* ***Author/Authors:*** Tradd Schmidt
* ***Motivation***: I thought it was interesting to use code to represent cards in a deck. I wanted to expand upon the program by creating code to represent another game, but this one will actually have user input. The War card game does not have user input and I think that is a bit boring.
* ***Purpose***: My software was designed to give instructions on how to play, and allow two people to play the game Trashcan.
* ***Audience***: My expected audience is anyone that wants to play the card game Trashcan, but does not have any cards to play it with, or just want a program to do everything but the placing of Kings for them. Users should mainly get out of this program some sort of entertainment,
* ***Instructions:***

Use enter to advance through the game.

Each player will have a field they will need to try to clear that is initially filled with random cards.

For a player to complete their field, they must continually draw cards to fill each slot with the appropriate card

When the correct card is placed in its appropriate spot, the card that was already there is picked up. If that card's slot has not been filled yet, it is placed in that slot and the card that was there is your new card. This continues until you have a card whose slot has already been filled.

Here is an example field: [7, 5, 4, 5, 2, 5, 9, 11, 10, 12]   
You draw a 4. It is swapped with the card in the 4 spot.   
 [7, 9, 4, 'Four', 2, 5, 5, 11, 10, 12]  
The four slot now says 'Four' to indicate that that slot has been filled with a 4.  
The card that was there was a 5. The 5 card is now placed in the 5 slot.  
[7, 5, 4, 'Four', 'Five', 5, 9, 11, 10, 12]  
A 2 was in the 5 slot. The 2 is placed in the 2 slot.  
[7, 'Two', 4, 'Four', 'Five', 5, 5, 11, 10, 12]  
A 5 card was in the 2 slot. Since the 5 slot has been filled, nothing can be done.  
  
Cards that go into slots are 'Ace' up to '10'. If you draw a Jack or Queen, or if you draw a card whose slot has been filled, your turn ends.  
If a King is drawn, you have the option of where to place it. If a card is drawn whose slot is filled with a King, the King is picked up, and it can placed in another slot.  
  
Each round that is completed, the number of slots on the field goes down for each player who cleared their field.  
Whoever finishes the number of rounds first, wins.

* ***Design***: The program asks the user if they want instructions. If they do, instructions are printed. Then it asks how many rounds the user wants to play. Then each player’s cycle is run through until Kings are drawn, or they complete their field. If a king is drawn, the programs asks the player to pick where they want the king to go. When a round ends, for each player that completed their field, their new field is the old field size minus one.
* ***Enhancement***: I used the stacks and queues created for the War card game as well as one of the functions to create the deck for dealing out cards. I repurposed it to create a deck for drawing cards. In addition I added a function to both Queue and Stack to clear the items.
* ***Functionality***: It allows the user to play Trashcan for 1 to 10 rounds
* ***Files***:
  + Queue.py
  + Stack.py
  + Trashcan.py
  + War.py
  + Schmidtt-csc236FP-driver.py
* ***Utilized Data Structures:*** I used the Stack and Queue data structures. The Stack made the most logical sense for the draw and burn piles because you always draw from the top of the pile. I used the Queue for the hands to have the first card drawn to be used first, whether it was for creating the fields or for replacing cards in the field. It helped me keep logical sense of the flow of cards as well. If I were to not use these, I would probably try to build some type of an array.
* ***Big O Analysis:*** The Big O for this code is O (2n). The function has 2 actions. 1 to put cards in the hand and one to take them out. Both actions will happen the same amount of times, and the actions happen n times.
* ***Resources***:
  + I used Pycharm as my platform
    - I used Python as my language
      * I used the built in module random in the Stack queue for shuffling the draw pile
* ***Challenges***: My biggest challenge has been presenting instructions and the data in a way where it is easy to be picked up by anyone who uses my program.
* ***Testing***: I used inputs of 1-10 for the round numbers.
* ***Errors***: One error that I could not really figure out a workaround is the number representation of cards. A “0”represents an Ace and every number up to 9 is that number plus 1 (e.g. “6” is actually 7). This does not affect the data or the actual structure behind the game in any way, however, a player may think they are using a king to get for example a 2, but they are actually getting a 3.
* ***Measures and Assessment***: I believe that I gave adequate additions to my code. I was able to repurpose the two data structures of Stack and Queue and use them in a different implementation. I did use code from the original program I based this off of and I barely had to change any code which I believe is a good sign of good code reusability.
* ***Summary***: My initial design plan changed a little bit. The first thing that I added was instructions on how to play the game. My first draft for the instructions were atrocious, however, so I had to rewrite a lot of it. The second thing that I amended was the exact format of how to win. I did not realize when I started making the program that with each successful completion of a round, the next round you have to fill one less slot in your field. I changed my code a little bit to reflect how many rounds the user would like to play. This allowed for them to choose if they want to play a quick one round game, or if they wanted to play a longer 10 round game. I believe I spent about 7 hours working on this program.
* ***Comments***: I really enjoyed this assignment. I am used to the general format of freedom of choice for my final project, but I feel like we worked with some interesting programs in 236. I also debated working with the pets program, but the War card game intrigued me a bit more so I decided to work with that one. The program I made for 226, while useful, was a bit boring in my opinion. This project allowed me to have a little bit more fun with my code.