Python Project: Snake

Concept

In this project you will create the game Snake. Common among old Nokia cell phones and school computers alike, this game will challenge you knowledge of class construction, pygame functionality, and event management.

Overall, the program will require you to program three files: snakeMain, snakeClass, and foodClass. snakeClass will require about 50 lines of code, foodClass will require about 25 lines of code, and snakeMain will require about 100 lines of code.

The greatest difficulty of this project will be managing to keep several files worth of information in your head at once. Try and get one part done and move on, instead of trying to finish all three files at once.

Instructions

snakeMain.py

1. First, you will want to follow along with the instructions provided in the file until you are required to construct the Food and Snake objects.
    a. Create the base __init__ for both Food and Snake objects once required
2. Afterwards, you will have to create the window using pygame.
    a. Open the pygame documentation website on your browser and search for display (https://www.pygame.org/docs/ref/display.html#pygame.display.set_mode)
    b. Once there, you will want to find `display.set_mode()` and save that as a variable with a name of your choosing
3. Finally, the last big section of this file falls under a singular `while running:` loop
    a. Pay close attention to your indentation on this segment, as you want to make sure all parts that should fall under the loop do so
    b. The segment labeled “Do this Later” should be done last, as it manages the end of the game and will take some time to finish
4. The key press conditions should be similar to ones previously done in earlier programs; use those to finish this section
5. Follow along with the instructions in the “Do this Later” part after you’ve finished everything else
    a. This stuff is going to be hard, so follow along as best you can. We will go over the work together. Make liberal use of pygame documentation for pygame’s Font

function (https://www.pygame.org/docs/ref/font.html?highlight=font#module-pygame.font)

snakeClass.py

1. Follow along with the instructions in the file for the initialization of the variables
    a. When building windowHeight and windowWidth, get the size of the window using `pygame.display.get_surface().get_size()`
2. Constructing the `Move()` function is fairly easy, the main difficulty is making sure you have the snake moving in the correct direction.
    a. Remember, moving up on the screen is negative Y, moving down on the screen is positive Y
    b. The `tailQueue` should be a list that contains lists of integers; these lists are the coordinate points for each position of the tail.
    c. When `Move()` returns True, that means the snake has either hit a wall or its own tail, and that the game should be over. You do not need to return False otherwise, as you return a NoneType by default.

foodClass.py

1. Follow along with the instructions in the file for the initialization of the variables
    a. When building windowHeight and windowWidth, get the size of the window using `pygame.display.get_surface().get_size()`
2. When constructing the `spawnFood()` function, make sure you subtract 10 from the `windowWidth` and `windowHeight` before dividing both by 10
    a. The reason you do this is because the snake moves in units of 10 pixels. The `eatFood()` function checks if the position of the food and the snake is the exact same. If the food were on a coordinate not divisible by 10, then there'd be no way for the snake to eat the food and it would inevitably starve to death. How sad.
3. Finally, `eatFood()` is a fairly straightforward function. If the snake and the food are on the same coordinate point, call `spawnFood()` and return False.

Code

snakeMain.py

```
import pygame
from snakeClass import Snake
from foodClass import Food

def main():

    """
    Initialize pygame's display and font functions
    """



    """
    Create variables for:
        windowWidth = An integer that is the total width of the
screen
        windowHeight = An integer that is the total height of the
screen
        window = The pygame display set_mode that is passed a list
containing windowWidth and windowHeight
        windowColor = A pygame Color that takes a string as input
        clock = The pygame time that contains the initialization for
Clock()
        score = An integer that should be initialized to 0
        snake = A Snake() object
        food = A Food() object
        moveBuffer = An integer that should be initialized to 2
        running = A boolean that should be initialized to True
    """



    """
    Set the caption of the window to print the title of the program
and the score
    Fill the window with windowColor
    Update the display
    Use food to call spawnFood()
    Draw a rectangle in the window at the snake's position with a
width and length of 10
    """
```

```
    """
    Create a while running loop:
        Update the display
        Fill the window with windowColor
        Call tick(60) using clock
        If the move buffer equals 0:
            If snake.Move() returns True:
                pygame.quit()
                !!! DO THIS LATER !!!

                Delete pygame.quit()

                create variables for endFont, endText, and
endTextRect

                Set endFont equal to a pygame Font, giving it a font
type to use and a font size
                Set endText equal to the rendered endFont, giving it
"Game Over" as input for the string
                Set endTextRect equal to the rectangle from endText
                Then, set the center of endTextRect to half the
windowWidth and a quarter of the windowHeight

                Repeat this process for scoreFont and menuFont

                Update the display

                Create a while running loop:
                    Get each event from pygame:
                        If a button is pressed:
                            And that button is escape:
                                Set running equal to False
                            If the button is R:
                                call main()

            Set moveBuffer to equal 2
        Otherwise:
            Subtract one from moveBuffer
        Draw a rectangle in the window at the food's position with a
width and length of 10
        Draw a rectangle in the window at the snake's position with a
width and length of 10
        For each segment in tailQueue:
```

```
            Draw a rectangle in the window at the segment's position
with a width and length of 10
        If moveBuffer equals 0:
            Create a variable called foodNotEaten and make it equal
to the result of food's eatFood()
            If foodNotEaten is False:
                pop() the first element in tailQueue
            Otherwise:
                Increase score by 1
                Update the display caption to reflect the new score
                (Note: Use 'Your Caption Here {0}'.format(score) to
do this easily)
                (Note: The {0} will be replaced by whatever is inside
format())
    """


    """
    Inside the while running loop:
        Create a for loop checking each pygame event:
            If the event is a keydown:
                If the key is Escape:
                    Set running equal to False
                If the key is W:
                    Set snake's direction to 'North'
                If the key is A:
                    Set snake's direction to 'West'
                If the key is S:
                    Set snake's direction to 'South'
                If the key is D:
                    Set snake's direction to 'East'

    Once the while loop is done, quit pygame
    """


    """
    Inside the !!! DO THIS LATER !!! part:
        Create an endFont, scoreFont, and menuFont variables
        ...
    """

if __name__ == "__main__":
    main()
```

