**Goal -**
Create a program that simulates a simplified game of blackjack. There are two players to the game: the individual playing the game, from which there is a Hit/Stand input option, and the dealer which will have predefined behavior.

**Notes -**

- The game is only really a few main parts:
    1. The player and dealer get 2 random cards (display them minus 1 dealer card)
    2. The player gets to Hit until they Stand, get a 21, or bust (display updated hand each time)
    3. The dealer draws as dictated by predefined behavior (display updated hand)
    4. If at any point any player busts, they lose (including the dealer)
    5. If no busts, the winner is the one closest to 21

**Questions -**

- **Exceed 21:** What happens if the deck runs out? No need to worry because we will def reach a sum >21 before then.

**Potential parts -**

- A **Hand** system to represent each player's hand (including the individual values).
- A **Deck** representation of 52 cards: 4 of each number 2-10 with their respective number as their value, 4 jokers, queens, and kings all of which have a value of 10, and 4 aces which have a value of 1 or 11 (chosen optimally).
- A **Draw** mechanic that draws a card at random from the deck (w/o replacement)
- A **Hit** mechanic to get the next card in the deck (randomly).
- **Stand** would basically just mean move on to the dealer's turn.
- **Bust** mechanic would trigger a loss for the player if they bust, and a win if the dealer busts
- A **Display hands** mechanic that displays the hand of both players on the command line (example hand: "7 10 2 = 19")
    - Dealer edge case is starting hand which must display 1 card as hidden "?"
- A **Main** system that calls all necessary parts of the game
- An **Input** system (maybe the main system?)

**Ideas -**

- Players would be defined as a class with
    - 1 attribute: A **Hand**
    - 3 methods: Hit(), Stand(), and getHandValue()

- **Hand (an array)** storing individual cards, worst case max cards for getting 21 is 8 not counting aces, so we can play it safe with a **static size of 12**
- **Hit()** will take in a deck as arg, drawing a random element from it (popping), and adding it to the player's hand
- **Stand()** should send a message to the game System to move to dealer's turn
- **Ace optimality:** Theres only really 2 permutations for any number of Aces, either all 1s or an 11 and the rest as 1. This is because any 2 or greater Aces = 11 will exceed 21, so we can simply take the number of Aces in our hand, and evaluate these two options, adding the most optimal one to our handSum

- **Card** is a class defined as having a string representation and numerical value. Could maybe just put this as a map? String -> int mapping

- **Deck** is a class holding an array of strings (cards)
    - Starting deck will consist of 4 of each unique card ([“A”, “A”, … “7”, “7”...“K”, “K”, “K”, “K”)
    - 2 methods: shuffle() and draw(), which do as their names imply
    - size() method maybe nice but not necessary (Exceed 21)

- **Bust** will be a method that takes in a **Hand** and returns a bool indicating whether the hand is over 21 or not.

- **Input** system should indicate **“H” as Hit** and **“S” as Stand**
    - Maybe a while true loop that accepts input, and the break conditions will be if the player busts or if they hit **“S”**

- **Display** system
    - Should display both players' **Hand** any time after Hit is called.
    - Should display win and lose messages.
    - Maybe just well-placed print statements, or maybe a class

- **Main** system
    - Initialize the deck
    - Initialize both players
    - Call hit on both players
    - Set up the termination system for busts/end of game
    - Player's turn: Let them Hit/Stand accordingly (maybe a while true loop)
    - Dealer's turn: Hit until **Hand** total is >= 17
    - Evaluate scores if nobody busted

Assumptions -

- Hands, being represented by arrays, don’t need to concern about real-time expanding as the hand will never be able to grow infinitely (will exceed 21 before drawing the whole deck and more). Thus hands can be set to a static size in this one case.
- Dont have to worry about the deck ever running out or any problems that arise from that. (Exceed 21). If we had more players then maybe we would take this into account but not necessary for this program.