Matching Puzzle Game

You must first complete the start menu tutorial before working on this one.

Settings Menu

Let’s allow the user to choose the number of rows and columns of tiles in the game before starting a game. We’ll do this with a settings menu that can be found from the start menu.

Displaying the Settings Menu

Inside `PuzzleData.py`'s `drawSettingsMenu` method, display the menu choices with this code:

```python
self.drawBackground(surface)
self.drawTextCenter(surface, "Settings", self.text_color,
                   self.width/2, self.font_height, self.font)
row_str = "Rows: %d (up or down)" % (self.rows)
self.drawTextCenter(surface, row_str, self.text_color,
                   self.width/2, 2*self.font_height, self.font)
col_str = "Columns: %d (left or right)" % (self.cols)
self.drawTextCenter(surface, col_str, self.text_color,
                   self.width/2, 3*self.font_height, self.font)
self.drawTextCenter(surface, "Back to menu (b)", self.text_color,
                   self.width/2, 5*self.font_height, self.font)
```

Now, make the settings menu drawing code be called if the game is in the settings menu mode, by adding this code to the `draw` method of `PuzzleData.py`.

```python
elif self.mode == MODE_SETTINGS_MENU:
    self.drawSettingsMenu(surface)
```

Add this after the if, but before the else statements. Look at the code in `draw` to be sure that the pattern looks correct after you add this code.

Handling User Choices

When the menu is displayed, we want the user to use the arrow keys to change the value of the rows and cols datamembers. We must handle the use input in the `evolveSettingsMenu` method of `PuzzleData.py`.

```python
if pygame.K_UP in newkeys:
    if self.rows < self.max_rows:
        self.rows += 1
elif pygame.K_DOWN in newkeys:
    if self.rows > self.min_rows:
        self.rows -= 1
elif pygame.K_RIGHT in newkeys:
    if self.cols < self.max_cols:
        self.cols += 1
elif pygame.K_LEFT in newkeys:
    if self.cols > self.min_cols:
        self.cols -= 1
elif pygame.K_b in newkeys:
    self.mode = MODE_START_MENU
```

Notice that this code will change the game mode back to the start menu if the user chooses. Otherwise, the user’s choices are used to set the number of tiles in the game, within limits.

We must also cause the settings menu handling code to be called when in the settings menu mode, by adding this code to the `evolve` method in `PuzzleData.py`.

```python
elif self.mode == MODE_SETTINGS_MENU:
    self.evolveSettingsMenu(keys, newkeys, buttons, newbuttons, mouse_position, dt)
```

Just like in the `draw` method, be sure that the indentation and pattern of the code looks correct.
Accessing the Settings Menu

To access the settings menu, we must add an entry to the start menu to allow the user to move to the settings menu.

Let’s display the settings menu option in the start menu with this code in `drawStartMenu` of `PuzzleData.py`:

```python
self.drawTextCenter(surface, "Settings (s)", self.text_color,
                   self.width/2, 2*self.font_height, self.font)
```

You’ll need to change the drawing of the quit option to a lower part of the screen by changing the 2 to a 3.

Finally, we’ll need to handle the user’s choice to enter the settings menu by added the option in `evolveStartMenu` of `PuzzleData.py`:

```python
elif pygame.K_s in newkeys:
    self.mode = MODE_SETTINGS_MENU
```

Be sure to place this correctly in that function.

Now you can run the program and see the settings menu.

Run the program several times. Check that the rows and columns don’t go past the minimum and maximum values. Also be sure that the number of tiles matches the value set in the menu.