# Assignment P6

**Due Date: November 23**

## Purpose

This assignment brings together almost every Python feature covered to date. Some of the items you'll get practice with include: one-dimensional lists (arrays), classes, and more functions/methods.

## Problem

Mick Gousilinski just invented a new card game called MickSlam. He thinks that he will make millions if he had a computer version of it. Thus, he has hired you to write a program to implement his (very easy) card game.

## Rules of MickSlam

The player and the computer are each dealt five cards from a standard French deck. The cards and total points of the player's hand *(only!)* are displayed. The player then decides whether she thinks the computer has a better hand or not. After the player's input, the computer's hand and points are displayed. The player wins or loses that round depending on whether she guessed the outcome correctly. Play continues until the player quits or there are no more cards in the deck.

Point values are determined as follows:

| Card | Point Value |
|---|---|
| Ace | 11 |
| J, Q, K | 10 |
| 2-10 | 2-10 |

## Input

After the player's hand is displayed, she will input $p$ if she thinks the player will win or $c$ if she thinks the computer will win. An input of $q$ ends the game (if there are no more cards, the game ends as well).

## Output

The program should play the game, and show each hand as the game progresses. For each hand, all of the player's and computer's cards should be displayed. The cards are drawn at random from a standard deck of 52 French cards. The card display should include the suit and the value of the card, including face cards (A=Ace, K=King, Q=Queen, J=Jack). The totals for the player's and computer's hand should also be displayed as well as an indication if the player won or lost that round. The total number of winning and losing guesses are displayed at the end of the game.

For example, a hand of the basic game might proceed as follows:

```
Your cards:
4♡  5♣  3♡  A♣  3♢   Total:  26
Do you predict the player (p) or computer (c) will win (q to quit)?  c
Computer's cards:
9♢  J♠ 10♠  6♠ 10♣   Total:  45
You Win!
```

Play should be suspended between hands; the player should press the `return` key to continue. Play stops when there are not enough cards left for a complete hand or when the user quits.

## Specifics

- Create a **Deck** class, similar to what we did in class, that will store a deck of cards and whether a particular card has been played. You will have to modify one or more class methods for the class to work properly for the game. **Note that this class should have access only to the remaining cards in the deck. It should not** *play* **any part of the game!** (See below)
- Create a **Game** class that stores the player's and computer's hands and contains methods that access and modify those hands.
- Create one or more methods in the Game class that play the game. Because many of the useful variables will be contained in your class, you will have fewer parameters to worry about.
- Use named constants as appropriate. For example, it should be easy to modify your program if a programmer wants to change the game so it uses six cards per hand instead of five.
- The design of much of the program is up to you. However, you must break your program into functional parts. Each part should accomplish a minor task, such as displaying one hand.
  - No function/method (including `main()` should be longer than than about 35 lines, **including** comments and declarations. Methods should solve small problems!
  - Use parameters correctly; do not use global variables.
  - Use the `return` statement when/if appropriate.
  - Do not write essentially identical code repeatedly! If you find you are doing this, replace that code with a method/function that you **call** repeatedly, put your code in a loop, or both.
  - Comment all functions/methods including pre- and post-conditions. Give a short description of each function/method as well as what each parameter is for.
  - Because most of the game functionality is accomplished in the Game class, your `main()` may be very short. This is normal.
- Suits can be displayed with the control character `"\u"` and a numeric code. For example, `print ("\2665")` will print ♡.
- All output should be clear and aligned.

## Notes

As usual, submit your source code via email as usual; mine would be named `mgousieP6.py`.

*Anybody who can master a telephone can certainly program.*
– Bob Frankston, creator of VisiCalc, an early spreadsheet program