15-110: Principles of Computing, Spring 2018

Programming Assignment 8

Due: Tuesday, April 3 by 9PM

Note: You are **responsible for protecting your solutions** to the following problems from being seen by other students both physically (e.g., by looking over your shoulder or verbal discussion) and electronically. In particular, since the lab machines use the Andrew File System (AFS) to share files worldwide, you need to be careful that you do not put files in a place that is publicly accessible.

If you are doing the assignment on the Gates-Hillman Cluster machines we use in the lab or on `unix.andrew.cmu.edu`, please remember to have your solutions inside a `private` folder (which is under your home directory). Our recommendation is that you create a `pa8` folder under `~/private/15110` for this assignment. That is, the new directory `pa8` is inside the directory named `15110`, which is inside the `private` directory.

Overview

For this assignment, you will create a Python file for each of the problems below. You should save all of these files in a folder named `pa8`. Once you have every file, you should zip up the `pa8` folder and submit the zipped file on Autolab.

This assignment will help you explore some functions for games and some simple computer graphics. **NOTE: If you use your own laptop and have Python 3 installed, you should have the tkinter module available. See a TA if you wish to install Python 3 on your laptop. If you ssh into the Andrew servers, tkinter is now available to import, but you need to use the –X option when connecting.**

Problems

1. [4 points] In the file `hearts.py`, define a function `score(hand)`, which takes a list of playing cards (as defined in class) called `hand` and returns a score for that hand based on the card game Hearts. You may assume the list has at least one card.

   In this version of the game of Hearts, the following rules are used to score a hand:

   - Each card that has a suit of Hearts is worth 1 point, unless the card is the Ace of Hearts, which is worth 5 points.
   - The Queen of Spades is worth 13 points.
   - The Jack of Diamonds deducts 11 from the score.

- The 10 of Clubs doubles the final score. (See hint below.)
- All other cards do not affect the score.
- If the final score is negative, it is reset to 0.

Your function should have a variable to hold the score, initialized appropriately. You should use a loop to look at each card one at a time and update the score appropriately. HINT: For the 10 of Clubs, instead of doubling the current score, since you don't know the final total yet, you can set some Boolean variable to True to indicate that you saw the 10 of Clubs. –How do you initialize this Boolean variable?– Then you can double the score once all cards are examined. Remember that if the final result is negative, the player must get 0 instead. Return the final score.

Include the code for `create_deck`, `get_rank` and `get_suit` in your file! These are posted in the `cards.py` file on the course website.

Sample usage:

```
python3 –i hearts.py
>>> deck = create_deck()
>>> hand = [deck[26], deck[27], deck[28], deck[38]]
>>> hand
[['2', 'hearts'], ['3', 'hearts'], ['4', 'hearts'], ['A', 'hearts']]
>>> score(hand)
8
>>> hand = [deck[22], deck[49]]
>>> hand
[['J', 'diamonds'], ['Q', 'spades']]
>>> score(hand)
2
>>> hand = [deck[8], deck[26], deck[38]]
>>> hand
[['10', 'clubs'], ['2', 'hearts'], ['A', 'hearts']]
>>> score(hand)
12
>>> hand = [deck[22], deck[8], deck[26], deck[38]]
>>> hand
[['J', 'diamonds'], ['10', 'clubs'], ['2', 'hearts'], ['A', 'hearts']]
>>> score(hand)
0
>>> hand = [deck[22], deck[26], deck[8], deck[49]]
>>> hand
[['J', 'diamonds'], ['2', 'hearts'], ['10', 'clubs'], ['Q', 'spades']]
>>> score(hand)
6
>>> hand = []
>>> for i in range(0,13):
...     hand.append(deck[26+i])
...
>>> score(hand)
17
```

2. (2 points) In the file `portrait.py`, write a function `portrait()` that draws an abstract portrait of you on a canvas of size 400 X 500. We should be able to identify a face, at the very least. You must use at least one rectangle, one circle and one polygon in your portrait, although we expect you'll use significantly more for full credit. Be creative! (We may show some of the really creative results in class.)

