Q&A about LP2A « Llama Card Game » project

The goal of this document is to answer a few questions that might emerge after reading the rules of the « Llama card game » that you are supposed to code in Java as a project.

In particular, the multiplayer aspect of the game and the limited experience you have with the Swing library are the most likely to raise questions. So we decided to provide you with some answers on those aspects.

A) Llama game is clearly multiplayer. How are we meant to handle this ?

In the version you have to do, there is only 1 human player. The other (or others) are bots with a very basic behaviour.

You can go for :

- 1 human player VS 1 bot *(easier to make a good UI)*
- 1 human player VS 3 bots *(harder, but makes game more interesting)*
- Give user the choice of 1 or 3 opponents *(for those who wnat a challenge!)*

Only 1v1 is required. The other options are increasingly difficult because it requires a more complex UI. But you will be rewarded in points for achieving those more advanced versions.

We don't ask for you to make some kind of advanced A.I. for the bots : we just want them to follow a simple algorithm based on if/then/else statements, simple loops, and a tiny bit of randomness.

Here is a description of this algorithm/behaviour :

1. On their turn, bots check if they have a card in their hand that has the same number as the one on the « play pile ». If they have one, they play it.

2. If step 1 has not resulted in a play, bots check if they have a card « one above » the number on the play pile. If they do, they play it.
   Reminder : according to the rules, a card « above 6 » is a « Llama ». And the card « above Llama » is a 1.

3. **If bots have no playable card, they randomly decide to draw a card or to « quit », locking in their cards until end of round.**
   The probability of quitting should increase at end of turn, and more so if the « total value of cards in hands is low». Probability to quit should also increase « every time the bot can not play a card and had to draw ».
   You can integrate other factors and make this behaviour more clever if you want to. And if you can figure out an algorithm that makes the bots better at the game by doing so, **you will be rewarded for it.**

4. **If the bot didn't « quit », they draw a card and their turn end** (just like the player would in this case)

5. **If the bot « quitted », it can no longer play any cards and skips its turn until the round ends**. (again, just like player would in this case)

Note that bots, by following this simple logic at the speed of the computer, are able to play « instantly ». **So fast, in fact, that the player will have no idea what's going on !**

Try to find a way to delay the « addition of their card played to the pile », in real time, so the player can understand what's going on.

To put you on track for this :
Link to a Stack Overflow question/answers about « waiting in Java »

B) What can the player actually do on their turn ? Could you clarify ?

It depends on what is in their hand, and what is the card on top of the « play pile ». But, it boils down to :

- Playing an acceptable card and ending turn
  (acceptable card = same value than card on top of play pile, or « one above », with Llama being above 6 and 1 being above Llama)
- Drawing a card and ending turn
- « Quitting », which locks the cards in the player's hand. Player has to wait for bots to finish playing so « round score » can be calculated.

You can see that these are the same options that the bots have. The difference being that the human player will have to click buttons in the UI to make their choices, instead of deciding with a robotic, automatic algorithm.

Try to use various indicators on your GUI in order to show information on what player is currently playing, what players have « quit », and everything else you think is relevant to the game.

C) How does that scoring work ? The rules talk about score tokens, do I have to display those on the screen ?

You do **NOT** have to display « tokens » on the screen. A simple label text saying « Score : X », with X being the score number for that player, will do just fine.

Reminder : just like in Golf, the lowest score is the winner. Don't forget that if you want to display a crown next to the person winning, show a end game congratulations message, or something like that.

In the « real life version » of the game, palyer keep tracks of their score between rounds with score tokens. There are « 1 point » and « 10 point » tokens.

The rules say that, when a player ends the round with a « perfect » (emptying his hand completely), they get to remove one of their score tokens.
And because of how score tokens work, it means a player with 10 points or more can discard a higher value « 10 point token », when someone with less points will only be able to discard a « 1 point token ».

This creates a « comeback mechanic », allowing a player with higher score to reduce it more easily.
You can simply emulate this behaviour with an integer number score, though :

- If the player has 10 points or more when achieving a perfect, their score decreses by 10 points.
- If the player has less than 10 points, his score is only reduced by one for achieving a perfect.

While we're on the topic : a round is over once all the players have either « perfected » or quitted. A player who « perfected » stops playing until end of round, just as a player who quitted does.

When the round is over, scores are updated according to the cards left in hand (or lack thereof = perfect), a popup window appears, showing the rankings
= displaying the scores in order from lowest to highest, showing « who's winning ».