snakeClass.py

```
import pygame

class Snake:
    def __init__(self):

        """
        Create variables for the position, tailQueue, direction,
headColor, tailColor, windowWidth and windowHeight
            position = A list that contains two integers X and Y
           tailQueue = A list that starts empty and will contain the
coordinates of the tail segments
           direction = A string that contains the direction the snake
is facing
                North --> Up
                East  --> Right
                South --> Down
                West  --> Left
           headColor = A pygame.Color that takes a string as input
           tailColor = A pygame.Color that takes a string as input
         windowWidth = The width of the window
        windowHeight = The height of the window
        """

    def Move(self):

        """
        Setup a match/case where you're matching the direction to
four different cases:
                North:
                    Append to the tailQueue a list containing the
snake's X and Y coordinates
                     (Note: this should be a new list containing
self.position[0] and self.position[1])
                    Subtract 10 from snake's Y coordinate
                    Check each tail segment in the tailQueue using a
for loop
                        If the tail segment's coordinates equals the
snake's coordinates:
                            Return True
                    If the snake's Y position is less than 0:
                        Return True

                East:
```

```
                    Append to the tailQueue a list containing the
snake's X and Y coordinates
                    (Note: this should be a new list containing
self.position[0] and self.position[1])
                    Add 10 to the snake's X position
                    Check each tail segment in the tailQueue using a
for loop
                        If the tail segment's coordinates equals the
snake's coordinates:
                            Return True
                    If the snake's X position is greater than the
windowWidth - 10:
                        Return True

                South:
                    Append to the tailQueue a list containing the
snake's X and Y coordinates
                    (Note: this should be a new list containing
self.position[0] and self.position[1])
                    Add 10 to the snake's Y coordinate
                    Check each tail segment in the tailQueue using a
for loop
                        If the tail segment's coordinates equals the
snake's coordinates:
                            Return True
                    If the snake's Y position is greater than the
windowHeight - 10:
                        Return True

                West:
                    Append to the tailQueue a list containing the
snake's X and Y coordinates
                    (Note: this should be a new list containing
self.position[0] and self.position[1])
                    Subtract 10 from snake's X coordinate
                    Check each tail segment in the tailQueue using a
for loop
                        If the tail segment's coordinates equals the
snake's coordinates:
                            Return True
                    If the snake's X position is less than 0:
                        Return True
        """

if __name__ == "__main__":
```

```
print()
```

foodClass.py

```
import pygame
import random

class Food:
    def __init__(self):

        """
        Create variables for color, position, windowWidth and
windowHeight
                    Color = A pygame.Color that takes a string as
input
                Position = A list that contains two integers X and Y
             windowWidth = The width of the window
            windowHeight = The height of the window
        """

    def spawnFood(self):

        """
        Create two variables for the new X and Y positions of the
food
            randX = a random integer from 0 to the windowWidth minus
10, and then divided by 10
            randY = a random integer from 0 to the windowHeight minus
10, and then divided by 10
        """

        """
        Multiply randX and randY by 10
        """

        """
        Set food's position to be equal to a list containing randX
and randY
        """

    def eatFood(self, snakePos):

        """
        Create an if statement checking if snakePos is equal to the
food's position
            Then call spawnFood and return False
        Otherwise return True
```

"""