CPSC 8700
FINAL PROJECT

TRAIL OF TERROR

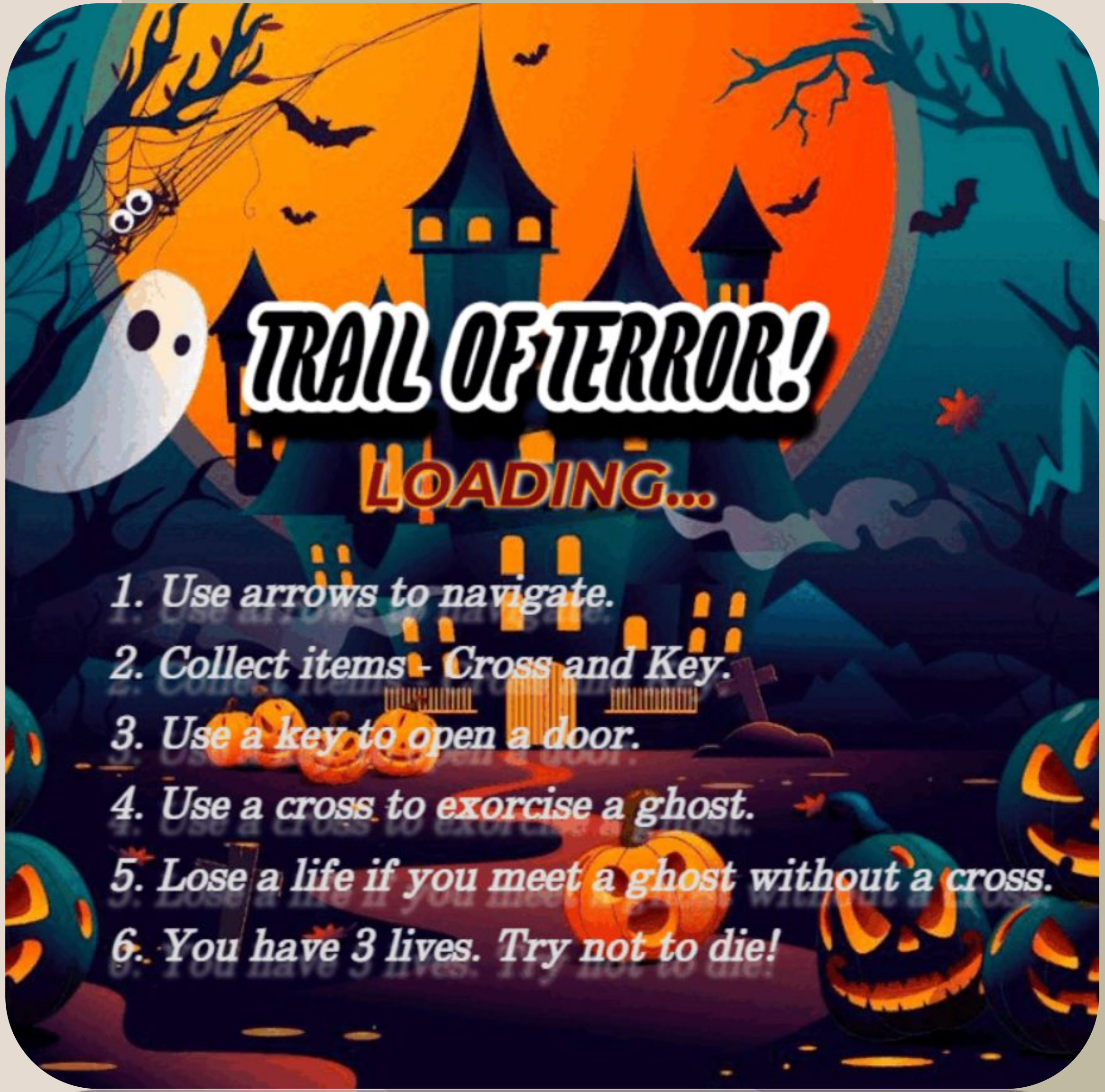
ADITHYA RAVI
ANJANA SRIRAM
ASHWINI BALASUBRAMANIAN
SRIVATSA KANDALAM

agenda

Introduction

Trail of Terror is a fun game in which the objective is to escape the maze you are trapped in without being killed. You first should locate the key to get out of the door. To exorcise the ghosts and keep your head from getting snapped, you'll need a weapon, which in this context is a cross.

