1. **Data structure and reason.**
   1. **Dealing pile**: I shuffled the cards by first adding them to a plain python list and then using the random library. After that, I put all of the even cards in player one’s stack and all of the odd cards in player two’s stack, which is called the player’s playing pile. By separating the evens and odds, that allows an even playing field by not only having the same amount of cards, but also allowing the cards to be somewhat similar in value. The card stacks are still randomized since I shuffled the cards first.
   2. **A player’s playing pile**: I saved the player’s playing pile as stacks. I chose stacks because when you play War in real life, the piles represent stacks in the sense that you flip the top card of the deck, similar to how stacks in python pop from the top.
   3. **A player’s storage pile**: I decided to save this into a queue. This is because in War, when you win a hand and add it to a storage pile, you want the first card you started to play with to be your first card again. This resembles when you take the pile of faced up cards and flip them over so that you cannot see the face. Therefore, the card that went first was on the bottom, but then ends up on top to represent “first in, first out”.
   4. **The opponent’s playing pile**: I saved the player’s playing pile as stacks. I chose stacks because when you play War in real life, the piles represent stacks in the sense that you flip the top card of the deck, similar to how stacks in python pop from the top.
   5. **The opponent’s storage pile**: I decided to save this into a queue. This is because in War, when you win a hand and add it to a storage pile, you want the first card you started to play with to be your first card again. This resembles when you take the pile of faced up cards and flip them over so that you cannot see the face. Therefore, the card that went first was on the bottom, but then ends up on top to represent “first in, first out”.
   6. **Loot pile**: This was saved as a queue, because when you add it to your pile, you want it to be in the same order that it was laid down.
2. **Initial pile**
   1. The initial pile has to be saved into a python list so that you can use the random library on it. This provides us with the option of using the random.shuffle(deck) method which would not be available if I used a stack. Therefore, it has to be a list in order to provide the deck with a type of shuffling.
3. **Advantages and disadvantages.**
   1. Opponent’s playing pile:
      1. Advantage: I used a stack because it allowed for me to quickly access the top card on the deck
      2. Disadvantage: It does not allow to keep track of what order the cards were put in
   2. Opponent’s storage pile:
      1. Advantage: I saved it into a queue so that I could keep track of the order of things
      2. Disadvantage: It does not allow you to easily compare the last two cards played though
   3. Loot pile:
      1. Advantage: I saved it into a queue so that I could keep track of the order of the cards that they appeared in, so that they could go back into the storage pile the way they came in.
      2. Disadvantage: You can never tell what the last card play was, in case you need to compare them for something like the war case
   4. Queue:
      1. Biggest Advantage: Able to store the order in which something happened, in which I implemented in the loop pile in this program
      2. Biggest Disadvantage: Not possible to know the last thing that happened
   5. Stack:
      1. Biggest Advantage: It allows you to peak and see what the last thing that happened was. This was used when compared cards.
      2. Biggest Disadvantage: They are not helpful if you need to get the order of things
4. **Disadvantage of stack or a queue**
   1. For stack and queues you can only read the top or beginning of the list. If you want to see more than that you have to remove the value at the top or bottom. Whereas a python list allows you to easily look through all of the values in the list using a loop. Also, stacks and queues provide no opportunity to sort the values randomly, whereas you can easily sort a python list.
5. **One data type**
   1. I would probably use all stacks if I had to choose one data type. This is because, with stacks, it is easier to grab the card from the top since in python stacks, you would grab the top item. Also, I value knowing the last card on the stack more than I value knowing the order of the cards on the stack. I believe knowing the last card allows us to make comparisons easier. Although it can somewhat be done with a queue, it logically does not make sense to me to implement it this way because then the outputs would always be in the same order.