* LIST OF CONCEPTS FERSURE NEED TO BE INCLUDED
  + the stack pointer (deck of cards?? research blog post as you develop)
  + PUSH POP - > saving registers onto the stack
  + indexing through the stack pointer - post/pre
  + local memory versus global memory
  + shifts ?? try adding places you mult by the same num
  + Possibly arrays? maybe actually create a predefined area of memory and then ldr that memory into a register, then use that to store the values produced by the random number generator, just have it create the values, store them in the deck and then return a card value to returned for the player. Have this function also print out the card prior to returning back to the main function.
    - so this function would print out the card, and then calculate the card value, then the card value would be returned in the given register, and then passed to the add total function to be added to the players overall card total which would be stored in a variable in main.
* go through each of the lessons on the blog and make sure you include all or most of those concepts. -> make the list of concepts included as you go.
* clean up the code - > get rid of all the extra variables, push appropriate variables for each function if its a register above 3 and you use that register in the function, ex if you use register 5, make sure you push it at the start of the function and pop it back so the previous register five is saved -> figure out first, make an outline for each function before you code it, write it up in C first.
* MAKE outlines of each function first and determine how each should function and which values need to be passed back between the functions.
* figure out a way to incorporate the stack into the function
  + can probably place the card values of the deck into the stack and then increment along the stack pointer for each card, then when the stack pointer hits a certain threshold, creates a new deck of cards and resets the stack