APPENDIX C

Simple Graphics Using Turtle

C.1 INTRODUCTION

Turtle is a module in Python which is like a robot and allows you to easily draw intricate shapes and pictures all over the screen. With the methods defined in the turtle module, you can set the turtle's position and heading, control its forward and backward movement, specify the type of pen it is holding, etc., as given in Table C.1. However, before using the turtle module, you should first import it.

Table C.1 Methods in `turtle` Module

Method	Purpose
`turtle.forward()`	Moves the turtle forward by specified steps.
`turtle.backward()`	Moves the turtle backward by specified steps.
`turtle.left()`	Takes a number of degrees which you want to rotate to the left.
`turtle.right()`	Takes a number of degrees which you want to rotate to the right.
`turtle.reset()`	Clears the drawing.
`turtle.shape()`	Change the shape of the turtle.
`turtle.exitonclick()`	Closes the window when you click on the turtle.
`turtle.undo()`	To undo the last action performed on the turtle.
`turtle.color()`	To set the color of the turtle.
`turtle.hideturtle()`	To hide the turtle from the screen.
`turtle.showturtle()`	To show the (hidden) turtle on the screen.
`turtle.setposition()` or `turtle.setpos()`	Sets the position of turtle in the graphics window.
`turtle.pendown()`	Draws a line while moving the turtle.
`turtle.penup()`	Does not draw the line while moving the turtle.
`turtle.clear()`	Clears the drawing from the graphics window leaving the turtle in its current position.
`turtle.reset()`	Clears the drawing from the graphics window and returns the turtle to its starting position (center of the screen).
`turtle.circle()`	Draws a circle. It has radius as the mandatory argument. It also has optional arguments to specify the degrees of arc that are drawn, and the `"steps,"` which are the number of straight-line segments used to approximate the circle. If the radius is positive, the circle is drawn by turning to the left (counterclockwise), otherwise a negative radius draws the circle from the current position by turning to the right (clockwise).

Contd

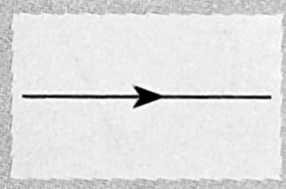
Table C.1 *Contd*

Method	Purpose
`turtle.speed()`	Controls the speed of drawing graphics. It takes speed as an integer argument (0 – 10). Default speed is 3, minimum speed is 1, and maximum is 10.
`turtle.tracer()`	Makes the animation faster. It takes two arguments—one controls how often screens should be updated and the other controls the delay between these updates. For fastest possible rendering, both these arguments should be set to zero. To reset `tracer()` to its original settings, its arguments should be 1 and 10.
`turtle.pensize()`	To change the thickness of the turtle's pen. It takes an integer argument that specifies the thickness of the pen.
`turtle.bgcolor()`	Changes the background color of the graphics window.
`turtle.fill()`	To fill a closed shape with the currently set color.

Example C.1 Program to move the turtle forward and then backward after a delay of 2 seconds

```
import turtle
import time
turtle.forward(50)
time.sleep(2)
turtle.backward(20)
```

OUTPUT



Example C.2 Program to change the shape of the turtle, turn it left, and then move forward

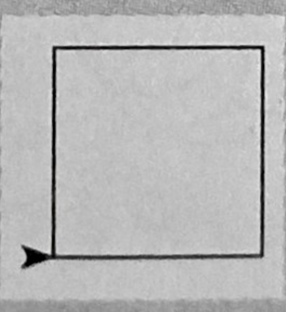
```
import turtle
import time
turtle.shape("square")
turtle.left(45)
turtle.forward(50)
turtle.exitonclick()
```

OUTPUT

Example C.3 Program to draw a square

```
import turtle
import time
turtle.forward(100)
turtle.left(90)
time.sleep(1)
turtle.forward(100)
turtle.left(90)
time.sleep(1)
turtle.forward(100)
turtle.left(90)
time.sleep(1)
turtle.forward(100)
turtle.left(90)
```

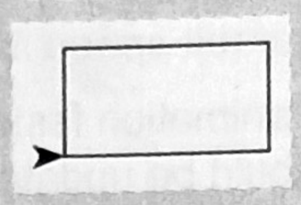
OUTPUT



Example C.4 Program to draw a red color rectangle

```
import turtle
import time
turtle.color("red")
turtle.forward(100)
turtle.left(90)
turtle.forward(50)
turtle.left(90)
turtle.forward(100)
turtle.left(90)
turtle.forward(50)
turtle.left(90)
```

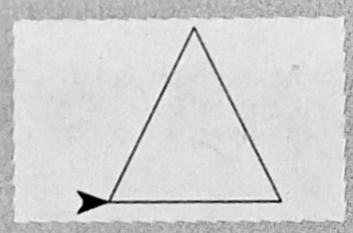
OUTPUT



Example C.5 Program to draw a green color equilateral triangle

```
import turtle
import time
turtle.color("green")
turtle.forward(70)
turtle.left(120)
turtle.forward(70)
turtle.left(120)
turtle.forward(70)
```

