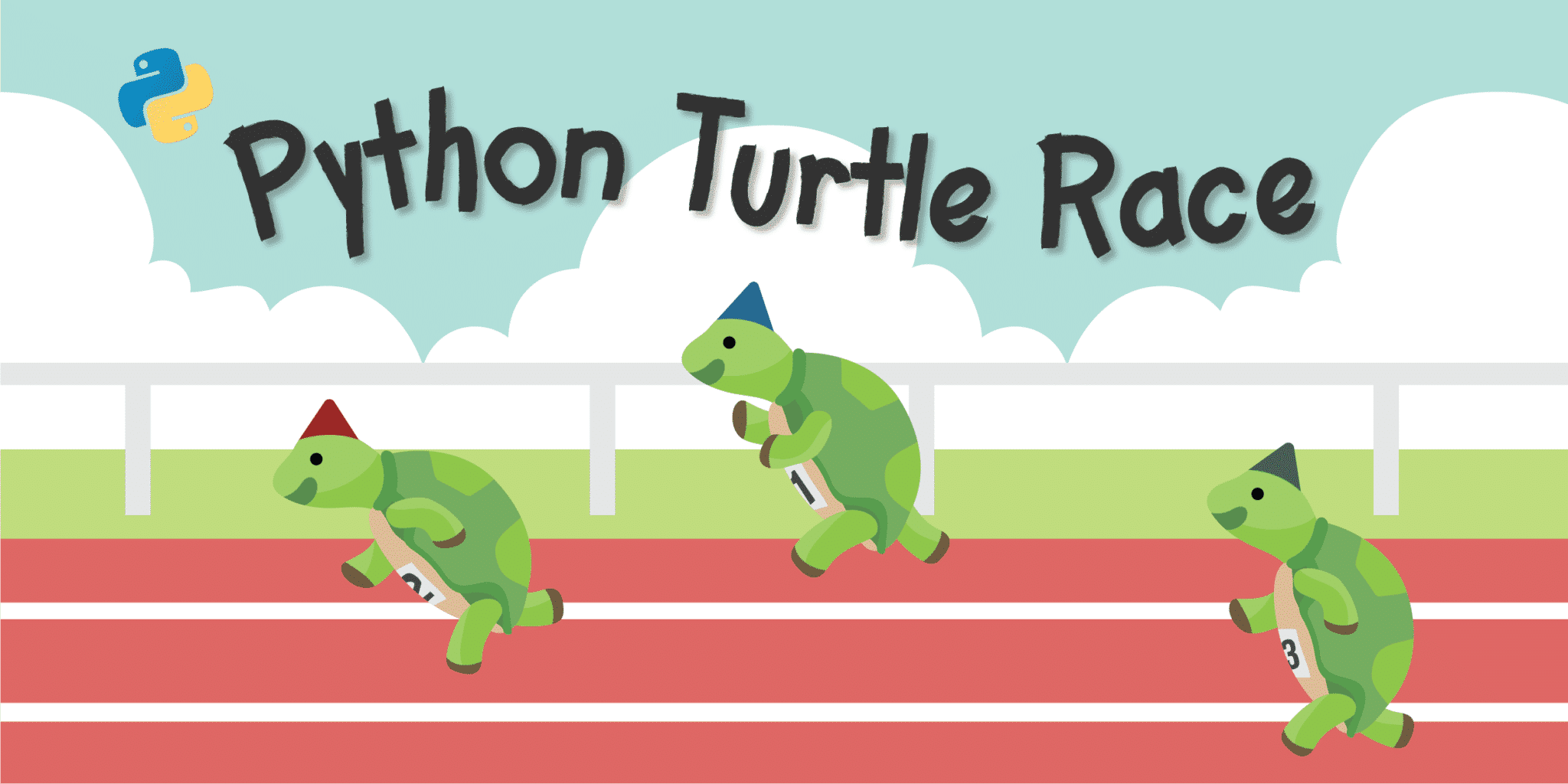
**Mini-Project-2: Turtle Race**

****



**Introduction**

The turtle race project combines elements of programming, graphics, and user interaction to create a fun and interactive game. It's relevant in the current world in several ways:

1. **Programming Education:** Projects like this offer a playful and hands-on approach to learning programming concepts. It introduces beginners to basic programming structures like loops, conditional statements, and event handling. Learning through games makes coding more engaging and accessible, especially for students or newcomers to programming.
2. **Interactive User Experience:** It demonstrates the integration of graphical user interfaces (GUI) using libraries like Tkinter alongside visual output with the Turtle graphics library. This combination showcases how programming can create interactive experiences for users, which is fundamental in various software applications today.
3. **Gamification and Entertainment:** It illustrates how coding can be used to create entertaining and engaging content. Gamification is a powerful tool in education, marketing, and various industries to increase user engagement and retention.
4. **Relevance in STEM Education:** Projects like this are particularly valuable in STEM (Science, Technology, Engineering, and Mathematics) education. They demonstrate practical applications of STEM concepts and encourage students to explore technology, fostering curiosity and interest in these fields
5. **Coding as a Creative Outlet:** This project showcases how coding can be a creative outlet. It allows individuals to express creativity by designing and implementing games, fostering problem-solving skills and innovation.

In today's world, where technology plays a pivotal role in various aspects of life, understanding programming concepts and their applications is increasingly valuable. Projects like the turtle race game not only teach coding but also demonstrate its relevance in creating engaging experiences and solving real-world problems.

**Problem Statement:**

Develop an interactive Python-based Turtle Race program that employs Turtle graphics and a Tkinter GUI to create an engaging gaming experience where users select a turtle and participate in a thrilling race.

**Major Functions:**

**1.** **Start Race Function:**

- Initialize the Turtle Race environment, setting up the graphical window using Tkinter.

- Position the turtles at the starting line and assign colors to each turtle.

- Control the race, moving the turtles forward randomly until one reaches the finish line.

- Determine the winning turtle and display the result based on the user's chosen color.

**2.** **Set Color Function:** - Retrieve the user's selected turtle color from the Tkinter dropdown menu.

- Hide the Tkinter window and initiate the race with the chosen color.

**3.** **Tkinter Window Setup:**

- Create a Tkinter window for the color selection interface.

- Display a dropdown menu for users to select their preferred turtle color.

- Include a 'Submit' button to start the race upon color selection.

**4.** **Race Control and Display:**

- Control the race flow through a while loop, moving turtles and determining the winner.

- Update and display the user's score based on the race outcome.

- Prompt the user to play again or exit the game.

**5.** **Turtle Graphics Management:**

- Utilize Turtle graphics to represent the racing turtles on the screen.

- Position the turtles at the start and manage their movement during the race.

- Clear and hide the turtles after each race to prepare for the next round.

The objective is to create an interactive gaming environment that combines graphical visualization through Turtle graphics and user interaction via Tkinter, allowing users to engage in an entertaining turtle race experience.

**Algorithm:**

Following is the algorithm breakdown of the Turtle race project:

1. **Initialization:**
   * Import necessary modules: Turtle for graphics, random for randomizing movement, and Tkinter for the UI.
   * Define a function start\_race to manage the entire race process.
   * Create a Tkinter window for color selection and game initiation.
2. **Race Setup:**
   * Initialize the Turtle screen for the race with a title and specific dimensions.
   * Define initial positions (y\_positions) for the turtles and a list (colors) of available turtle colors.
   * Create six turtles, assigning each a color and positioning them on the starting line.
3. **Race Execution:**
   * Start the race loop (while is\_race\_on).
   * Iterate through each turtle and move them forward randomly between 0 to 10 units.
   * If any turtle crosses the finish line (xcor() > 230), end the race.
   * Determine the winning turtle's color and compare it to the user's chosen color, updating the user's score accordingly.
4. **Post-Race Actions:**
   * Clear and hide all turtles after the race.
   * Display the user's current score.
   * Prompt the user to play again (yes/no).
   * If yes, reset the turtles' positions for a new round and display the color selection window.
   * If no, display a closing message and close both the turtle graphics window and the tkinter window.
5. **User Interface:**
   * Define a function (set\_color) to get the user's selected color and start the race accordingly.
   * Create a tkinter window with a dropdown menu for color selection and a submit button to initiate the race.
   * Run the tkinter event loop to manage user interactions.
6. **Conclusion:**
   * The algorithm orchestrates the turtle race, managing race setup, execution, user interactions, and post-race actions through a blend of Turtle graphics and Tkinter UI.

This algorithm provides a structured overview of the project's flow, from initialization to conclusion, outlining the sequential steps in managing the turtle race and user interactions.

**Project Abstract**

**Interactive Turtle Race**

This Python project combines the power of Turtle graphics and Tkinter to create an engaging turtle racing game with user interaction. The program opens a graphical window using Tkinter, allowing users to select a turtle color they believe will win the race.

Once the user selects a color and hits "Submit," the Turtle race begins in a separate window. The race involves six turtles of different colors moving forward randomly until one reaches the finish line.

