```
# Navya Aenugu: dby4ka
# Vaibhav Choudhary: gau7jk

"""
Description of game:

This game is called the Emoji Chase. There are two players and they must aim to get more red squares than each other. If a player hits a blue square, he/she loses.
The red and blue squares are constantly being created, so players must keep moving around to avoid the blue squares being created where their positions are. Players also cannot move to the left.
Click the space bar to begin the game.
Player 1 uses the d, w, and s keys to move and player 2 uses the up, down, and right arrows.
Player 1 (laughing emoji) has a blue score at the top and player 2 (starry eye emoji) has a purple score at the top.
When the game ends, you can hold down the space bar to see who won the game if you would like to.


3 basic features:
1. user input
- players click the space bar to restart from game over
- player 1 uses the "w", "s", and "d" keys to move
- player 2 uses the right, up, and down arrow keys to move

2. game over
#NEW FEATURE - 60 seconds instead of 15 seconds in timer
- the game ends when the timer runs out (60 seconds after the start time)
- the game can also end if a player hits a blue square
- a game over screen is shown when the game ends if
```

```
 the player holds down the space bar

3. graphics/images
#NEW FEATURE - sprites for players instead of squares
- player 1 is represented as a laughing emoji
- player 2 is represented as a starry eyed square
- the obstacles that players cannot hit are the blue squares
- there is a timer countdown that shows how much time is left
- the collectibles are represented as red squares


4 additional features:
#NEW FEATURE - sprite animations
1. sprite animations - the players are both sprite representations of emojis and they turn into upside down smiley faces if they hit obstacles (blue squares)
2. collectibles: players must collect as many of the red squares as they can (trying to get more than opponent)
3. timer: the timer will count down from 15 seconds; whoever has the most red squares at the end of this time wins unless
a player loses by being knocked out by the other player or by hitting the blue square
4. two players simultaneously: player 1 is a yellow square and player 2 is a blue square and they can knock each other
out of the game by bumping into each other (whoever bumps the other one out wins); player 1 uses the
"d", "s", and "w" keys and player 2 uses the right, up, and down arrows

"""

"""
#NEW FEATURES SINCE PC #1

```

```
1. Sprite animation - each player is represented by a different emoji that changes its expression when it loses (both players are animated)
2. No more restart from game over feature (replaced by sprite animation feature)
3. Sprite animations replaced the yellow/cyan boxes that were characters
4. Players are allowed to overlap each other without it being considered a loss
5. Collectibles and obstacles are being made every second, not just constant from start to end (increases difficulty of game)

4 additional features (updated): collectibles, timer, two players simultaneously, sprite animation
"""

# UVAGE CODE TO RUN THE GAME
import uvage
import random


camera = uvage.Camera(800, 600)
game_on = False
timer = 60
p1_score = 0
p2_score = 0
# the overall variable is just so that the collectibles/score/obstacles are updated every second, not every tick
overall = 360
game_off = False

# loading sprite sheet and creating character icons
p_load = uvage.load_sprite_sheet("p1_smile.png", 5, 5)
p1 = uvage.from_image(100, 200, p_load[11])
p2 = uvage.from_image(100, 500, p_load[2])

# these lists will be appended to so we can create the obstacles and collectibles
```

```python
obstacles = []
collectibles = []

def tick():
    """
    The tick function runs 60 times a second and contains all the functions for the game to run.
    :return: none
    """
    global game_on
    global timer
    global p1_score
    global p2_score
    global overall
    global game_off

    overall -= 1

    if game_on == True:
        # timer decreases by 1 every second
        if overall % 60 == 0:
            timer -= 1
    # game ends when timer is at 0
    if timer == 0:
        game_on = False

    def player_functions():
        """
        This function states which keys must be pressed for the players to begin the game and move around.
        :return: none
        """
        # player 1 (p1)
        global game_on
        if uvage.is_pressing("space"):
            game_on = True
        if uvage.is_pressing("d"):
            if p1.x != 800:
                p1.x += 10
        if uvage.is_pressing("w"):
            if p1.y != 0:
                p1.y -= 10
        if uvage.is_pressing("s"):
            if p1.y != 600:
                p1.y += 10

        #player 2 (p2)
        if uvage.is_pressing("right arrow"):
            if p2.x != 800:
                p2.x += 10
        if uvage.is_pressing("up arrow"):
            if p2.y != 0:
                p2.y -= 10
        if uvage.is_pressing("down arrow"):
            if p2.y != 0:
                p2.y += 10


    def create_collectibles():
        """
        This function creates the red squares which the players can collect to earn points.
