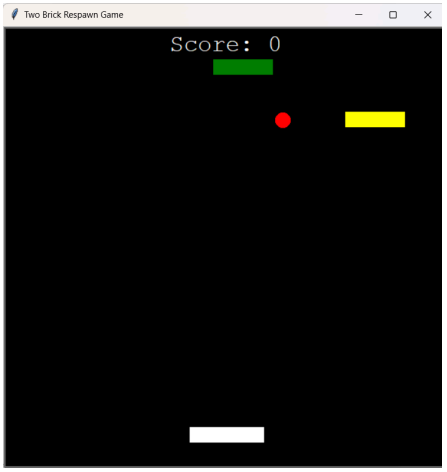
INDEX

Overview of the projects in Volume 1

In this volume, we will learn Python's turtle module along with core Python concepts while creating a **snake game**. The **turtle module** in Python is a simple graphics library used for drawing and animations.

The **Snake Game** uses the turtle module to create a moving snake. The snake moves in four directions and grows when it eats food, increasing the score. The game ends if the snake collides with the wall or itself, resetting the score but keeping the high score.

At the end of each chapter, we have a challenge activity to help you create another game using turtle - **Paddle Ball Game** - based on the concepts learned in that chapter. This will reinforce your learning while creating another interactive match all by yourself.



Tools used

- MU editor to be installed <link>

Libraries used

- turtle
- random
- time

Chapter 1: Objects and variables in Python turtle

1.1 What is an Object?

An **object** is a thing that has attributes (features) and functions (actions it can perform). Example: a real-life object like a *chair*.

A *chair* is an object because it has:

- **Attributes (Features)** → Color, material, number of legs, height
- **Functions (actions)** → Someone can sit on it, move it, fold it

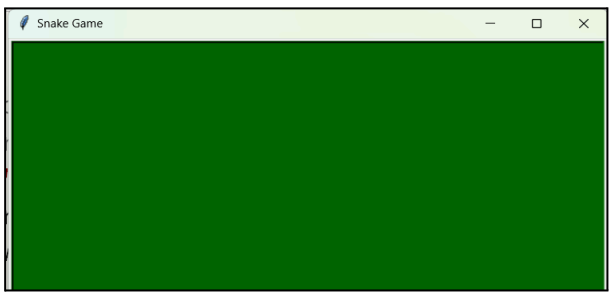
What is an Object in a Turtle?

In Python's **turtle module**, an **object** is like a **real turtle** that you can control on the screen and set attributes like color, coordinates, shape, size, etc.

1.1 Create turtle screen with object concept

```
1   import turtle  # Import the turtle module, which provides graphics functions for drawing.
2
3   # turtle.Screen() creates and returns a screen object, which is used to control the window.
4   turtle.Screen().title("Snake Game")  # Sets the title of the window to "Snake Game".
5
6   # turtle.Screen().bgcolor("Dark Green") sets the background color of the screen to dark green.
7   turtle.Screen().bgcolor("Dark Green")
8
9   # turtle.Screen().setup(width=600, height=600) sets the dimensions of the window to 600x600 pixels.
10  turtle.Screen().setup(width=600, height=600)
```

Output



Here, we have created a screen with dimensions 600 X 600 pixels and have given it a dark green background. If we need to use the same screen when we restart the game, we need to store this screen in some container to reuse it. This container in Python is called a variable.

1.2 What is a variable?

A **variable** is like a **box** that holds information. This box has a **name**, and you can put different things inside it, like numbers, words, or even lists of things.

Why do we use variables?

We use variables so we can **store** information and **use it later** in our programs.

Example of Variables in Real Life:

- A School Notebook
  - You write your notes in a notebook.
  - The notebook stores your notes so you can read them later.
  - The notebook is like a variable!

A **python variable**

- Must not have a space in between
- Must not start with a number
- Can have only _ as the special character

  Example: age, number1, full_name

1.2 Create a variable to store the screen

```
1   import turtle  # Import the turtle module, which provides graphics functions for drawing.
2
3   # Create a screen object using the turtle.Screen() function and assign it to variable 't'.
4   t = turtle.Screen()
5
6   # Set the title of the game window to "Snake Game".
7   t.title("Snake Game")
8
9   # Set the background color of the window to "Dark Green".
10  t.bgcolor("Dark Green")
11
12  # Set the size of the window to 600 pixels wide and 600 pixels high.
13  t.setup(width=600, height=600)
```

Here, on line 4, *t* is the variable that stores the turtle **Screen** object.