Key Features:

The key features of this project include the following:

1. User Interaction: Through a Tkinter interface, users select their preferred turtle color.
2. Turtle Racing: Turtles move forward randomly until one reaches the finish line.
3. Gameplay: After the race concludes, users are informed whether their chosen turtle won or lost, earning points accordingly.
4. Continuity: Users have the option to play again, resetting the turtles' positions and showing the color selection window for another round.

The project not only demonstrates the usage of Turtle graphics for visual representation but also integrates a simple user interface through Tkinter for an interactive gaming experience.

**Source Code**

Following is the code of the Python Project:

from turtle import Turtle, Screen # Importing necessary modules for graphics

import random

from tkinter import \* # Importing the Tkinter library for the user interface

def start\_race(color, chosen\_color):

global user\_score

# Set up the screen for the turtle race

screen = Screen()

screen.title("Make Your Bet") # Title of the graphical window

screen.setup(width=500, height=400) # Setting up the size of the graphical window

# Initial positions for the turtles in the race

y\_positions = [-70, -40, -10, 20, 50, 80]

all\_turtles = [] # List to hold all the turtles

# Colors for the turtles available for selection

colors = ['red', 'orange', 'yellow', 'green', 'blue', 'purple']

# Create turtles for each color and position them on the screen

for turtle\_index in range(0, 6):

new\_turtle = Turtle(shape="turtle")

new\_turtle.color(colors[turtle\_index])

new\_turtle.penup()

new\_turtle.goto(x=-230, y=y\_positions[turtle\_index])

all\_turtles.append(new\_turtle) # Add each turtle to the list

is\_race\_on = True # Flag to control the race

user\_score = 0 # User's score for winning the race

# Start the race

while is\_race\_on:

for turtle in all\_turtles:

if turtle.xcor() > 230:

is\_race\_on = False # End the race if a turtle reaches the finish line

winning\_color = turtle.pencolor()

print(f"You chose the {chosen\_color} turtle.") # Display user's chosen turtle

if winning\_color == color:

print(f"You've won! The {winning\_color} turtle is the winner!")

user\_score += 10 # Update user score if they win

else:

print(f"You've lost! The {winning\_color} turtle is the winner!")

break

rand\_distance = random.randint(0, 10)

turtle.forward(rand\_distance) # Move the turtles forward randomly

# Clear and hide turtles after the race

for turtle in all\_turtles:

turtle.clear()

turtle.hideturtle()

print(f"Your current score is: {user\_score}") # Display user's score

# Ask the user if they want to play again

play\_again = input("Do you want to play again? (yes/no): ").lower()

if play\_again == "yes":

# Reset turtles' positions for a new round and show the color selection window

for turtle in all\_turtles:

turtle.goto(x=-230, y=y\_positions[all\_turtles.index(turtle)])

root.deiconify()

else:

print("Thanks for playing!") # End of the game message

screen.bye() # Close the turtle graphics window

root.destroy() # Close the tkinter window

def set\_color():

color = var.get()

chosen\_color = var.get() # Get the user's chosen color

root.withdraw() # Hide the tkinter window when the game starts

start\_race(color, chosen\_color)

# Setup the Tkinter window for color selection

root = Tk()

root.title("Turtle Race")

root.geometry("300x100") # Set the initial window size

var = StringVar(root)

var.set("Select the color of your turtle") # Text displayed in the color selection menu

option = OptionMenu(root, var, "red", "orange", "yellow", "green", "blue", "purple") # Dropdown menu with color options

option.pack() # Display the color selection menu

submit\_button = Button(root, text="Submit", command=set\_color) # Button to start the race

submit\_button.pack() # Display the button

root.mainloop() # Run the Tkinter event loop for user interaction

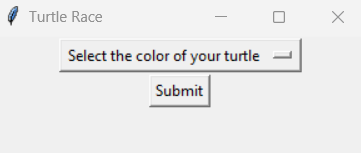
**Output and Screenshots**

The output generated by this Python project is shown as follows:

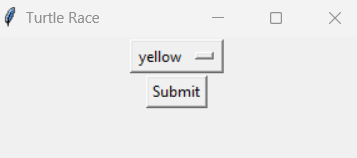
Sample-Output-:

1. Tkinter Window:

In the following window, the user is asked to make their bet. They select a turtle of their choice and submit.

