

**TP3 DESIGN UPDATES:** Didn't end up using sockets, just went with Local Multiplayer and smart AI.

### **Mahjong: 15-112 term project**

Project Proposal ( *The name of the term project and a short description of what it will be*):

My 15-112 term project is titled Mahjong, and is a version of the the popular Chinese tile game. Mahjong is played with 4 players each who possess 14 tiles. The ultimate goal of the game is for a player to achieve a winning hand (4 sets[which can be made up from "Peng" or "Chi" moves] and a pair; more information about how to have a winning hand can be found online. Players work towards the goal of a "Hu"(a winning hand) by placing tiles from their hand in the center of the board, and taking a tile (either from the center of the board or from a pile) in order to reach the "Hu" state. The game can either be played with AI or local multiplayer; with AI, there only needs to be a single user. The other three players will be represented as CP1/CP2/CP3 and will have some semblance of intelligence in making moves. With local multiplayer, four users are needed. The computer will need to be passed around to each player. The game will be able to progress and make moves on the behalf of each player with the press of an arrow key. When a user achieves a winning hand, it will automatically be registered by the game and a Game Over screen will pop up, and will list the name of the winning player and the point total of the hand.

**Competitive Analysis** [5 pts]: *A 1-2 paragraph analysis of similar projects you've seen online, and how your project will be similar or different to those.*

Other projects of the game Mahjong that I have seen online mostly involve different versions of Mahjong(ranging from Australian to Japanese rules) that progress through AI or server based multiplayer with sockets. Lots of Mahjong games online have their own sets of rules that include having time limits for each round, a running counter of the pieces that have been removed, and even restrictions on when one can "peng". My game will be following the standard Americanized rulebook of Mahjong, which can largely be found online.

In addition, most Mahjong projects that I have seen through the 112 Gallery and Github use Pygame for the aesthetics of the game. My game is similar to those online in that it has the option of playing with AI; however, it's different in that it also offers up the option of playing with local multiplayer(as opposed to usage of sockets). My game is also completely constructed with Tkinter and will not be using Pygame at all for aesthetics.

**Structural Plan** [5 pts]: *A structural plan for how the finalized project will be organized in different functions, files and/or objects.*

My project will be organized into several different parts, the key parts being the Start/Help Screens, Hand Actions, Rule Functions/Objects, and different modes.

**Start/Help Screens:** The Start and Help screens can be accessed by pressing specific letters on the keyboard. The help screen will just display rules on how to play the game, while the start screen will transition into the actual play game mode.

**Hand Actions:** Hand actions include highlighting/selecting various tiles, being able to send tiles into the middle of the board, and being able to pick a tile from the middle of the board and add it into the user's existing hand. I plan on writing helper functions that will be called in the event of mouse clicks/keypresses in order to conduct said actions.

**Rule Functions/Objects:** Each hand of the players will be their own subclass that will be under the parent class Hand. Hand will have the methods "isHu", "isPeng", and "isChi", that will detect when a hand is able to perform any of those actions and will then alert the user that that action can be taken.

**Different Modes:** At the beginning of the game, the user can choose to either play in AI mode or local multiplayer mode. I will essentially have two different versions of game play that will only be accessed if the player chooses a certain option.

**Algorithmic Plan** [5 pts]: *A detailed algorithmic plan for how you will approach the trickiest part of the project.*

I foresee the trickiest part of my project being the AI/local multiplayer portion. As stated before, I will essentially have two different versions of game play that will only be accessed if the player chooses a certain option. For example, in my init, I might have `data.AI = False` and `data.local = False`. If the user chooses to play with local multiplayer, that would set `data.local` to `True` and run all of the functions associated with playing local multiplayer. The functions associated with using AI would not be run in that gameplay and vice versa.

For local multiplayer, I plan on cycling through each player's turn by pressing an arrow key. Once the arrow key is pressed, it is now another player's turn and his/her tiles will show up on screen. The player will then make his/her move. Once he/she is finished, they can press the arrow key and it will be the next user's move. No other players' hands can be modified by other users during a turn; a user can only modify his/her own hand and see his/her own hand.

For AI, I want to make CP's that have a little bit of knowledge. For example, the CP's will Peng/Chi when alerted to do so; however, they will still put out random tiles at every turn. I

plan on randomizing (`random.randint`) the tiles that they put out into the center of the board from their hand.

