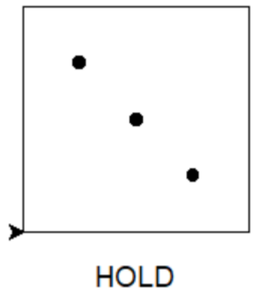
Computational Problem Solving CSCI-603
Pig Dice Turtle Game Lab 2

6/1/2021

1 Problem

In your programming assignment you will be developing a graphical application to play the simple dice game "Pig".

1.1 Problem Solving Session



For this lab, you will deal only with drawing a 2-dimensional face of a die using the Python turtle package.

You will work in a group of three or four students as determined by the instructor. Each team will work together to complete the following activities.

Assume the following:

- All drawing functions begin and end with the turtle facing east at the lower left hand corner of the die, pen up.
- There is a constant `SIDE` that is the dimension of one side of the die.

Assume you have already written the following functions:

- `draw_outline` draws the square outline of the die.
- `draw_center_dot` draws a single pip at the center of the die.

1. Write code (not a whole function) that draws a die representing the number 1.
2. Write a function `draw_two_dots` that draws the pips for the die face representing 2.
3. Using code that only calls `draw_outline`, `draw_center_dot`, and `draw_two_dots`, and code that moves the turtle with the pen up only, write code that draws the entire die face representing the number 4. Again, no function definition is needed.
4. Write a function `draw_die` that takes a single argument `pips`, the number of pips to show on the die face. It should be capable of drawing the entire die face for the numbers 1, 2, 3, 4, or 5.

   Any other parameter value should result in an error. Do this by putting the following statement in the proper place:

   ```
   assert boolean-expression, "Illegal #pips: " + str( pips )
   ```

   If the boolean-expression evaluates to False, the assert statement will raise an exception. Define that boolean-expression.

   The same restrictions to the code in the previous question still apply here.

If you finish earlier, define the content of the `draw_outline` and `draw_center_dot` functions.

At the end of problem-solving, put all group members' names on the sheet, number each item and hand in your work, one copy per team.

2 Implementation

For the implementation, you will write a program called `pigdice.py`. This will be a graphical representation of the game Pig.

2.1 Game Play Rules

Each turn, a player starts with a turn total of 0 and then repeatedly rolls a die until either a 1 is rolled or the player decides to *hold*.

- If the player rolls a 1, they score nothing and it becomes the next player's turn.
- If the player rolls any other number, it is added to their turn total and the player's turn continues.
- If a player chooses to hold, their turn total is added to their score, and it becomes the next player's turn.

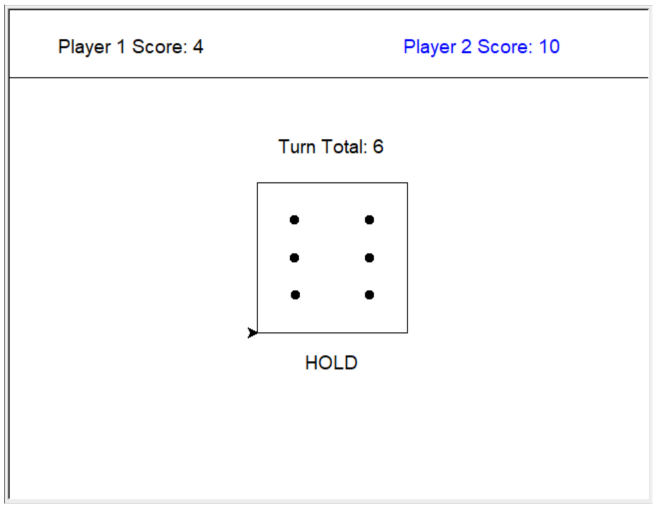
The first player to score 50 or more points wins. (The real game goes to 100, but for testing purposes, we are dropping it to 50.)

Here is an example of play.

1. Tom goes first. He rolls 5, 3, 2, 2, then holds. His score is 12.
2. Jim goes second. He rolls 6, 6, 3, 4, then holds. His score is 19.
3. Tom rolls 4, 6, 6, 1. He gets no new points. His score is 12 + 0 = 12.
4. Jim rolls 5, 4, 3, 4, 5, and holds. His score is 19 + 21 = 40.
5. Tom rolls 2, 2, 2, 5, 6, and holds. His score is 12 + 17 = 29.
6. Jim rolls 1. He gets no new points. His score is 40 + 0 = 40.
7. Tom rolls 4, 5, 6, 3, 6, 6, and holds. His score is 29 + 30 = 59.
8. The game is over, and Tom has won.

2.2 Lab Details

The game will be run assuming two players, both sharing the same instance of the application.



Your program must be graphical, using the `turtle` package. You are free to develop your GUI how you wish, but it must contain at least the following features:

- a drawing of the die face
- a HOLD button
- mouse event handler (using the function `turtle.onscreenclick`) for
  - detecting a left-button mouse click within the die
  - detecting a left-button mouse click within the HOLD button
- the turn total
- both player's score

Clicking on the die causes simulation of rolling the die to get a new number. (No rolling animation necessary!!) Use the `randint` function in the standard library `random` module. If the roll results in the die representing the number 1, the turn total will be lost and play will switch to the other player.

The HOLD button can be of your own design, as long as it's obvious that you should click on it. Clicking on the HOLD button will cause the current player's score to be updated and the state of the game to change to the other player.

The players must be notify somehow what has changed every time there is a mouse action. Minimally, after each action, your game should display

- whose turn it is
- the turn total so far
- the scores for both players (not including current turn total)

To keep track of these things, a class is provided to you. It is the class `Keeper` in the file `score.py`. Read the test program to learn how to use classes in Python. The `score.Keeper` instance will have to be a global variable. This is not at all ideal, but the functionality of the mouse event handler in `turtle` is so limited that we have no other choice.

Note: You are not allowed to modify the `score.py` file.

Once the game has been won, additional mouse clicks should have no effect. The only thing left to do is to close the turtle screen window, which will end the program.

2.3 Grading

The assignment grade is based on these factors:

- 20%: attendance at problem-solving and results of problem-solving teamwork
- 15%: screen drawing accuracy
- 20%: responsiveness to mouse actions and messages
- 25%: functional behavior of game *model* behind the scenes
- 10%: The design promotes code reuse. This includes, but is not limited to
  - defining and reusing functions (See problem-solving session work.)
  - not using any global variables besides the `Keeper` instance
- 10%: The code follows the style guidelines on the course web site.

Note: The submitted file must be a **Python 3 program**.

3 Submission

Submit your `pigdice.py` file to the MyCourses assignment before the due date.

- Go to the MyCourses assignment.
- Upload your `pigdice.py` file to submit your work into Lab2 dropbox. You can upload to myCourses as often as you like, but only the last submission counts.
- Check that your uploaded submission worked. If you forget a step, your work may not be submitted, and you will lose credits.

Double check by going to the dropbox and refreshing. Also, you should receive an email when you successfully submit.