Collect the cross, exorcise the ghosts, find the way out of the maze through numerous doors. Beware, you have got only 3 lives!



1. Use arrows to navigate.
2. Collect items - Cross and Key.
3. Use a key to open a door.
4. Use a cross to exorcise a ghost.
5. Lose a life if you meet a ghost without a cross.
6. You have 3 lives. Try not to die!

DESIGN PATTERNS:

Design patterns are conventional answers to common software design problems. Each pattern is similar to a blueprint that you can modify to tackle a specific design problem in your code.

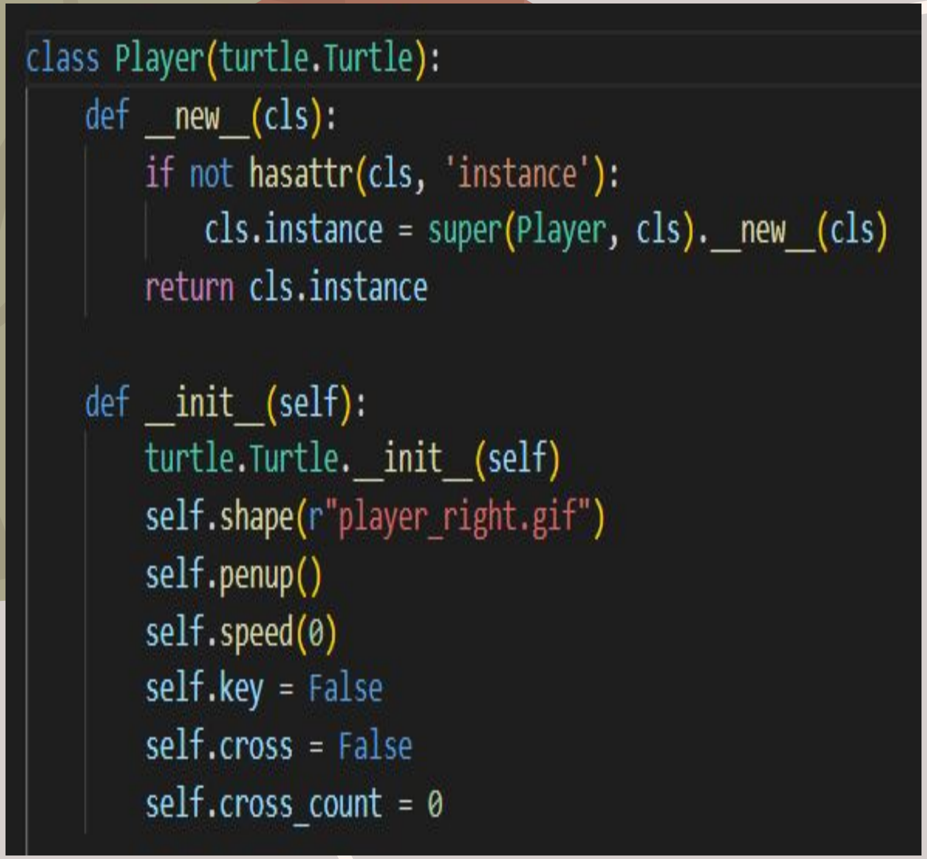
The two design patterns used in our code are:

- Singleton method
- Factory method

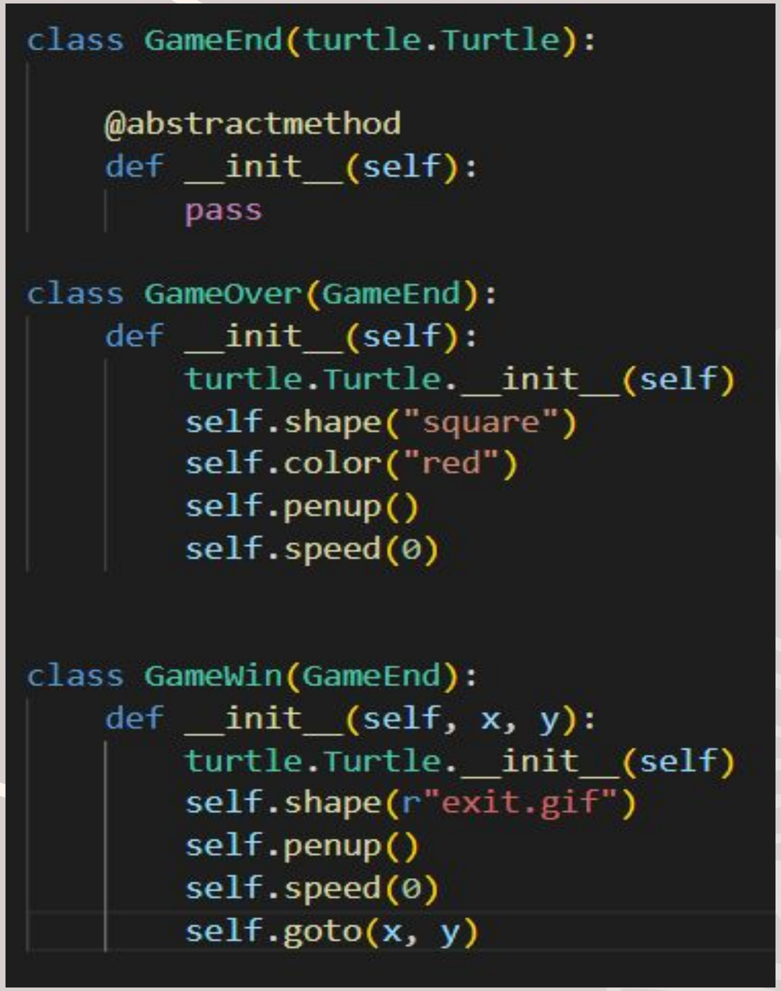
The Singleton design pattern ensures that a class has only one instance while offering a global access point to this instance.

A Factory Method is a creational design pattern that provides an interface for creating objects in a superclass while allowing subclasses to choose the type of objects created.

FACTORY PATTERN



```
class Player(turtle.Turtle):
    def __new__(cls):
        if not hasattr(cls, 'instance'):
            cls.instance = super(Player, cls).__new__(cls)
        return cls.instance

    def __init__(self):
        turtle.Turtle.__init__(self)
        self.shape(r"player_right.gif")
        self.penup()
        self.speed(0)
        self.key = False
        self.cross = False
        self.cross_count = 0
```

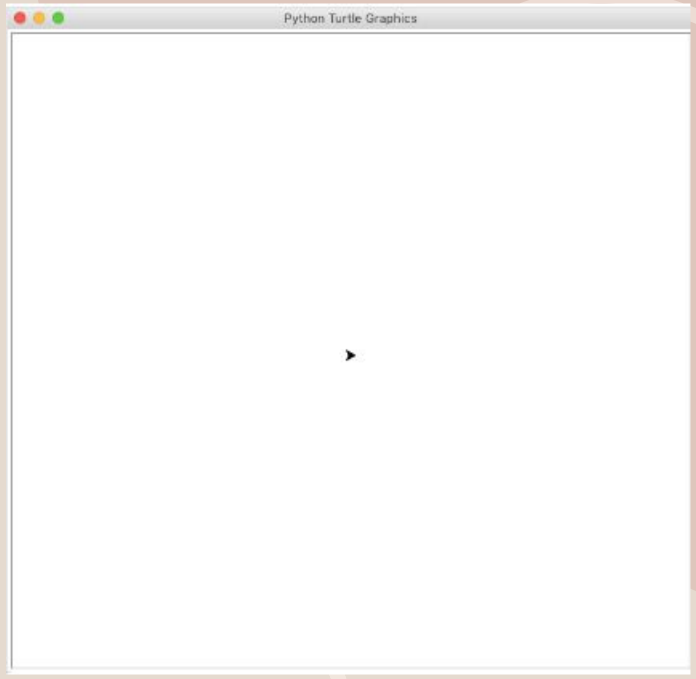


```
class GameEnd(turtle.Turtle):

    @abstractmethod
    def __init__(self):
        pass

class GameOver(GameEnd):
    def __init__(self):
        turtle.Turtle.__init__(self)
        self.shape("square")
        self.color("red")
        self.penup()
        self.speed(0)


class GameWin(GameEnd):
    def __init__(self, x, y):
        turtle.Turtle.__init__(self)
        self.shape(r"exit.gif")
        self.penup()
        self.speed(0)
        self.goto(x, y)
```

SINGLETON PATTERN

Turtle package

- *Turtle* is a pre-installed Python module that allows users to draw drawings and shapes on a virtual canvas. The onscreen pen used for drawing is known as the *turtle*, and it is this that gives the library its name. In short, the Python turtle library provides new programmers with a fun and interactive approach to learn about Python programming.



DEMO