**Programs are to be the sole effort of the individual turning in the assignment. Collaborative efforts and code sharing/copying are not allowed. Programming problems are only to be discussed with the Tas and instructor. Getting assistance from other students is not permitted**

**Are You a Card Psychic?**

**Due: Thursday April 6, 11:55 p.m.**

**Objective:** Gain experience using lists and functions in Python.

**Program:** For your next project, you are going to write a program to simulate a test to determine if a person is “card psychic” based on the user’s (subject’s) ability to guess the cards in the computer’s (tester’s) “hand”. The test works as follows:

The “tester” draws some number cards from a deck and holds them in their hand.

The subject has some maximum number of tries to guess the cards that the tester is holding. If the subject guesses a card being held by the tester, the tester removes the card from their hand and lays the card on the table.

After the subject has either guessed all of the cards in the tester’s hand or has used up the maximum number of guesses, the tester lays the remaining cards from their hand on the table (if there are any cards still in their hand) so that the subject can see what cards the tester is still holding. The performance of the subject is then evaluated as follows:

If the subject guesses all the cards in the tester’s hand (that is, the tester’s hand is empty), then the subject is determined to be **“definitely card psychic”**, if all but 30% of the original cards in the tester’s hand (rounded up, so that if the tester started with 7 cards, the subject would have had to guess all but 3) they are determined to be **“possibly card psychic”,** otherwise they are determined to be **“not card psychic”**.

**The Code:**

Your program should maintain two global variables that are lists: **deck**, which will hold the deck of cards, and **hand** which will hold the cards in the tester’s (computer’s) hand, that is, the cards whose values the subject (user) must try to guess.

Your program MUST contain the following functions, which must have **EXACTLY** the names, parameters, return values and functionality specified:

* **getDeckName()** – takes no parameters and will prompt the user to enter the name of the file containing the “deck” of cards. It will continue to prompt the user for a name until the name entered ends in “.txt”. When a valid name has been entered, it will **return** the name of the file.
* **getDeck(string)** – the function takes a string parameter, which is the name of the file containing the cards. The file will contain one “card” per line with each card consisting of a character representing the suit (H,S,D,C) followed by the value (2-10, J,Q,K,A). The information should be read from the file whose name is passed as the parameter and its contents placed in the list representing the deck. The first card in the file should be the first card in the list representing the deck etc.
* **showForTest(int)-** this method takes an integer parameter and is used for testing your program (so that the calls can be easily removed when the program goes into “production”). If the function is passed a 0, it should output the contents of the deck and if it is passed a 1, it should output the contents of both the deck and the tester’s hand, with labeling to make the output clear. If any other integer is passed, it should still output an error message.
* **getValidNumber** (string, int, int) – this method will accept a string parameter that it will be used to prompt the user to enter a value, and two integers. The function will continually prompt the user to enter a value that is between the first integer and the second integer (inclusive). It will **return** the valid integer value entered. For instance, if it is called with

num = getValidNumber(“Enter your age:”, 1, 120)

the function will continually prompt the user with “Enter your age:” and will insure that the user has entered a number between 1 and 120, re-prompting if necessary. It will return the valid value entered by the user (in this case, a number from 1-120).

* **chooseCards(int)** – will take one integer parameter indicating the number of cards that should be removed from the deck variable and placed into the variable representing tester’s hand. So, for example, if it is passed 5, then 5 cards should be removed from the deck and placed in the tester’s hand. The positions of the cards to be removed from the deck are randomly chosen. However, remember that each time a card is removed, the deck will shrink by 1, so the deck will first have 52 cards, then 51 cards etc. Make sure that the range of your random number choice adjusts accordingly. It is assumed that 0 means pick the card on the top of the deck (first card in the list).
* **getGuess()** – will prompt the user to enter a card and will continually prompt the user until a valid card value is entered. The function will **return** a string that is the valid card entered by the user.
* **runTest(int)-** will be passed an integer parameter indicating the maximum number of guesses the subject is allowed in order to try to guess all the cards in the tester’s hand (remember, if the person is psychic or close to it, they may not need all of their guesses). This function will continually call *getGuess* to get the user’s next guess, check that guess against the cards in the tester’s hand and remove it the guessed card from the hand and display it on the screen if a match is made. The function will repeat this process until the user has used up all of their guesses or has guessed all of the cards in the tester’s hand.
* **showResult()** – this function will determine and output the evaluation of the subject’s “card physic ability” based on the discussion at the beginning of the assignment. It will first display the cards remaining in the tester’s hand (if any) then output the phrase describing the psychic ability they demonstrated based on the number of cards remaining in the tester’s hand.

**The Main**

The main code you your program should follow the algorithm:

* Use getValidNumber to ask the user to enter the number of cards the tester should draw from the deck (it must be a number between 3-52).
* Use getValidNumber to ask the user to enter the maximum number of guesses the subject is allowed (it must be a number that is greater than or equal to the number of cards in tester’s original hand and 52).
* Call getDeckName() to get the name of the file for the deck
* Call getDeck() to get the deck
* Call showForTest passing it 0, so that the deck will output (for program testing)
* Call ChooseCards passing it the value entered by the user for the number of cards in the tester’s original hand.
* Call showForTest passing it a 1, so that the deck and the tester’s hand will output (for testing)
* Call runTest passing it the maximum number of guesses that the user had entered earlier
* Call showResult to determine and display the final result

Note: If you need to remove the ‘\n’ from the end of the string representing cards that are read from the file, use the rstrip String method. It is used as follows:

String\_variable.rstrip()

It will return the value contained in the String\_variable with any whitespace characters removed from the end of the string. Remember that it is not changing the contents of the String\_variable itself, it is just returning a value that was derived from the contents of the String\_variable. You will need to use this value (and therefore the method call) in another statement, such as an assignment statement. So if name = “Jim\n” the your call could be

new\_name = name.rstrip(), and new\_name would then contain “Jim”.