If you want an overview of Wits and Wagers, click on the following link:

http://www.annarbor.com/entertainment/party-games-that-do-something-other-than-end-the-party/

Canopy Files

What to Update:

Each team will update their values in the teamx.py files, the instructor will update the witsnwagers.py file.

Managing the Game:

Playing the game requires changing the game state variables as teams respond to questions and place their bets. To start the game, or set the game to the initial state (see below), set the game_state variable to 's', run the application, commit and sync, then have the teams sync. **Restart the kernel in Python after syncing**.

Setting the game to the initial state is not necessary if each team has cloned the repository from the initial state.

```
13 #import team data
14 import team1
15 import team2
16 import team3
17 import team4
18 import team5
19 import team6
20 import team7
21
22 # ******* Remember to restart kernel to update memory after syncing *******
23 #---------------------- GAME PLAY INPUTS ---------------------------------
24 # game state values: 's', start-game; 'a', after teams answer; 'b', after teams bet
25 game_state = 's'
26 # answer/bet marker position 1 - 8
27 answer = 0
28 # increment each time team place bets, when game_state = 'b'
29 play_round = 0
30 # each time bets are placed, update all three inputs
31 #-------------------------------------------------------------------------
```

1) Pose a question that requires a numerical answer, have each team update their teamx.py file with their answer (answer_question_amt, see below), save their file, then commit and sync. Each team only syncs their teamx.py file, not the other files that are changed. I also required teams to provide a description when they commit, like "Team 1, Answer 1" so I could see when all teams responded on GitHub.

```
9
10      #update this variable in responding to questions
11      self.answer_question_amt = 1
12
13      #update these variables in placing your bet
14      self.bet_position = 1
15      self.bet_amount = 1
16
```

After each team has provided an answer, change the game_state variable to 'a', save the file, commit and sync, then have the teams sync to pick up the changes. **After syncing, everyone will have to restart the kernel before they run the application**. After running the application, everyone will see the answers teams provided.

2) Next, have teams place their bets, choosing one of the bet positions 1-8. They update the bet_position and bet_amount values on the teamx.py file, save their file, then commit and sync. Again, I required that they only sync their teamx.py file. I also required they provide a description in their commit, like "Team 1, Bet 1", so I could see when all teams responded on GitHub.

After each team has betted, change the game_state variable to 'b', change the answer to the correct numerical answer, and increment play_round by 1, save the file, then commit and sync, then have the teams sync to pick up the changes. **As before, after syncing, everyone will have to restart the kernel before they run the application**. After running the application, everyone will see the results of betting.

Continue to do steps 1-2 until the game is complete**. Everyone must restart the kernel after syncing, and it is very important that the instructor does this to update the answers/bets/scores, etc. for each round**.