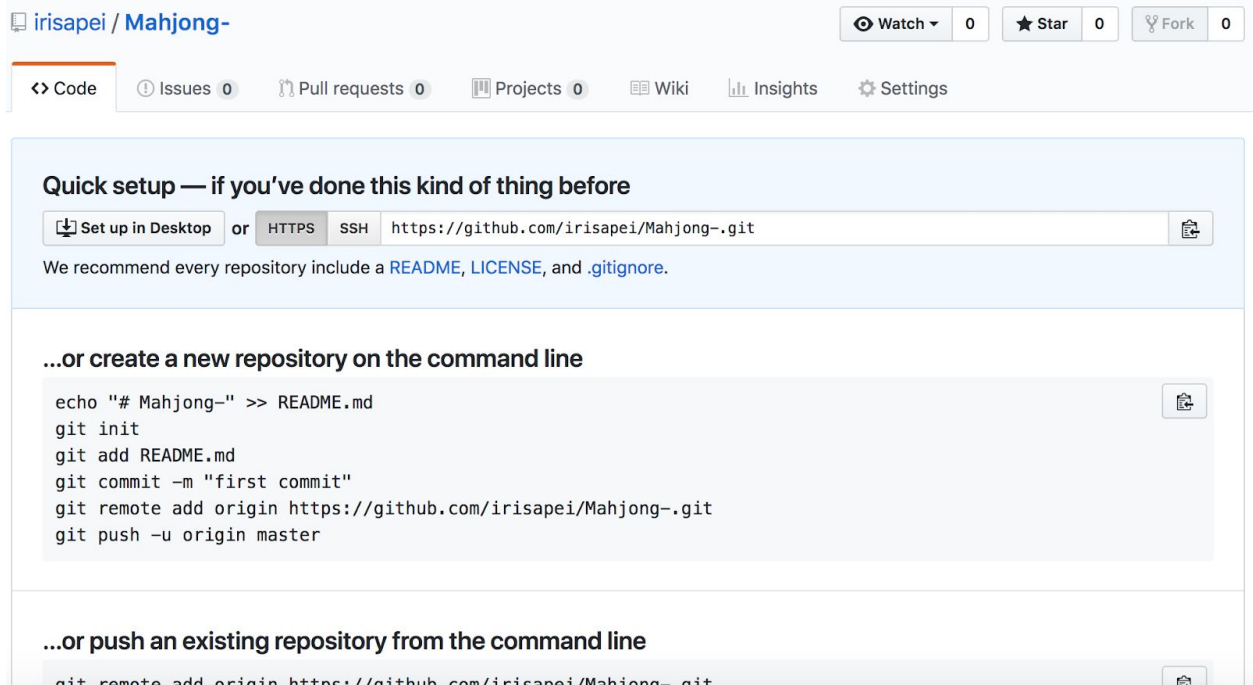
For both multiplayer and AI, I will be storing the tile data in separate lists that will be destructively modified given the action of the user. For example, If a user chooses to send a tile out to the middle of the board, that tile would be popped from the list and would thus not appear in the user's hand on screen anymore. By keeping the tiles in a list, I will also be able to call `isPeng/isChi/isHu` on them.

**Timeline Plan** [5 pts]: *A timeline for when you intend to complete the major features of the project.*

By TP1, I want to be able to have drawn my board with 14 randomly chosen tiles, switch through each players' tiles with the press of an arrow key, and have some range of control over the tiles (i.e highlighting them, moving them around). I also want to get started on the Hand Class and the methods `isPeng` and `isChi`. By Friday of this week (April 20th), I want to have a janky but working version of the game in local multiplayer. By TP2 (April 25th), I want to have both basic AI and local multiplayer versions of the game up and running and a slightly better UI/UX design. By TP3, my project will be completely done.

**Version Control Plan** [3 pts]: *A short description and image demonstrating how you are using version control to back up your code. **You must back up your code somehow!!!***

I'll be uploading my updated code to GitHub everyday!



The screenshot shows the GitHub interface for a repository named 'Mahjong-' by user 'irisapei'. At the top, there are buttons for 'Watch' (0), 'Star' (0), and 'Fork' (0). Below these are tabs for 'Code', 'Issues' (0), 'Pull requests' (0), 'Projects' (0), 'Wiki', 'Insights', and 'Settings'. The main content area has a light blue header with the text 'Quick setup — if you've done this kind of thing before'. Below this header, there are three tabs: 'Set up in Desktop', 'HTTPS', and 'SSH'. The 'SSH' tab is selected, showing the URL 'https://github.com/irisapei/Mahjong-.git'. Below the tabs, there is a recommendation: 'We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).' The next section is titled '...or create a new repository on the command line' and contains a code block with the following commands: 

```
echo "# Mahjong-" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/irisapei/Mahjong-.git
git push -u origin master
```

 The final section is titled '...or push an existing repository from the command line' and contains a code block with the command: 

```
git remote add origin https://github.com/irisapei/Mahjong-.git
```

**Module List** [2 pts]: *A list of all external modules/hardware/technologies you are planning to use in your project. Note that any such modules must be approved by a tech demo.*

Currently not using any external modules/hardware/tech!