We can use this object to call the turtle screen functions. Observe the change on lines 7 to 13 where turtle.Screen() is replaced by the object "*t*".

1.3 Create the snake head and food object

A turtle object will **listen to your commands** and move, turn, and draw as you tell it.

To create a turtle object, we use a turtle.Turtle().

1.3 Create a variable for snake head

```
1     import turtle
2     t = turtle.Screen()
3     t.title("Snake Game")
4     t.bgcolor("Dark Green")
5     t.setup(width=600, height=600)
6     # Create a turtle object named 'head' that will represent the snake's head.
7     head = turtle.Turtle()
8     # Set the shape of the 'head' turtle to a square.
9     head.shape("square")
10    # Set the color of the 'head' turtle to black.
11    head.color("black")
```

Lab Activity 1

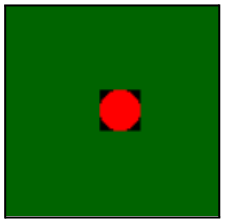
➢ Create a **food object** for the snake game. The food should:

✅ Be a **new turtle object** with the name of food.

✅ Have a **circle shape**

✅ Be **red in colour**

Output



1.4 Position the snake head using coordinates

What is coordinate?

Consider the screen a sheet where the food and the head appear in the center. Check this Link to understand the coordinates. The center of the coordinate is (0,0). This is the **starting point** for your turtle where x = 0 and y = 0.

We use **coordinates** (X, Y) to tell the turtle where to go on the screen!

- **X coordinate** → Moves the turtle **left (-) or right (+)**
- **Y coordinate** → Moves the turtle **up (+) or down (-)**

How does the turtle move?

The `goto(x, y)` function in the turtle module moves the turtle directly to a specific position on the screen using coordinates (X, Y). Example:

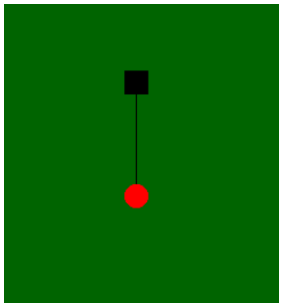
- `turtle.goto(100,50)` moves the turtle right to X = 100 and up to Y = 50
- `turtle.goto(-50,-100)` moves the turtle left to X = -50 and down to Y = -100

1.4 Change head coordinate

```
11      # Snake Head
12      head = turtle.Turtle()
13      head.shape("square")
14      head.color("black")
15
16      #move the head turtle 100 pixels up along the Y coordinate
17      head.goto(0,100)
```

Here, we have moved the head turtle position to (0,100) using the goto() function.

Output



Lab Activity 2

- ➢ Move the **food object** in the snake game
  - ✅ move it horizontally left
  - ✅ move it vertically down.

There is a line that is drawn when the head or food turtle moves to a new position.

We need to avoid drawing this line when the head/food turtle moves.

1.5 Other turtle functions - penup(), pendown(), speed(), shapesize()

We can use the following functions of the turtle to solve the above problem of not drawing while moving the turtle to a different position.

penup( ) 🖊 **(Pick up the pen)** – The turtle moves **without drawing**.

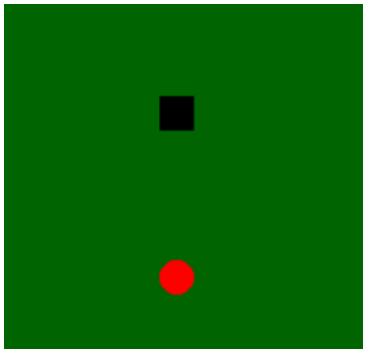
pendown( ) 🖊 **(Put the pen down)** – The turtle **starts drawing** again.

1.5 Move the head without drawing a line

```
12      # Snake Head
13      head = turtle.Turtle()
14      head.shape("square")
15      head.color("black")
16
17      #penup() - helps head turtle to move to (0,100) without drawing a line
18      head.penup()
19      head.goto(0,100)
```

In the above code, in line 18, we have added penup() before the snake head moves to a new position without drawing a line.

Output



We can control the speed at which the turtle moves using the speed() function.

Speed ranges from **0 (fastest), 1 (slowest) to 10 (very fast)**. Try to change the speed of the turtle and observe the change.

