Name: Hannah

Step 1 - **canPlayOn** method pseudocode

If color on upCard == Color on card
Return true
If number on up card == number on card
Return true
If the up card is a wild card and called color == color on card
Return true
If rank == rank && rank is not == Number
Return true
If rank == wild or == wild_4D or Wild == custom
Return true
Else
return false

Step 2 - **getWinner** method pseudocode

Declaire a winner variable and winnerscore

For listPlayers.lenght, i++

If score[i] > winnerscore

winnerScore = score i
Winner = i

Return winner

Step 6 - **doCustomCard** method pseudocode

Can play on return true if rank == CUSTOM

Step 7 - **your strategy**

1. Explain your strategy. What is your algorithm for choosing a card to play? What is your algorithm for choosing a color when playing a Wild?

My algorythmen for chosing what card to pay is first to chek if any of the other players have less than 2 cards in their hand. If one of them do I don't want to play the color that they last called. If that is possible with my hand. I am doing this to prevent them putting down their last card which will give them all the points for that round.

After my algorythm has checed this it will check if any of the possible cards i have to play have a forfeight cost above 10 I do this so that I can get rid of any of my cards with the highest cost first so that if I loose the other players will not get as meny points. I wil however not play the Wild or the custom card because I want to save that for a time that i really need it.

I made my player Exstend Eager Player because i dont like the part of the play method for the somewhat less eger player that just plays the custom card right away, because that can be a good card to save for later since it can be played on enything.

For the rest of the game I am just playing the first possible card that can be played in my hand.

To decide what color to call when I have a wild card I'm choosing the color that I have the most cards of. Because then there will be a greater chans that that color is still playing when its my turn to play again.

2. Compare your strategy to the other two players (EagerPlayer and SomewhatEagerPlayer). You should modify the main method in MultipleUnoGame so it plays three or more players against each other. Be sure to run your code at least three times for 100,000 or more games. If the results are not consistent, you should increase the number of games and run the code again.

Report the number of games needed, the final scores in the scoreboard (average the three simulations) and the ratios to demonstrate which player is best (and how much better it is than the other strategies). The ratio is the number of points earned by the player divided by the total number of points earned by all players.

Number of games played: 100,000

Player	**Average Score**	**Ratio**
EagerPlayer	1999699 1992166 2018660 Average: 2,003,508	0.2961
HannahPrettyGoodPlayer	2614632 2590566 2597913 Average: 2,601,037	0.3845
SomewhatLessEagerPlayer	2147119 2186130 2149001 Average: 2,160,750	0.3194

3. Which player is best? By best, we mean which strategy accumulates the highest number of points over a large number of consecutive games.

The best player is HannahPrettyGoodPlayer!

Project Reflection

4. List all files that you made any changes between Milestone 2 and now. If changes were made to files you turned in for Milestone 1 and 2 besides Card.java, list the changes.

I made changes to:

Card.java

Deck.java

SomewhatLessEagerPlayer.java

HannahPrettyGoodPlayer.java

TestHannahPrettyGoodPlayer.java

MultipleUnoGame.java

5. How did using inheritance benefit you in writing this program? How did it hinder you?

   It benefitted my project when when i was writing the Player methods because i dint have to write the methods again.

   At first I made the HannahPrettyGoodPlayer extend SomewhatLessEagerPlayer but when i figured out I didnt want one part of a play method I knew no good way to leav out this part. So i decided to use make it Extend Eager player instaid, but then i had to write the call Color method again because EagerPlayer's callColor method was very bad.

6. Describe any problems you encountered in this assignment. What was hard, or what should we warn students about in the future? How can we make this assignment better?

   There was a lot of for loopes and if statments inside eachouther and sometimes it was hard to to put everything inside the right amout of { to get my code to do what i want when I wanted it to do that.

   I really liked this project it helped me understand what we had been working on in class better and there was a lot of fun problemsolving included in the project. I can't think of enything that will make this assignment better.

7. What did you learn from doing the assignment? Describe any "a ha" moments from working on the project. You can also provide additional comments about the assignment or your submission here.

I learned how inheritance work and how to use diferent files together. I also learned a how to writhe methods and use objects from other clases and how that is often what connect the classes. (I think this was an a-ha for me)

I learn how to use a Main method and the bug to debug my programs. (this was also an a-ha)

This project put a lot of peaces together for me from what we learned in class.