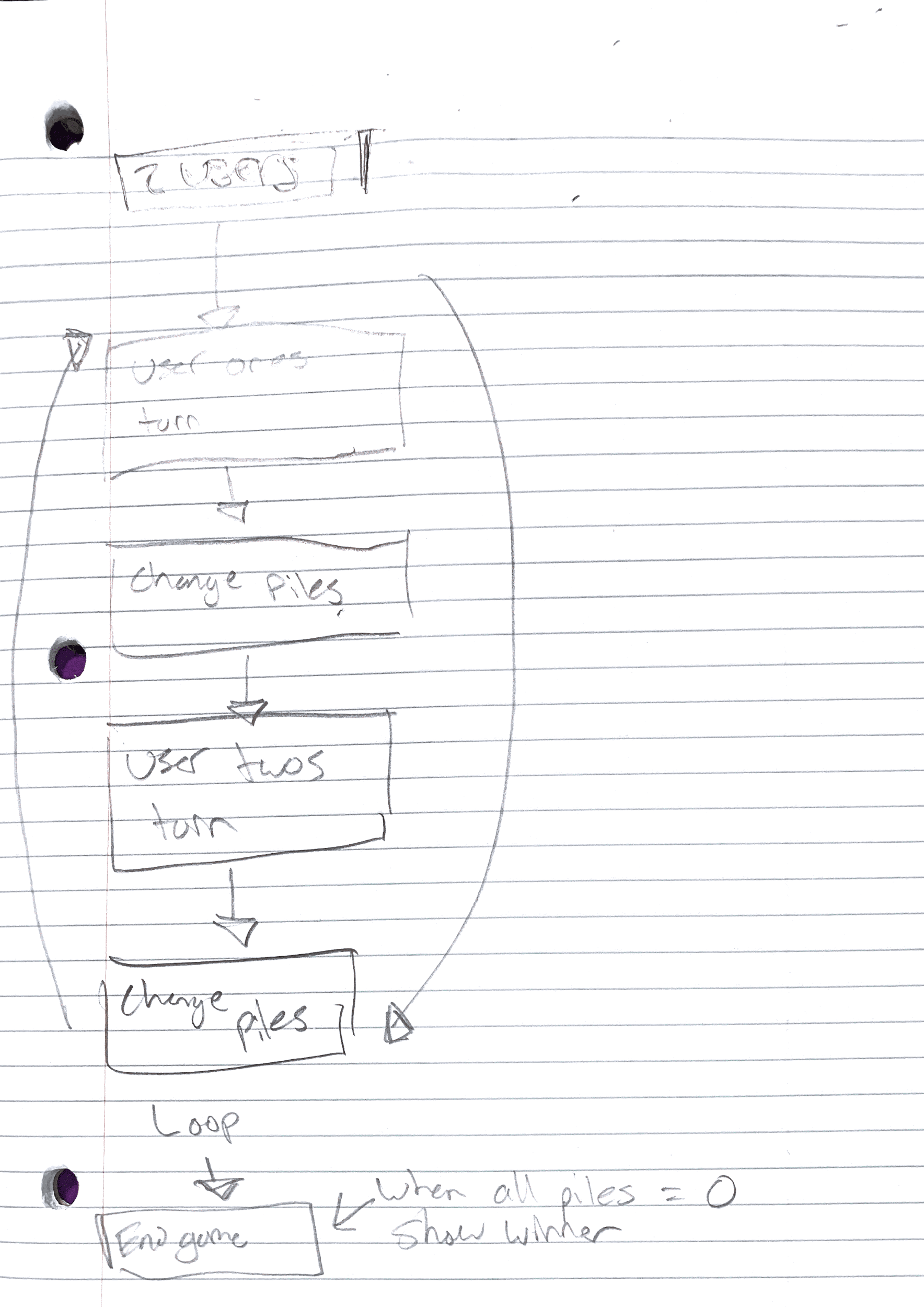
Michael Smith

Step one:

1. The user inputs are the names of each user. They are also which piles the current player wishes to choose from and the amount they want to take from it. So there is basically three kinds of inputs, the users name which is only asked once, and the pile and amount they wish to take which happens every turn until all piles are empty.
2. The outputs are which of the two players turn it is and the resulting piles and amount left after a users turn. The output can also be the ending of the game fro when one of the users win.
3. The assumptions I would make about the users input would be that the first response would be in the form of a string. Each turn would consist of two inputs from the user, one would be the pile they select and the second would be the amount they want to take out of it. These inputs would loop for each user until the game ends.
4. The tasks of this problem would be to create four “piles” or group of numbers and allowing the user to manipulate those “piles” until they all equal 0. The subtasks of this would be creating the variables, how the user’s inputs should affect those variables and ending the loop when the piles are empty.

Step two:

1. The calculations the program would be doing is subtracting the user’s input from the amount already in the “pile” and then storing the new value for the next turn. It also needs to know when the values of each piles reaches 0 so it can get rid of it.
2. You should read inputs from the user when it's their turn and when they enter their name. The users should be alternating inputs until the game is over so you’ll be reading inputs from two alternating users throughout the game.
3. Each turn of asking the user what pile and how much they wish to take out should be looped until all piles equal zero. The only difference from a normal loop is that it alternates from one user to another back to the other.
4. The bad input would consist of number greater than the amount contained in a pile or negative numbers that could add to a pile.



Step three:

Inputs:

Integer = good unless negative and too high of value

String = bad unless its at the beginning when they enter their names

Pile = good, unless the user enters a pile that does not exist or is already empty