1.6 Add the speed of the turtle

```
11      # Snake Head
12      head = turtle.Turtle()
13      head.shape("square")
14      head.color("black")
15
16      #speed() - to control the speed of the turtle movement
17      #0-(fastest), 1-(slowest) to 10-(very fast)
18      head.speed(0)
19
20      head.penup()
21      head.goto(0,100)
```

On line 18, we have added the speed for the head. This moves the head faster to the new position.

Lab Activity 3

➢ Change the following for the food object:

✅ move the food with penup()

✅ change the speed of the food object.

✅ make the food size smaller using shapesize(0.75, 0.75).

Note: Syntax: shapesize(height, width). By default, the shapesize is (1,1).

Python Concepts learned in this chapter:

- Variables in Python
- Objects and their properties
- Turtle functions like - goto(), penup(), shapesize()

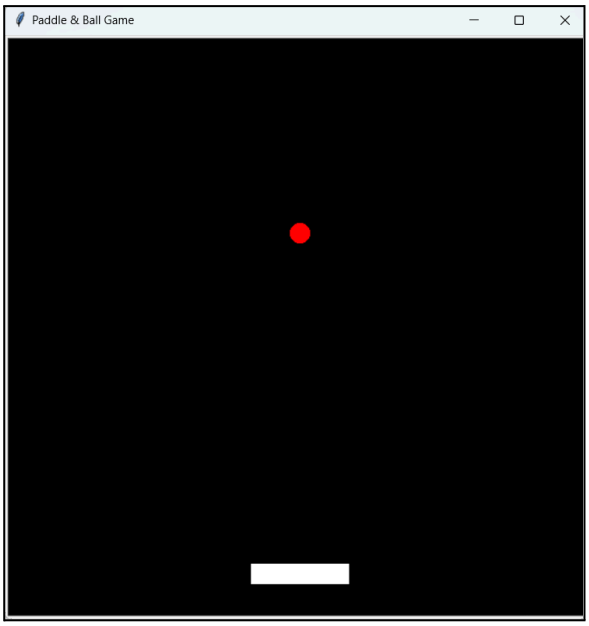
CHALLENGE 1:

We will create a new game - "**Paddle-Ball Game**", similar to the above game.

Let's go ahead and create the objects required for the game:

- The screen must have
  - Black background
  - Title as "Paddle and Ball Game"
  - Width and height as 600
- Create a paddle object:
  - The shape is a square
  - colour is white
  - Position it at X = 0, Y = -250
  - Use shapesize(1,5) to stretch the length-wise to make it appear as a rectangle.
- Create a ball object:
  - The shape is a circle
  - Colour is red
  - Position it at X = 0, Y = 100)

Output:



Chapter 2: Event Handling and Functions to Move the Snake

2.1 Why event handling?

Now, our snake has to move towards the food with the help of arrow keys. The game will have to listen and detect which arrow keys are being pressed. This can be done with the help of listen() and onkey() functions.

turtle.listen()

It makes sure the program is ready to detect key presses.

turtle.onkey()

- t.onkey(function_name, "Key") **binds a key press to a function**. Function_name is the name of the function that is called when a particular "Key" is pressed.

2.1 Event handling with keys

```
27      #turtle listens for key press
28      t.listen()
29
30      #when up arrow key is pressed, it calls move_up() function
31      t.onkey(move_up, "Up")       # Press ↑ to move up
32
33      #when left arrow key is pressed, it calls move_left() function
34      t.onkey(move_left, "Left")   # Press ← to move left
```

Here, onkey() on line 31 and 34 calls the move_up function when the UP arrow is pressed and the move_left function when the LEFT arrow is pressed. These functions are discussed below in the next section.

Lab Activity 4

➢ Write the onkey functions for these keys:

✅ Right

✅ Down

2.2 What is a function?

A function is like a block of code that tells the turtle what to do. This block of code runs only when the function is called. To create a function, we use the following syntax:

```
def function_name():
    # block of code within the function
```

✅ `def` is used to **define** a function. Observe the brackets and colon after the function name.

✅ **Observe the indentation of the code within the function.**

For the 4 arrow keys, we need 4 different functions that can move the head in four different directions.

2.2 Add Functions for Event Handling

The functions can be defined as follows on line 28 and 32 for the up and left arrow keys

```
27      #function for up arrow key pressed
28      def move_up():
29          # move the head up by 20 pixels
30
31      #function for left arrow key pressed
32      def move_up():
33          # move the head left by 20 pixels
```

As there's nothing defined in the function body, no output will be seen when the arrow keys are clicked.

