Assignment 7: Modular Design

Due Tuesday, May 5

This program is meant to give you more practice with functions and large-scale program design, as well as an introduction to strings. You will be creating a version of the card game “baccarat” (probably most famous for being in James Bond movies).

Game Rules

- The game is played by two players – the “player” (played by the user) and the “bank” (played by the computer).
- This is a card game, in which the goal is to have the hand with the highest “value”. If the player and bank tie, the player wins.
- The “cards” are numbered between 1 and 9 (we will not worry about suits, face cards, etc.).
- The “value” of a hand is the total of the numbers of each card in the hand. However, if the total is greater than 10, we subtract 10 (in other words, the value of a hand is always between 0 and 9).

For example, if a player has 4, 8, and 3, the hand value is 5 (as 4 + 8 + 3 = 15, and 15 – 10 = 5).

Game Play

1. Initially, each player is dealt two cards. The player

2. The player is then given the option of drawing one additional card.

3. The bank may then also draw one additional card. Whether they do so or not is based on the following rules:
   - If the player did not decide to draw a card, the bank draws a card if the value of their hand is less than 6.
   - If the player draws a 2 or 3, the bank draws a card if the value of their hand is less than 5.
   - If the player draws a 4 or 5, the bank draws a card if the value of their hand is less than 6.
   - If the player draws a 6 or 7, the bank draws a card if the value of their hand is less than 7.
   - If the player draws an 8, the bank draws a card if the value of their hand is less than 3.
   - If the player draws a 1 or 9, the bank draws a card if the value of their hand is less than 4.
User Interface and Gameplay Examples

As described above, your program must do the following:

1. Generate two random cards for the player and the bank, displaying them and the totals:

   >>>
   Dealing to player:
   Dealt a 7 and a 4 for a total of 1
   Dealing to bank:
   Dealt a 2 and a 1 for a total of 3

2. Prompt the player whether or not they want to draw another card. If they do, generate another card and add it to their total:

   Would you like another card (yes or no):
   You were dealt a 6 for a total of 7

3. Use the above rules (based on the card drawn by the player and the total of the bank hand) to decide whether or not to draw a card for the bank. If they do, generate a card and add it to their total:

   The bank takes a card.
   The bank was dealt a 5 for a total of 8

4. Decide who had the higher hand, and decide a winner (remember, the player wins ties):

   The bank wins.

Another few examples:

   >>>
   Dealing to player:
   Dealt a 4 and a 4 for a total of 8
   Dealing to bank:
   Dealt a 9 and a 7 for a total of 6
   Would you like another card (yes or no): no
   The bank stands.
   You win!!

   >>>
   Dealing to player:
   Dealt a 2 and a 3 for a total of 5
   Dealing to bank:
   Dealt a 1 and a 6 for a total of 7
   Would you like another card (yes or no): yes
   You were dealt a 9 for a total of 4
   The bank stands.
   The bank wins.
Input Validation and String Processing

You are also required to validate that the player enters either yes or no when asked whether they want another card, and to keep asking until they answer yes or no:

```
>>> Dealing to player:
Dealt a 3 and a 5 for a total of 8
Dealing to bank:
Dealt a 3 and a 1 for a total of 4
Would you like another card (yes or no): no
The bank takes a card.
The bank was dealt a 8 for a total of 2
You win!!!
```

In order to make the input more user friendly, you should also accept input in capital and lower case letters, and with extra spaces. Note that this can be done with the `strip` and `lower` functions in the `string` class. For example:

```
Dealing to player:
Dealt a 1 and a 9 for a total of 0
Dealing to bank:
Dealt a 2 and a 6 for a total of 8
Would you like another card (yes or no): maybe
Please answer yes or no: what?
Please answer yes or no: yes
You were dealt a 9 for a total of 9
The bank stands.
You win!!!
```

Design and Implementation Requirements and Hints

You are required to break this program down into a series of functions. This is one of the major components of the grading for this assignment

Note that there are many good ways to break this down into functions. However, I do have some suggestions:
• Create functions for things that you must do **multiple times** throughout the program. Specifically, you should minimize the code that you are repeating over and over by just writing it **once** as a function and then calling it when needed.

For example, consider creating simple functions to:
  - Generate a **random card** with a value between 1 and 10.
  - Compute and return the **total** of two things, subtracting 10 if necessary (this might be very helpful to add a new card to the total for the player and bank hands.
  - Deal someone (either the player or bank) **two cards**, printing the cards and returning their total.

• Create separate functions for the **major things** your program must do, in order to simplify the structure of the main program.

  Some possibilities for functions include:
  - Prompting the player whether or not they want to draw a card, including the string processing for user friendliness and the loop to keep prompting until the answer **yes** or **no**.
  - Determining whether or not the bank should draw a card, based on the card drawn by the player, and the bank’s current card total. Hint: it might be a good idea if this function returned **True** or **False**.

Finally, you are not to use any **global** variables in your program – all variables must be local to their functions (your **import** statements may be done globally, however).

---

**Documentation and Readability**

As always, your program must be **well documented** and **readable**. In particular, for this program make sure that you put a “**function header**” before each function definition that provides a brief description of what that function does. Keep in mind that this will be a significant part of your grade!

**What to Turn In**

**Email** your **assignment7.py** program to me at **jrsullins@ysu.edu**.

• The **assignment7.py** program should be an **attachment** to the email.
• The **subject** of the email should be “**1595 Assignment 7 from your name**”.