I have struggled to flowchart and conceptualize workflows in the abstract because I am so new to this. O have been figuring out everything best by trial and error! This was my general planning process. I know it doesn't 100% adhere to conventions, but it helped me work through the project really well and conceptualize what was going on with the game as a whole!

First, I wrote the general rules so that I could ensure I understood the game and start to conceptualize it in terms of Python:

(JENEVAL TIME
General Flow
- ROUND ONE
starting player chosen randomly starting player spin wheel -IF bankrupt, lose & & turn: NEXT -IF lose turn like turn: NEXT
7 Starting player spins wheel
-IF Bankrupt, lose & & turn: NEXT
-If lose turn, lose turn: NEXT
-IF lose turn, lose turn: NEXT -IF (I), quest consonant (or vowel past turn 1) IF consonant not in word: NEXT IF consonant in word, replace blanks, apt \$
. If consonant not in word: NEXT
IF consonant in word replace blanks.
get \$
· round over when guess == word · winner's s stored in prize pot
· Winner's S stored in prize not
loser's I goes away
- ROUND TWO. goes away
· all banks => 0
· random starting player
· same process as round one
· same process as I round one . store as the same or dif?
- ROUND THREE.
- planer == person whost t
· moulate R.S.T. LINE
· starter autifice 3 concornants I record from
· don'au doem't change?
populate R.S.T. LINE player gusses 3 consonants, I vowel free display doesn't change? 5 sec to guess timal answer (one guess)
· if correct is cash prize
The correct of them prive
- 2

Then, I started to write out pseudo code in my notebook. I quickly realized that typing would be better as it would allow me to make changes as I realized things needed to move around:

0	Process per round
	Choose initial player Choose word
	still_in_turn = True while still_in_turn = True
	Cound_player = init_player Spin wheel - if bankrupt; players [current player][rt'] = 0 still- in - twn = Falle - if lose turn:
	still- in - twn = Falle
9	still-in-turn = False - else: still-in-turn = True input (Would you like to [s], [b], or [g]
	To do: - manage rounds - manage winners - create third round
	- created third round

Then, I finally ended up typing the beginning of my flow and color coding it into functions! Once I started doing this, the logic of the process "clicked" and I started working in VSC so that I could trial and error as I worked through everything.

process per round:

choose initial player global init player init player = random.choice(list(players.keys())) choose correct word global correct word global display word correct_word = random.choice(word_list) display_word = '_'*len(correct_word) current_player = init_player still_in_turn = True while still in turn == True input('Would you like to [s]pin the wheel, [b]uy a vowel, or [g]uess the answer?') if input == s global wheel return wheel_return = random.choice(wheel_values) if get wheel return == 'BANKRUPT': print(f'You spun the wheel and it landed on {wheel_return}. Sorry, you lost your turn and your money from this round. Better luck next time!') players[current_player]['roundtotal'] = 0 still in turn = False if get wheel return == 'LOSE TURN': print(f'You spun the wheel and it landed on {wheel_return}. Sorry, you lost your turn. Better luck next time!') still in turn = False else:

print(f'\nYou spun the wheel and it landed on {wheel return}!')

while correct_word.find(guess, i) != -1:

i = correct_word.find(guess, i)

display word = display word[:i] + guess + display word[i+1:]

global guess

i = 0

guess = input('\nGuess a letter! ')

if guess in correct word:

```
i += 1
players[current_player]['roundtotal'] += wheel_return
continue
if correct_word == display_word:
    print(f'You win! The word was {correct_word}.')
    players[current_player]['roundtotal'] += wheel_return
    players[current_player]['gametotal'] = 'roundtotal'
    still_in_turn = False
else:
    print('Sorry, wrong guess.')
    still_in_turn = False
```