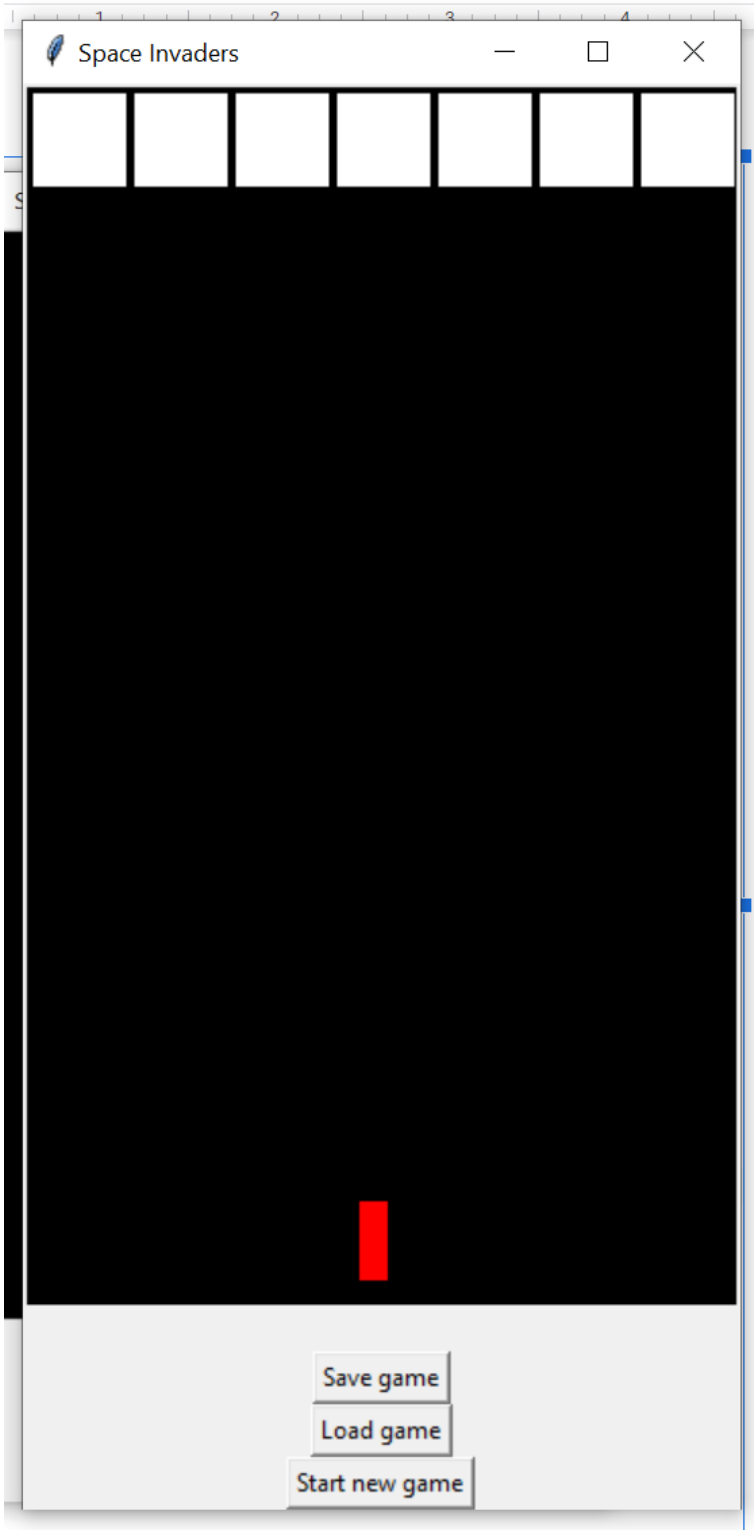
Joshua Isakson CS 162 Final Project
Recreated Space Invaders.

Sorry for being MIA. I got behind early due to work and other classes and thought I would be able to catch up until it was too late.