        :return: none
        """
        if game_on == True and overall % 60 == 0:
            position_x = random.randint(300, 700)
            position_y = random.randint(0, 600)

            collectibles.append(uvage.from_color(position_x, position_y, "red", 20, 20))

    def collectibles_disappear():
        """
        This function makes the red squares "disappear" if the players touch them and it adds 1 point to the player's score.
        :return: none
        """
        global p1_score
        global p2_score
        global overall

        for collectible in collectibles:
            if p1.right_touches(collectible):
                collectible.x = 900
                collectible.y = 700
                p1_score += 1


        for collectible in collectibles:
            if p2.right_touches(collectible):
                p2_score += 1
                collectible.x = 900
                collectible.y = 700

    def create_obstacles():
        """
        This function creates a new blue square (obstacle) which the players must avoid every time the tick function runs.
        :return: none
        """
        global obstacles
        position_x = random.randint(300, 700)
        position_y = random.randint(0, 600)
        if game_on == True and overall % 120 == 0:
            obstacles.append(uvage.from_color(position_x, position_y, "blue", 20, 20))

    def obstacle_touch():
        """
        This function makes a player lose if he or she touches an obstacle. When the player loses, the icon changes into an upside down smiley face.
        :return: none
        """
        global game_on
        global first_obstacle
        global second_obstacle
        global third_obstacle
        global p1
        global p2
        for each in obstacles:
            if p1.touches(each):
                p1 = uvage.from_image(each.x, each.y, p_load[14])
                camera.draw(uvage.from_text(400, 300, "Game Over; Player 2 Won!", 30,"green", bold = True))
                game_on = False

        for each in obstacles:
            if p2.touches(each):
                p2 = uvage.from_image(each.x, each.y, p_load[14])
                camera.draw(uvage.from_text(400, 300, "Game Over; Player 1 Won!", 30, "green", bold =True))
                game_on = False



    #def player_touch():
     #   global game_on

      #  if p1.touches(p2):
       #     camera.clear("white")
        #    camera.draw(uvage.from_text(400, 300, "Game Over; Player 2 Won!", 30, "green", bold=True))
         #   game_on = False

#        if p2.touches(p1):
 #           camera.clear("white")
  #          camera.draw(uvage.from_text(400, 300, "Game Over; Player 1 Won!", 30, "green", bold=True))
   #         game_on = False


    def create():
        """
        This function creates the visuals for the
```

```
game, such as the scores, text, and timer.
        :return: none
        """
        camera.clear("black")
        camera.draw(p1)
        camera.draw(p2)
        camera.draw(uvage.from_text(100, 30, str(
p1_score), 50, "blue", bold = True))
        camera.draw(uvage.from_text(100, 70, str(
p2_score), 50, "purple", bold = True))
        camera.draw(uvage.from_text(400, 250,"
Press the space bar to begin the game. Player 1 (
yellow) must use the d, w, and s keys to play and
player 2 (cyan) must use the up, down, and right
keys." , 13, "white", bold=True))
        camera.draw(uvage.from_text(400, 275, "
Collect more red squares than the other player in
the alloted time (15 seconds). If you touch a blue
 square or the other player, you lose. Keep in
mind, you cannot move to the left!", 13, "white",
bold=True))
        camera.draw(uvage.from_text(400, 300, "Be
careful, an obstacle may pop up right below where
your icon is. Make sure to be constantly moving!"
, 15, "red", bold = True))
        camera.draw(uvage.from_text(400, 325, "To
see who won the game after it ends, hold down the
space bar", 15, "red", bold = True))
        camera.draw(uvage.from_text(700, 30, str(
timer), 50, "red", bold = True))
    def make_parts():
        """
        This function creates the obstacles (blue
) and the collectibles (red) for the game.
        :return: none
        """
        global obstacles
        global collectibles

        for each in obstacles:
            camera.draw(each)

        for part in collectibles:
            camera.draw(part)

    create()
    player_functions()
    if game_on == True:
        make_parts()
        create_collectibles()
        collectibles_disappear()
        create_obstacles()
        obstacle_touch()

    #if game_on == False:
        game_off = True

    #if game_off == True:
    #    camera.clear("white")
    #    camera.draw(uvage.from_text(400, 300, " Game over!", 100, "red"))
    camera.display()

ticks_per_second = 60
uvage.timer_loop(ticks_per_second, tick)



```