This score pop up window has two buttons :

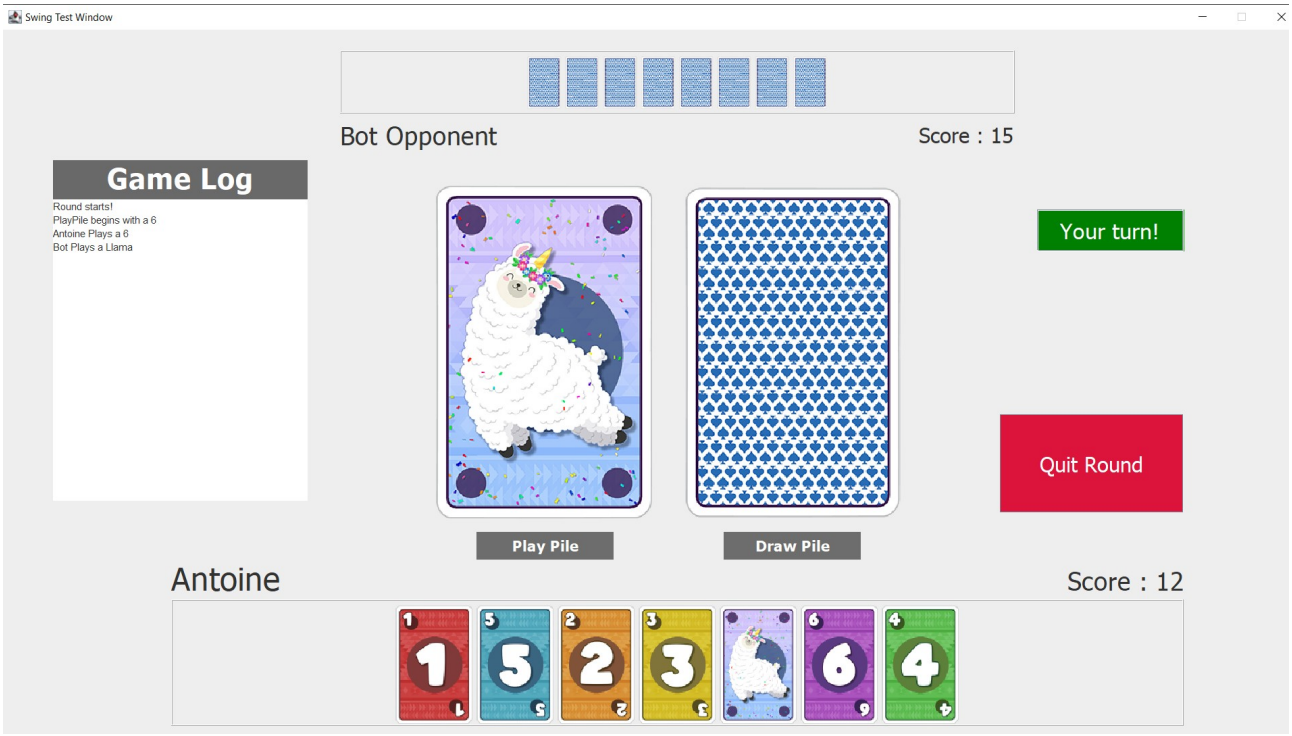
- **« Another round ! »**, which restarts a new round, but keeps the scores at their current state.
- **« Stop playing »** which closes the application.

D) How can I do a good graphic interface for this ? I don’t have much experience with Swing, making my toolbox very limited !

We allow you to use « Absolute Layout »for the main window of the app.
While this will not allow the window to be resized without looking like crap, this is the simplest type of layout to apprehend, as items can be freely placed through their x and y coordinates.
Since resizing won’t work well, please design the window at an appropriate for a typical laptop screen (1080p = 1920x1080). Using thos referential, make the window use most of the screen, but not all of it (a window of size around 1600x900 would probably do nicely).

You will have to use other layouts for « elements inside the window », though. **In particular, for the areas displaying the player’s hands.**
Because the number of cards in hand is variable, you should put them inside a Jpanel that automatically displays items next to one another, like for instance a **« Flow Layout »**

Here is a quick mockup that should give you an idea for a simple interface that is suitable to play the game, without being too complicated to implement either :



This mockup isn’t high art, but it should give you a decent idea of how « Llama » can be implemented in Swing, even with limited experience.

If you can design an interface that is prettier and/or more ergonomic than this, you will be rewarded in points !

> **Tip :** you can use pop-ups and secondary windows in order to convey information, and make your software more enjoyable to use !

For people who want to implement « 1 VS 3 » :

Because of limited screen space, it can be hard to display 4 players at once. **So if you implement more than one opponent in your Llama game, you might have to « swap the bot player displayed in the upper part», instead of dispalying everyone at once.**

« Swapping the player will mean » replacing all the data displayed there so it matches the bot currently playing (name, score, cards in hand, etc.).
The human player, however, should never be « switched out ». only his bots opponents.

To increase clarity and allow the player to process what is going on, **try to use extra visual indicators like a background color in order to emphasize the « swaps » and what opponent is currently playing.**
Using « wait time » to allow the human player to better process the sudden changes in display is also a good idea : after a swap, « wait » a little bit before the freshly displayed opponent plays their card. It will help avoid confusion.

If you are not confident in your ability to do this « opponent swapping » thing, remember than having more than one bot opponent is a « nice to have ».
It’s not worth many points at all, and you shouldn’t bother with this feature if you’re struggling a bit with the new tool ! But I am giving this information for the people who want to try this challenge and aim for the highest possible grade.

GOOD LUCK !