Default state



Start new game was pressed

Moving left and right with the arrow keys and shooting with space. The bullets collide with the white blocks and destroy them. You can save the game but it only currently saves the player position and white square positions not the squares destroyed state. The squares were supposed to move down slowly but there was a bug that I couldn't figure out and I ran out of time. I also didn't have enough time to format anything into the pycodestyle or add any docstrings or comments or make any tests.

```
def load_game(self):
    if not os.path.exists("game_state.txt"):
        print("Error: game state file not found")
        return

    with open("game_state.txt", "r") as f:
        values = []
        for line in f:
            values.extend([float(x) for x in line.strip().split(",")])

        if len(values) >= 4:
            x0, y0, x1, y1 = values[:4]
            if x0 is not None and y0 is not None and x1 is not None and y1 is not None:
                self.defender.x0 = x0
                self.defender.y0 = 550
                self.defender.x1 = x1
                self.defender.y1 = 590
            self.canvas.coords(self.defender.id, self.defender.x0, self.defender.y0, self.defender.x1, self.defender.y1)
```

Uses conditionals to provide a level of input validation on the game's save file and file IO.

```
def move_left(self, event):
    x0, y0, x1, y1 = self.canvas.coords(self.defender.id)

    self.canvas.move(self.defender.id,-10, 0)

    self.defender.x0 = x0 -10
    self.defender.x1 = x1 -10
def move_right(self, event):
    x0, y0, x1, y1 = self.canvas.coords(self.defender.id)
    self.canvas.move(self.defender.id,10, 0)

    self.defender.x0 = x0 + 10
    self.defender.x1 = x1 + 10
```

User IO, functions

```
class Space_invader:
    def __init__(self):
        self.start_time = time.time()

        self.refresh_rate = 20

        self.window = tk.Tk()
        self.window.title("Space Invaders")
        self.canvas = tk.Canvas(self.window, width=350, height=600)
        self.canvas.grid(column=0, row=0)
        self.canvas.configure(bg="black")

        defender_id = self.canvas.create_rectangle(165, 550, 180, 590, fill="red")
        self.defender = defender.Defender(defender_id)

        self.time_label = tk.Label(self.window, text="")
        self.time_label.grid(column=0, row=1)


        #adds the button for saving,loading and starting a new game
        save_button = tk.Button(self.window, text="Save game", command=self.save_game)
        save_button.grid(column=0, row=2)
        load_button = tk.Button(self.window, text="Load game", command=self.load_game)
        load_button.grid(column=0, row=3)
        Start_new_game_button = tk.Button(self.window, text="Start new game", command=self.start_new_game)
        Start_new_game_button.grid(column=0, row=4)



        self.defender.y1 = 590
        self.bullets = []


        self.invader_cluster = []


        self.window.bind("<Left>", self.move_left)
        self.window.bind("<Right>", self.move_right)
        self.window.bind("<space>", self.shoot)
        self.window.after(self.refresh_rate, self.update_bullets)
        self.window.after(self.refresh_rate, self.update_time_label)



    def start_new_game(self):
        invader_00 = self.create_invader(5, 5, 50, 50, )
        invader_01 = self.create_invader(55, 5, 100, 50,  )
        invader_02 = self.create_invader(105, 5, 150, 50,  )
        invader_03 = self.create_invader(155, 5, 200, 50,  )
        invader_04 = self.create_invader(205, 5, 250, 50,  )
        invader_05 = self.create_invader(255, 5, 300, 50,  )
        invader_06 = self.create_invader(305, 5, 350, 50,  )
        self.invader_cluster.append(invader_00)
        self.invader_cluster.append(invader_01)
        self.invader_cluster.append(invader_02)
        self.invader_cluster.append(invader_03)
        self.invader_cluster.append(invader_04)
        self.invader_cluster.append(invader_05)
        self.invader_cluster.append(invader_06)
```

```
def invader_cluster_shuffle(self):
    for invader in self.invader_cluster:
        invader.y0 += 1
        invader.y1 += 1

    #for invader in self.invader_cluster:
        #if invader.y1 >= self.canvas.winfo_height():
            #print("game over")
            #return
    #if all(invader.destroyed for invader in self.invader_cluster):
        #print("win")
        #return
    #self.window.after(self.refresh_rate, self.move_invaders_down)
```