CSS 605
Game Theory Competition

Due: October 21, 2014

For the assignment, you are to write the behavior of an agent that is playing three games. Your agent class should extend the `AbstractGamePlayer` class. This means that you only need to implement one method (`getGameStrategy`). The method will only return one of two values: `GameStrategy.STRATEGY1` or `GameStrategy.STRATEGY2`. The method header is:

```
public GameStrategy getGameStrategy(UUID opponentId,
                                    GamePayoffs gamePayoffs,
                                    GameHistory gameHistory)
```

To summarize the method inputs:

`opponentId` - a unique identifier for the opponent for this game

`gamePayoffs` - the payoffs for the game currently being played.

`gameHistory` - the history of all of the games played so far in this round. There are methods that provide access to the strategies played by your opponent for all games, for the last time they played a specified game, and the last time they played you. Furthermore, information about the number of games remaining and the current total payoff for the other players may be helpful in determining what strategy to play.

The code documentation provides much more information about the various classes used for the assignment.

Game Description

Each game played will be randomly selected from the following three games:

	Opponent Strategy	
	S1	S2
S1	1,1	-1,2
S2	2,-1	0,0

Prisoner's Dilemma
S1 = COOPERATE
S2 = DEFECT

	Opponent Strategy	
	S1	S2
S1	1,1	0,.6
S2	.6,0	.4,.4

Stag Hunt
S1 = STAG
S2 = HARE

	Opponent Strategy	
	S1	S2
S1	1,1	0,2
S2	2,0	-1,-1

Hawk Dove (Chicken)
S1 = SWERVE
S2 = STRAIGHT

For each play of a game, your payoff will be determined by what strategy you choose and what strategy your opponent chooses. Also, to make your code more readable, `GameStrategy` includes aliases for `STRATEGY1` and `STRATEGY2` that can be used instead (aliases included below payoff tables).

Competition Description

At the beginning of the competition, two instances of each `GamePlayer` will be created. During each round, a large number of games will be played against randomly assigned opponents. At the end of each round, two players will be eliminated and another round will begin. This will continue until the competition is down to the last two players. If you want to run the competition yourself, you can include your player in the game_players.txt. There are two simple implementations in source code that you can use as a starting point for your implementation (`Strategy1Player` and `Strategy2Player`).

What to Submit

Submit your ***well-documented*** Java class file that contains your implementation of a game player. This file should be in a package named something like:
`edu.gmu.<your username>.competition.gametheory`.

Also, submit a display name for your player that will be used when displaying the results of the competition in class.

Week 9 Assignment

Due: October 28, 2014
(Competition submission due: October 21, 2014)

Write a paragraph about how you approached the game theory competition and what your code does. Avoid discussing what you wanted your code to do and focus on what your code actually does. This assignment gives you practice in communicating your code's behavior into words. This description may only be a few sentences, but should be no more than one page.