Turtle Race Game

1. Project Overview

Turtle Race Game is a simple Python game where multiple turtles race toward a finish line. Each turtle moves a random distance in every turn. The first turtle to cross the finish line wins.

This project introduces students to Python through a visual and interactive game.

2. Features

- 4 racing turtles
- Random movement
- Finish line
- Winner detection
- Player prediction
- Score
- Multiple races
- Simple Turtle graphics

3. Technologies

- Python 3
- Turtle
- Random
- Loops
- Functions
- Conditions
- Lists
- Variables

No external packages are required.

4. Folder Structure

```
turtle_race/
□□□ main.py
□□□ game.py
□□□ README.md
```

5. How the Game Works

The game creates four turtles:

```
Red
Blue
Green
Yellow
```

Each turtle starts from the same side of the screen.

During the race, every turtle moves a random number of pixels.

For example:

```
Red    □ □ □ □ □ □ □
Blue   □ □ □ □ □ □ □ □ □
Green  □ □ □ □ □ □
Yellow □ □ □ □ □ □ □ □ □ □
```

The first turtle to reach the finish line becomes the winner.

The `random` module is used so that the result is different each time the game runs.

6. Complete Game Code

game.py

```
import random
import turtle


class TurtleRace:
    def __init__(self):
        self.screen = turtle.Screen()
        self.screen.title("Turtle Race Game")
        self.screen.setup(width=900, height=600)
        self.screen.bgcolor("white")

        self.finish_x = 350
        self.turtles = []

        self.colors = ["red", "blue", "green", "orange"]
        self.y_positions = [150, 50, -50, -150]

        self.create_turtles()
        self.draw_finish_line()

    def create_turtles(self):
        for color, y in zip(self.colors, self.y_positions):
            racer = turtle.Turtle()
            racer.shape("turtle")
            racer.color(color)
            racer.penup()
            racer.goto(-350, y)
            self.turtles.append(racer)

    def draw_finish_line(self):
```

```
        line = turtle.Turtle()
        line.hideturtle()
        line.penup()
        line.goto(self.finish_x, 220)
        line.pendown()
        line.goto(self.finish_x, -220)

    def race(self):
        winner = None

        while winner is None:
            for racer in self.turtles:
                distance = random.randint(5, 20)
                racer.forward(distance)

                if racer.xcor() >= self.finish_x:
                    winner = racer.color()[0]
                    break

        return winner

    def close(self):
        self.screen.mainloop()
```

7. Main Program

main.py

```
from game import TurtleRace


def main():
    print("================================")
    print("       TURTLE RACE GAME")
    print("================================")
```

```
print("\nChoose your turtle:")
print("1. Red")
print("2. Blue")
print("3. Green")
print("4. Orange")

choices = {
    "1": "red",
    "2": "blue",
    "3": "green",
    "4": "orange"
}

prediction = input("\nWhich turtle will win? ")

while prediction not in choices:
    print("Please choose 1, 2, 3 or 4.")
    prediction = input("Which turtle will win? ")

selected_color = choices[prediction]

game = TurtleRace()
winner = game.race()

print("\nWinner:", winner.upper())

if winner == selected_color:
    print("You predicted correctly!")
else:
    print("Better luck next time!")

game.close()
```

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

8. How to Run

Open the project folder in VS Code or another Python editor.

Run:

```
python main.py
```

The terminal will ask the student to predict the winning turtle.

Then the Turtle window will open and the race will begin.

9. What Students Learn

`random`

```
random.randint(5, 20)
```

Generates a random movement distance.

`for` loop

```
for racer in self.turtles:
```

Runs the same instructions for every turtle.

`while` loop

```
while winner is None:
```

Keeps the race running until somebody wins.

Condition

`if racer.xcor() >= self.finish_x:`

Checks whether a turtle has reached the finish line.

Function

`def race(self):`

Keeps the race logic organized inside a reusable function.

10. Student Challenge

After completing the basic game, ask students to add:

1. A fifth turtle
2. A different background color
3. A larger finish line
4. Best of 3 races
5. A scoreboard
6. A custom turtle shape
7. A message showing the winner inside the Turtle window

Final Challenge

Let each student customize their own version:

> Change the turtle colors, title, background, race distance and rules.

This turns the project from a simple coding exercise into a small game that students can personalize.

11. How Frontend and Backend Work

There is no separate frontend/backend application in this project.

Turtle graphics handles the visual interface.

`game.py` handles the race logic, turtle movement and winner detection.

`main.py` handles user input and starts the game.

12. Basic Testing

Test the following:

- Enter `1`, `2`, `3` or `4`.
- Try an invalid choice.
- Run the game multiple times.
- Check whether different turtles win.
- Check whether the selected turtle is correctly compared with the winner.
- Close the Turtle window after the race.

13. Small Improvements

Students can later add:

- Countdown: `3... 2... 1... GO!`
- Race sound
- Better graphics
- Multiple rounds

- Player names
- Leaderboard
- Difficulty levels
- Turtle obstacles
- Power-ups

Visit haas.dev for more resources and guides.