CS51 Project: Checkpoint

Project name: Japanese Mahjong Hand Helper
Name: Mason Tan
Email address: mtan810@gmail.com

Progress

A lot has changed over the past week, including the signatures/interfaces. Initially, there were two modules that would interact with each other: the tile module and the hand module. Since I defined hand as an array of tiles, I integrated the tile module into the hand module and have one whole module instead.

I redefined tile, which was initially a tuple (num * suit), to a record instead with type num, suit, honor, and a new type flag. Because a tile is either a num+suit tile or an honor tile, 'honor' for num+suit tiles will be 'Nope' and 'num' and 'suit' for honor tiles will be '10' and 'Hon'. The flag type will be used as a marker for the backtracking algorithm, which will be explained later.

The sorting algorithm is completed. It now sorts all the different types of tiles in the order from left to right: 1-9 characters, 1-9 circles, 1-9 bamboos, East, South, West, North, White, Green, and Red.

The backtracking algorithm is close to completion. The 'backtrack' function is a recursive function that sorts the hand and calls two track functions: 'track_pair' and 'track_set'. 'track_pair' marks the first set of pairs that it finds. 'track_set' marks the remaining tiles that form either a triple or a sequence. 'backtrack' verifies whether all the tiles in the hand have been marked or not. If so, then it is a winning hand. If not, then it retries by unmarking all the tiles first. However, the difference is that it will find the next pair instead. The algorithm continues until it either finds a winning hand or goes through the entire hand.

Problems

Most of my problems from last week have been resolved as the project progressed. Because the hand module is much more defined and complete, the functions are much more solvable. The idea of redefining a tile with records relieves many issues I had when using tuples.

My main concern is whether or not my backtracking algorithm works for all kinds of hand. It seems like the only way to find out is to come up with special hands that can win or lose based on how the pair/triples/sequences are formed. Thus more testing is necessary to confirm.

Teamwork

I divided parts of the module to me, me, and me. The problem is that me, me, and me are actually all me!

Plan

List of stuff that needs to be done:

- Complete backtracking algorithm
- Finish draw/discard algorithms
- Tie everything together and create the mahjong program that uses hand module
- Better UI if time permits