MODULE 3

JUNE 2023

QN:1 Illustrate the function of following methods in turtle

1. turtle.setheading(0) ii)turtle.forward(50) iii) turtle.left(90)

ANS:

**1. turtle.setheading(0)**

* This sets the turtle’s orientation to **0 degrees** (which points **east**).
* Example: If the turtle was facing **north (90°)** or **west (180°)**, it will now face **east (0°)**.

**2. turtle.forward(50)**

* Moves the turtle **forward** by **50 units** in the direction it is currently facing.
* Example: If the turtle is facing **east (0°)**, it moves **right** by 50 units.

**3. turtle.left(90)**

* Rotates the turtle **90 degrees** to the **left (counterclockwise)**.
* Example: If the turtle was facing **east (0°)**, after turning left by **90°**, it will face **north (90°)**.

QN:2

|  |
| --- |
| Describe two fundamental differences between terminal-based user interfaces  and GUIs. |

ANS:

**Interaction Method**

* **TUI:** Operates using **text commands** typed into a terminal or command-line interface (CLI). Users must remember and input specific commands.
* **GUI:** Uses **graphical elements** like buttons, menus, and icons for interaction, making it more visually intuitive and user-friendly.

**Resource Usage**

* **TUI:** **Lightweight** and consumes minimal system resources (RAM, CPU, and storage), making it ideal for low-power or remote systems.
* **GUI:** **Resource-intensive**, requiring more memory and processing power due to graphical rendering and event handling.

JANUARY 2024

QN: Explain the attributes and methods of Turtle object.

ANS:

**1. Attributes of Turtle Object**

Attributes define the state of the turtle.

| **Attribute** | **Description** |
| --- | --- |
| position | The current (x, y) coordinates of the turtle. |
| heading | The current direction in degrees (0° = east, 90° = north). |
| pensize | The width of the pen (default is 1). |
| pencolor | The color of the pen (e.g., "red", "blue"). |
| fillcolor | The color used to fill shapes. |
| speed | The turtle’s movement speed (0 = fastest, 1-10 = slow to fast). |
| isdown | True if the pen is down (drawing), False if it is up. |
| visible | True if the turtle is visible, False if hidden. |
| shape | The appearance of the turtle ("arrow", "turtle", "circle", etc.). |

**2. Methods of Turtle Object**

Methods define what actions the turtle can perform.

**A. Movement Methods**

| **Method** | **Description** |
| --- | --- |
| forward(distance) or fd(distance) | Moves the turtle forward by distance units. |
| backward(distance) or bk(distance) | Moves the turtle backward by distance units. |
| right(angle) or rt(angle) | Rotates the turtle **clockwise** by angle degrees. |
| left(angle) or lt(angle) | Rotates the turtle **counterclockwise** by angle degrees. |
| goto(x, y) | Moves the turtle to the given (x, y) coordinates. |
| setx(x) | Moves the turtle to the given x position. |
| sety(y) | Moves the turtle to the given y position. |
| setheading(angle) | Sets the turtle’s orientation (0° = east, 90° = north). |
| home() | Moves the turtle to (0,0) and resets heading to 0°. |

**B. Pen Control Methods**

| **Method** | **Description** |
| --- | --- |
| penup() or pu() | Lifts the pen (turtle moves without drawing). |
| pendown() or pd() | Lowers the pen (turtle moves and draws). |
| pensize(width) | Sets the width of the pen. |
| pencolor(color) | Changes the pen color. |
| fillcolor(color) | Changes the fill color of shapes. |
| begin\_fill() | Starts filling a shape. |
| end\_fill() | Stops filling and fills the shape. |

**C. Appearance Methods**

| **Method** | **Description** |
| --- | --- |
| shape("shape\_name") | Changes the turtle’s shape ("turtle", "circle", "square", etc.). |
| shapesize(stretch\_wid, stretch\_len) | Stretches the turtle shape. |
| speed(value) | Sets the movement speed (0 = fastest, 1-10 = slow to fast). |
| hideturtle() or ht() | Hides the turtle. |
| showturtle() or st() | Shows the turtle. |

**D. State and Position Methods**

| **Method** | **Description** |
| --- | --- |
| position() | Returns the current (x, y) coordinates. |
| heading() | Returns the current heading (angle in degrees). |
| isdown() | Returns True if the pen is down, else False. |

**E. Reset and Clear Methods**

| **Method** | **Description** |
| --- | --- |
| clear() | Clears the drawing but keeps the turtle’s position. |
| reset() | Clears the screen and resets the turtle to (0,0). |

QN: Discuss on the types of window components and their functions.

ANS:

In **Python**, especially when using GUI frameworks like **Tkinter**, **PyQt**, or **Kivy**, different **window components (widgets)** help create interactive applications. Here are some common window components and their functions:

**1️-Main Window (Tk or QMainWindow)**

* **Function:** The primary application window where all widgets are placed.
* **Example (Tkinter):**

>>>import tkinter as tk

>>>root = tk.Tk() # Main window

>>>root.title("Main Window")

>>>root.mainloop()

**2️-Labels (Label / QLabel)**

* **Function:** Displays text or images in the window.
* **Example (Tkinter):**

>>>label = tk.Label(root, text="Hello, World!")

>>>label.pack()

**3️-Buttons (Button / QPushButton)**

* **Function:** Used to trigger actions when clicked.
* **Example (Tkinter):**

>>>def on\_click():

>>>…print("Button clicked!")

