Today’s Goals

1. Show exactly how lists work in functions as parameters and how they get returned from functions.
2. Continue to convince you that functions are really important to good code design.
3. Utilize lists and functions to do a full on implementation of Blackjack, where we accurately mimic having a standard deck of 52 cards.
4. Show significant planning before starting the implementation

Planning Phase

Cards will be numbers 0 to 51.

0 – 12 Clubs (2-10 will be 2-10, J=11, Q=12, 0 = K, 1 = A)

13 – 25 Diamonds

26 – 38 Hearts

39 – 51 Spades

This system is such that cardnumber//13 will give us the suit (0 = Clubs, 1 = Diamonds, 2 = Hearts, 3 = Spades)

Within a suit, card%13 tells me its kind, as indicated by the Clubs above.

Random Shuffle

1000 times, I will pick two random cards and swap them in the deck.

Mimic A Real Shuffle

Split the deck in two lists.

We will alternate removing cards from the two lists, much like intertwining cards. We will choose 1 or 2 cards to take from each list before flipping to the other.

Function to Deal a Card

Remove the top card and return it.

Players Hands

Each hand will be a list as well. We will declare these in main.

Over the course of the game, we will call the deal function which will return a card, which we will then add to a player's hand.

Scoring Function for one card

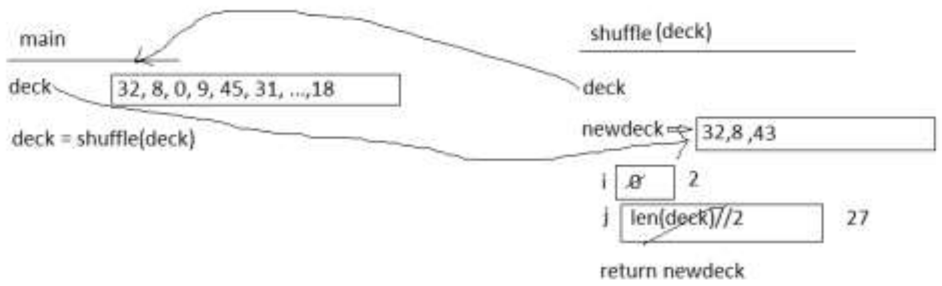
Similar to what we wrote before, and for aces, we will tentatively return 11, but in another scoring function, we will allow aces to change to 1, only if necessary.

Hand Scoring Function

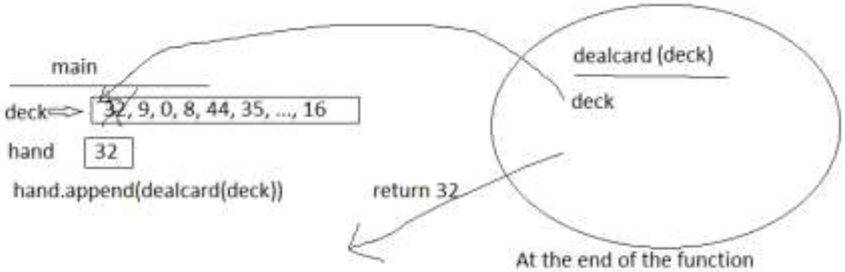
Initially add up the total points of all cards, but also, calculate the number of aces.

If the score > 21 and the ace count is > 0, we can convert each ace from 11 to 1. So while our score is above 21 and our ace count is > 0, we subtract 1 from ace count and subtract 10 from score.

Picture of Rifle Shuffle



Picture of what the Deal Card will look like



At the end of the function call, this memory disappears and all we are left with are the changes in main.