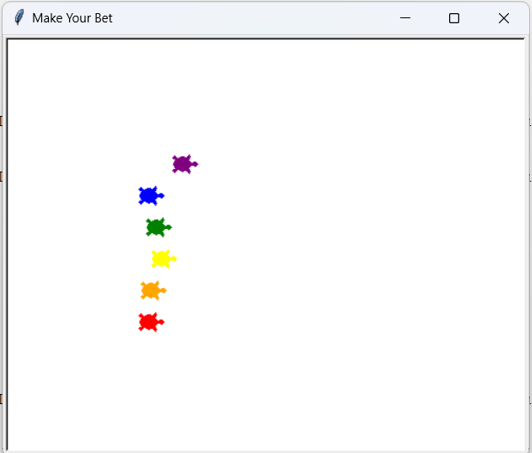


Once the user has chosen their turtle’s color, the color is stored in the variable - chosen\_color (Refer Source Code), as shown in following image:



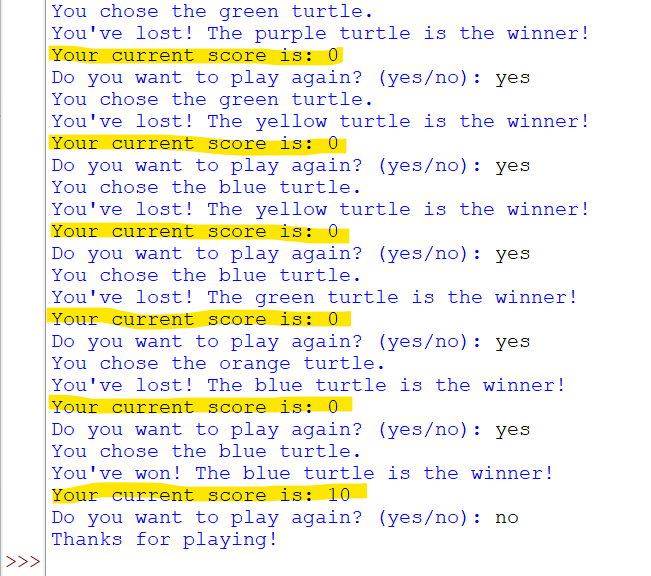
1. Turtle Race Screen:

Once the user has submitted their turtle’s color, the race starts as shown in the below image:



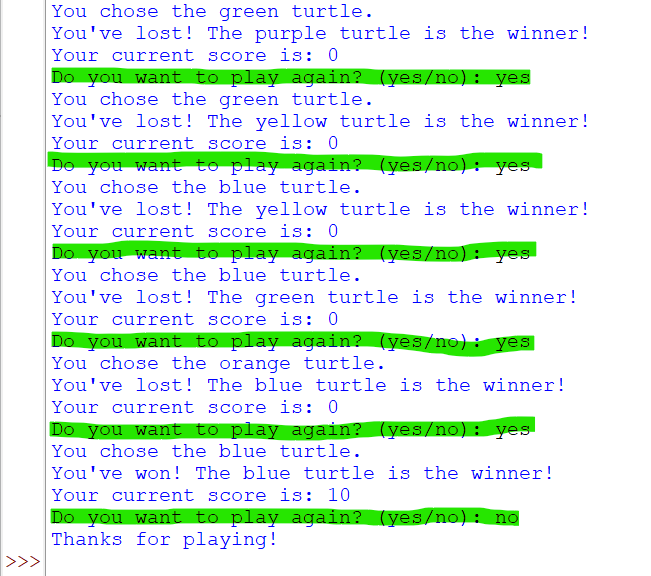
1. The points scored by the user:

If the user win’s his/her bet, they get 10 points, on the other hand, they get 0 if they lose (the previous scores of the user don’t get modified). Following image shows this:



1. Next Round of the Game:

Once the game ends the program displays the points of the user and also asks if they want to play another round of the game. If the user selects to play another round, then the old screen of the race is cleared, and a fresh screen is shown again, where the race starts. The old scores of the user remains unchanged, while the scores of the new round are added to the old ones and displayed accordingly.



On the other hand, when the user wishes to exit the game, the program comes out of the loop and exits the tkinter window and displays the message: “Thanks for playing!”.

**Conclusion**

This project offered a fantastic learning opportunity, allowing me to delve into the realms of Turtle graphics and GUI development using Tkinter. It provided hands-on experience in handling user input, creating visual elements, and managing game logic. Exploring this project bolstered my understanding of Python's graphical capabilities and honed my skills in building interactive applications. Witnessing the race come to life through code was an exciting journey that expanded my knowledge of programming paradigms and their practical applications.

**References**

Following were the references of this Python Project:

1. Udemy.com
2. https://en.wikipedia.org/wiki/Turtle\_graphics
3. Google.com