>>>…button = tk.Button(root, text="Click Me", command=on\_click)

>>>…button.pack()

**4️-Text Entry (Entry / QLineEdit)**

* **Function:** Allows users to input text.
* **Example (Tkinter):**

>>>entry = tk.Entry(root)

>>>entry.pack()

**5️-Box (Text / QTextEdit)**

* **Function:** Accepts multi-line text input.
* **Example (Tkinter):**

>>>text\_box = tk.Text(root, height=5, width=30)

>>>text\_box.pack()

**6️-Frames (Frame / QFrame)**

* **Function:** Acts as a container to organize widgets.
* **Example (Tkinter):**

>>>frame = tk.Frame(root, bg="lightgray", width=200, height=100)

>>>frame.pack()

**7️-Menus (Menu / QMenuBar)**

* **Function:** Provides a dropdown menu for navigation.
* **Example (Tkinter):**

>>>menu = tk.Menu(root)

>>>root.config(menu=menu)

>>>file\_menu = tk.Menu(menu)

>>>menu.add\_cascade(label="File", menu=file\_menu)

>>>file\_menu.add\_command(label="Exit", command=root.quit)

**8️-Checkboxes (Checkbutton / QCheckBox)**

* **Function:** Allows users to select multiple options.
* **Example (Tkinter):**

>>>chk = tk.Checkbutton(root, text="Check Me")

>>>chk.pack()

**9️-Radio Buttons (Radiobutton / QRadioButton)**

* **Function:** Allows users to select **one** option from multiple choices.
* **Example (Tkinter):**

>>>tk.Radiobutton(root, text="Option 1", value=1).pack()

>>>tk.Radiobutton(root, text="Option 2", value=2).pack()

2024 MAY

QN: How can you set the drawing speed of the turtle in the Turtle module? Give an

Example

ANS:

The **speed(value)** method in the turtle module is used to control the turtle’s drawing speed.

**Speed Values and Their Meaning**

| **Value** | **Speed Description** |
| --- | --- |
| 0 | Fastest (no animation) |
| 1 | Slowest |
| 3-5 | Medium speed |
| 6-10 | Fastest speeds (gradually increasing) |

PROGRAM:

import turtle

t = turtle.Turtle()

t.speed(5)

t.forward(100)

t.left(90)

t.forward(100)

t.hideturtle()

turtle.done()

QN: List the steps to create a GUI application using Tkinter

ANS:

->Import Tkinter  
->Create the Main Window (Root Window)  
->Set the Window Size (Optional)  
->Add Widgets (Labels, Buttons, Entry, etc.)  
->Use Layout Managers (pack(), grid(), place())  
->Implement Event Handling (Functions & Callbacks)  
->Run the Tkinter Event Loop using mainloop()

MAY 2023

QN: What are the attributes of a turtle object?

ANS:

**Position & Orientation Attributes**

| **Attribute** | **Description** |
| --- | --- |
| position | Current (x, y) coordinates of the turtle. |
| xcor() | Returns the current x-coordinate. |
| ycor() | Returns the current y-coordinate. |
| heading() | Returns the current direction in degrees (0° = east, 90° = north). |

**Pen Attributes**

| **Attribute** | **Description** |
| --- | --- |
| pensize | Width of the drawing pen (default is 1). |
| pencolor | Color of the drawing pen (e.g., "red", "blue"). |
| fillcolor | Color used to fill shapes. |
| isdown() | Returns True if the pen is down (drawing), otherwise False. |

| **Appearance Attributes**  **Attribute** | **Description** |
| --- | --- |
| shape | The shape of the turtle ("arrow", "turtle", "circle", etc.). |
| shapesize() | Returns or sets the size of the turtle shape. |
| visible | True if the turtle is visible, False if hidden. |

**Speed & Movement Attributes**

| **Attribute** | **Description** |
| --- | --- |
| speed | Speed of the turtle’s movement (0 = fastest, 1-10 = slow to fast). |

QN: What are the advantages of GUI based programs over terminal based programs.

ANS:

ADVANTAGES:-

* + **User-Friendly** – Easier to use with visual elements like buttons, icons, and menus.
  + **No Need for Commands** – Users don’t need to remember text-based commands.
  + **Better Visualization** – Supports images, graphs, and interactive elements.
  + **Multi-Tasking** – Allows multiple windows and applications to run simultaneously.
  + **Intuitive Navigation** – Uses drag-and-drop, click, and touch-based interactions.
  + **Accessibility** – Easier for non-technical users compared to command-line interfaces.
  + **Error Prevention** – GUI restricts invalid inputs with dropdowns, forms, and tooltips.
  + **Attractive Design** – Provides better aesthetics with customizable themes and layouts.
  + **Increased Productivity** – Reduces learning time and improves efficiency.
  + **Wider Adoption** – Used in almost all modern applications, making them more accessible.

JUNE 2022

QN: How do you display an image in Python GUI?.

ANS:

**Using Tkinter (PhotoImage)**

* Suitable for displaying PNG, GIF, and PPM images in a Tkinter window.
* **Steps:**
  + Import tkinter and PhotoImage
  + Load the image
  + Display it using a Label widget

QN: List any three image processing Python libraries.

ANS:

1️-**OpenCV (cv2)** – Used for real-time image processing, computer vision, and deep learning applications.

2️-**Pillow (PIL)** – A powerful library for image manipulation, including resizing, filtering, and format conversion.

3- **scikit-image** – A scientific library for image processing, offering advanced algorithms for feature extraction, segmentation, and transformation.