called `igel.XXXplayer`, where `XXX` is your first name. E.g. in my case the package would be called `igel.kristinaplayer`.
2. The package has to contain a class called `ComputerPlayer`. It may contain other classes if you need them.
3. `ComputerPlayer` has to implement the `igel.game.Player` interface.
4. The constructor for the `ComputerPlayer` has to have the following signature:

```
public ComputerPlayer(IgelView view)
```

5. The `ComputerPlayer` has to be able to play a game of Igel Ärgern without any input by a human player.

6 Submitting Milestone 1

Compress the subfolder that contains your computer player implementation (i.e. the subfolder that has a name of the form `XXXplayer` where `XXX` is your name) into a .zip file and upload it to **Nexus** by **Monday, 3/11, 2pm.**

It is important that this implementation runs without breaking so that it can participate in an Igel Ärgern tournament in class on Monday.

7 Submitting Milestone 2

Further, test and debug (if necessary), and refactor your implementation.

Remember: *Just because it works doesn't mean it's good.*

- Make sure that all classes and public methods are documented.
- Re-read your code and check for ways in which readability could be improved (e.g., good variable/function names, no magic numbers, consistent formatting, code is organized into logical units, each method serves one main purpose).
- Don't forget to cite who you received help from and include the honor code affirmation at the top of each Java file you have written or edited:

```
I affirm that I have carried out my academic endeavors
with full academic honesty. [Your Name]
```

Compress the subfolder that contains your computer player implementation (i.e. the subfolder that has a name of the form `XXXplayer` where `XXX` is your name) into a .zip file and upload it to **Gradescope** by **Tuesday, 3/12, end of day.**