2.3 How do we move the head?

We need to know the current position and then change this by a few pixels to move the head.

To know the current position, we can use the xcor() and ycor() functions of the turtle.

`turtle.xcor()` → Returns the **current X-coordinate** of the turtle.

`turtle.ycor()` → Returns the **current Y-coordinate** of the turtle.

As discussed in 1.4 about the coordinates (refer to Link), a turtle's position is set by its (x,y) coordinates. We need to do the following to move the snake head

Right - increase the x coordinate by few pixels (xcor() + pixel)

Left - decrease the x coordinate by few pixels (xcor() - pixel)

Up - increase the y coordinate by few pixels (ycor() + pixel)

Down - decrease the y coordinate by few pixels (ycor() - pixel)

To set the new position of the turtle, we can use the setx() and sety() functions giving it the new x or the new y coordinate.

`setx(x)` → Sets the turtle to the given **X-coordinate(x)** while keeping the Y-coordinate the same.

`sety(y)` → Sets the turtle to the given **Y-coordinate(y)** while keeping the X-coordinate the same.

2.3 Set the new coordinate for the head

Now, the code for move_up() can be written as:

```
28  def move_up():
29      # get the current y coordinate of the head
30      cur_y = head.ycor()
31
32      #change the y coordinate of head
33      #to move the head up by 20 pixels
34      new_y = cur_y + 20
35
36      #set the head to new y coordinate
37      head.sety(new_y)
```

To move the snake head up,

Line 30 - Get the y coordinate.

Line 34 - Increase this value by 20 pixels to move the snake head up.

Line 37 - Set the y coordinate to the new value.

Lab Activity 5

➢ Write the function code for:

✅ move_left

✅ move_right

✅ move_down

Note: Follow the function code written for move_up(). Apply similar operations on x or y coordinates based on the directions accordingly.

Python Concepts learned in this chapter

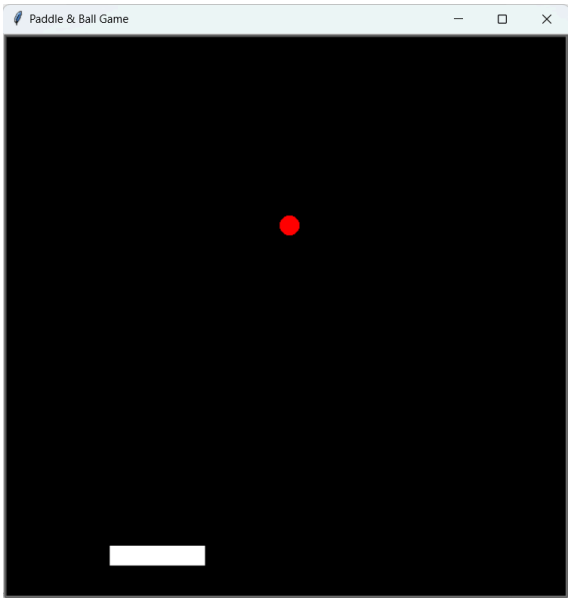
- Event handling in Turtle - listen() and onkey() functions
- How to define Functions in Python using the *def* keyword
- How to call Functions in Python by the name of the function.
- How to get and set the coordinates of the python turtle object.

CHALLENGE 2

We will continue with our new game - "**Paddle-Ball Game**".

- Write code to move the paddle only in left and right directions when the left and right arrow keys are pressed.

Output



Note: This picture shows the paddle has moved to the left when the left arrow is pressed.

Summer Project Task for Students

Create an **interactive drawing tool** where the user can control a turtle's movement using keyboard inputs.

Requirements

1. The program should open a **600x600 pixel window** with a white background and a title: *"Turtle Artist - Draw with WASD keys!"*.
2. A turtle with a **blue colour, turtle shape, and a pen size of 3** should be displayed at the start.
3. The turtle should move **10 pixels** per step when pressing the forward (**W**) or backward(**S**) keys.
4. The turtle should turn **90 degrees** when rotating left (**A**) or right (**D**).
5. The pen should toggle between **up and down** mode when the **spacebar** is pressed.
6. The program should continuously listen for key presses to control the turtle's movement dynamically.

Output

Sample output using WASD keys to move the turtle and draw something on the screen.