   Sample usage:
   ```
   python3 –i portrait.py
   >>> from tkinter import *
   >>> portrait()
   ```

   (your abstract portrait displays here!)

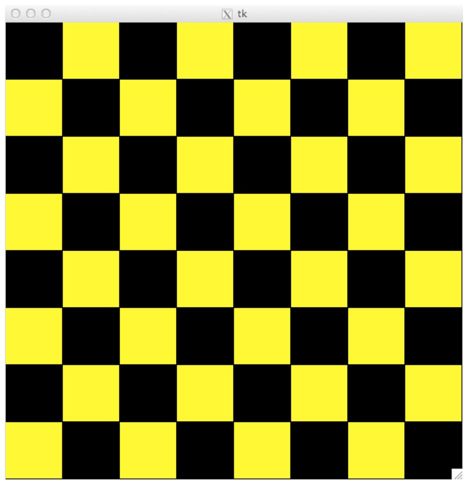
3. (2 points) In the file `checkerboard.py`, write a function `checkerboard()` that draws a standard 8 X 8 checkerboard on a canvas of size 600 X 600. Squares alternate black and yellow on the first row, then alternate yellow and black on the second row, then alternate black and yellow on the third row, and so on. Each square should be the same size and the board should fill up the canvas (minus the usual thin border strip that occurs due to tkinter). Your function must utilize a nested loop, one loop for the row and one for the column of your checkerboard.

   HINT: The color you pick for a square is a familiar function of the row and column. Here are some examples:

   Row 0, column 0 is black; row 2, column 4 is black; row 5, column 1 is black.
   Row 0, column 1 is yellow; row 3, column 4 is yellow; row 7, column 2 is yellow.

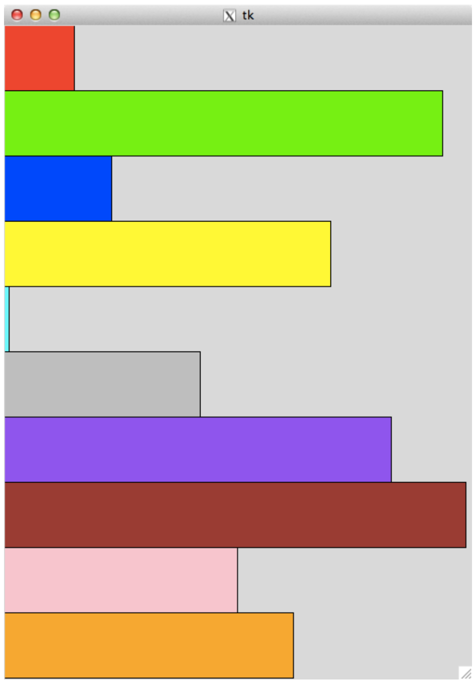
   Do you see a pattern here?

   Sample usage:
   ```
   python3 -i checkerboard.py
   >>> from tkinter import *
   >>> checkerboard()
   ```



4. (2 points) In the file `bargraph.py`, write a function `bargraph(datalist)` that takes a list of 10 data values and displays a bar graph of these data values. The bar graph should be displayed in a canvas of size 500 X 700, and each bar should have an equal width. The bottom of each bar must be anchored on the left side of the canvas, and the length of the bar must be proportional to the data value being displayed, with the maximum length representing 100. Each of the data values is guaranteed to be in the range 1 to 100 inclusive. Each bar should have a different color of your choosing, and these colors should be stored in a list so you can write a compact function.

   Sample usage:
   ```
   python3 –i bargraph.py
   >>> from tkinter import *
   >>> datalist = [15, 94, 23, 70, 1, 42, 83, 99, 50, 62]
   >>> bargraph(datalist)
   ```



Submission

You should now have the `pa8` folder that contains the following Python files:

a. `hearts.py`
b. `portrait.py`
c. `checkerboard.py`
d. `bargraph.py`

Zip up the folder and submit the zipped file named as `pa8.zip` on Autolab.
Be sure to check your submission to see that you submitted the correct code.