OUTPUT

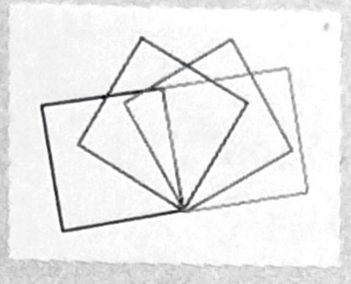


Example C.6 Program to draw a pattern of different color squares with different angles

```
import turtle
turtle.color("red")
turtle.left(10)
turtle.forward(50)
turtle.left(90)
turtle.forward(50)
turtle.left(90)
turtle.forward(50)
turtle.left(90)
turtle.forward(50)
turtle.left(90)

turtle.color("green")
turtle.left(20)
turtle.forward(50)
turtle.left(90)
turtle.forward(50)
turtle.left(90)
turtle.forward(50)
turtle.left(90)

turtle.color("blue")
turtle.left(30)
turtle.forward(50)
turtle.left(90)
turtle.forward(50)
turtle.left(90)
turtle.forward(50)
turtle.left(90)
turtle.forward(50)
turtle.left(90)
```

```
turtle.color("orange")
turtle.left(40)

turtle.forward(50)
turtle.left(90)
turtle.forward(50)
turtle.left(90)
turtle.forward(50)
turtle.left(90)
turtle.forward(50)
turtle.left(90)
```

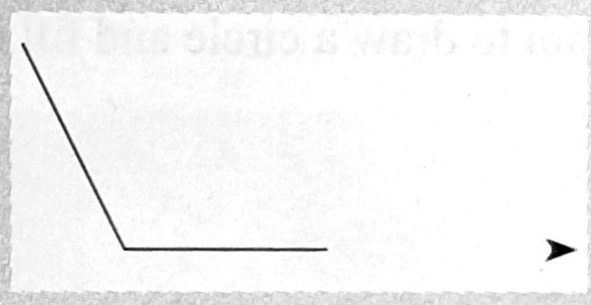
OUTPUT



Example C.7 Program to demonstrate the use of `setposition()`, `pendown()`, and `penup()` methods

```
import turtle as t
t.setposition(50, -70)
t.forward(50)
t.pendown()
t.forward(50)
t.penup()
t.forward(150)
```

OUTPUT



Example C.8 Program to draw a red color circle

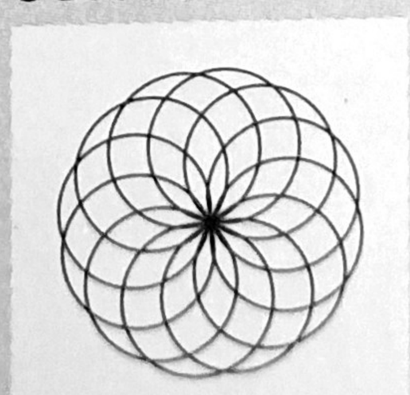
```
import turtle
turtle.color("red")
turtle.circle(50)
```

OUTPUT

Example C.9 Program to draw a red color thick pen on a yellow background

```
import turtle
turtle.bgcolor("yellow")
turtle.color("red")
turtle.pensize(10)
for angle in range(0, 360, 30):
    turtle.seth(angle)
    turtle.circle(100)
```

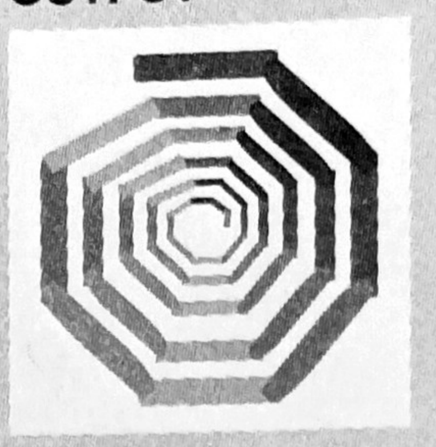
OUTPUT



Example C.10 Program to draw a pattern of straight lines

```
import turtle
colors = ["red", "blue", "yellow",
"green", "purple", "orange"]
#turtle.reset()
turtle.tracer(0, 0)
for i in range(45):
    turtle.color(colors[i % 6])
    turtle.pendown()
    turtle.forward(2 + i * 5)
    turtle.left(45)
    turtle.width(i)
    turtle.penup()
    turtle.update()
```

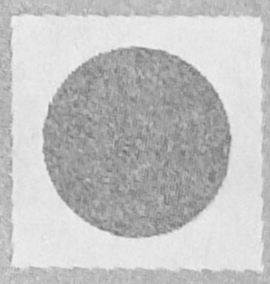
OUTPUT



Example C.11 Program to draw a circle and fill it with orange color

```
import turtle as t
t.color("orange")
t.begin_fill()
t.circle(50)
t.end_fill()
t.hideturtle()
```

OUTPUT



Example C.12 Program to draw a random pattern of circle, square, and rectangle

```
import turtle as t
def square(x):
    for i in range(4):
        t.forward(x)
        t.left(90)
def rectangle(x,y):
    t.forward(x)
    t.left(90)
    t.forward(y)
    t.left(90)
    t.forward(x)
    t.left(90)
    t.forward(y)

def draw_shape(x, y):
```

```
t.penup()
t.setpos(x, y)
t.pendown()
t.begin_fill()
if x <= 50: # Left third.
    t.color("red")
    square(10)
elif 50 < x <= 150: # Middle third.
    t.color("yellow")
    t.circle(10)
else: # Right third.
    t.color("purple")
    rectangle(10,20)
t.onscreenclick(draw_shape)
t.mainloop()
```

OUTPUT