CS102 – Algorithms and Programming II
Homework 2 - Simplified Okey Game
Spring 2024

This is group homework. You will complete this homework with your project group and submit one solution. Make sure you collaborate and divide the work equally as much as possible. Include a text file “Members.txt” inside your submission that includes your group member names and their IDs. For the group submission, it is enough for one member to upload it on Moodle.

You will also write a “Reflections.txt” where you reflect on the homework using at least 100 words: Was it easy to work on a partial implementation? Were there any difficulties working as a team? You will submit this file under the individual submission page, all members should complete reflections individually.

For this homework, you will complete the partial implementation of a simplified Okey game. For the complete set of rules of the game, you can check the following page:

https://en.wikipedia.org/wiki/Okey

Our version of the game will be a simplified console application. The following rules are altered to make the game simpler:

- There are no jokers, so we do not reveal a joker at the beginning of the game, and no tile can be a substitute for a different one.
- There are no false joker tiles and different tile colors, so instead of 106 tiles in the game, we only have 104 tiles; however, the range of the numbers is increased to [1,26]. That is 4 sets of 26 numbers.
- Any player with a complete chain of 14 consecutive numbers wins. If there are no tiles on the ground to draw, the game ends, and the player with the longest chain wins.
- No circular continuity of the numbers. You cannot have 25-26-1-2 as a chain, these would contribute to separate chains.
- The starting player is predetermined. The player at index 0 starts first.

The game will be single-player, played against 3 computer opponents. The human player starts first. The application will display the current tiles of the player and then will display the options to get the user’s choice. The high-level logic of the game is already included in the given partial implementation. You should complete any code marked with TODO based on the explanations in the comments. You can add new classes or methods if you think necessary. You can modify the existing code or fix their mistakes if there are any. Don’t forget that the partial implementation is there to guide and help you, but if it is not useful, you may shape it according to your design.

In the Okey game, players take turns to complete their tiles into a winning hand. At each turn, the player may pick a tile from the stack or the tile discarded by the previous player. The first starting player does not pick a tile.

At the beginning, the stack of tiles is shuffled, and each player is given 14 tiles except for the first player, who gets 15 tiles. A tile should be discarded without drawing a new one in the first player's first turn. Then, in any of the turns, each player has two options to draw a tile from: the tile discarded by the previous player or a tile from the tile stack. The discarded tile is visible to all players to strategize; the tiles in the stack are not visible unless a player draws it, and in that case, it is only visible to the player who draws it. The player whose turn now picks a tile to make their hand 15 tiles and then discards one of the least useful tiles.

A winning hand should have 14 consecutive numbers in our version. For example, the following hands are winning hands:

Ex1: 3-4-5-6-7-8-9-10-11-12-13-14-15-16-19

Ex2: 11-12-13-14-15-16-17-17-18-19-20-21-22-23-24

Note that the player whose turn is now draws one extra tile to have 15 tiles in hand, so the extra tile does not disturb the chain. For example, having a duplicate 17 does not disturb the chain of 14 consecutive numbers in the second example.

The existing classes of the partial implementation are as follows:

- ApplicationMain.java: Includes the main method that initializes and processes the game. Until the game finishes, the main loop receives input on the player's turn.
- OkeyGame.java: Includes the high-level logic of the game. Contains 4 players, and the tiles stack. You will complete the methods such as shuffling the tiles, handling the computer's turn, and checking the win condition inside this class.
- Player.java: Includes the logic for one player. It contains the player's tiles. Methods such as adding or removing a tile from that player's hand, displaying the player's tiles, and finding the longest chain are included in this class.
- Tile.java: Represents a tile with a specific value. Methods to compare and print a tile are included in this class.

The requirements for all the methods are included in the partial implementation.

Note that you should keep the tiles sorted in ascending order to check the win condition easily. You should make the add tile method of the player to insert new tiles in order so that each player's hand is always sorted without using a sorting algorithm.

For example, suppose our hand is:

3-6-6-7-8-8-9-15-15-16-17-18-19-22

If we draw 10, this should be inserted into the correct position to keep the order. This requires you to loop over the existing tiles to find the correct position, then shift the remaining tiles to the right by one:

3-6-6-7-8-8-9-10-15-15-16-17-18-19-22

The computer players should decide whether to draw from the stack or the tile discarded by the previous player following an algorithm. Here, you should check if drawing the discarded tile would increase the longest chain length of the computer player; if so, the computer should draw the discarded tile; otherwise, the computers should draw from the tile stack. The computer should also decide on the tile to discard considering the tile chains. If there is a duplicate tile, discarding it first is the best choice. Then, discarding the tiles contributing to the shortest chain would be meaningful. You can implement different strategies for determining the tile computer player discards. Try to come up with interesting algorithms.

Your program will also include an open-hand setting to see the computer players’ hands for debugging. If the game ends when a player completes a chain of 14, you will display that player as the winner. If the game ends with no tiles in the tile stack, display each player's longest chain and declare the winner with the highest amount.