Turtle is a special feathers of Python. Using Turtle, we can easily draw in a drawing board.

First we import the turtle module. Then create a window, next we create turtle object and using turtle method we can draw in the drawing board.

Some turtle method

|  |  |  |
| --- | --- | --- |
| **METHOD** | **PARAMETER** | **DESCRIPTION** |
| Turtle() | None | It creates and returns a new turtle object |
| forward() | amount | It moves the turtle forward by the specified amount |
| backward() | amount | It moves the turtle backward by the specified amount |
| right() | angle | It turns the turtle clockwise |
| left() | angle | It turns the turtle counter clockwise |
| penup() | None | It picks up the turtle’s Pen |
| pendown() | None | Puts down the turtle’s Pen |
| up() | None | Picks up the turtle’s Pen |
| down() | None | Puts down the turtle’s Pen |
| color() | Color name | Changes the color of the turtle’s pen |
| fillcolor() | Color name | Changes the color of the turtle will use to fill a polygon |
| heading() | None | It returns the current heading |
| position() | None | It returns the current position |
| goto() | x, y | It moves the turtle to position x,y |
| begin\_fill() | None | Remember the starting point for a filled polygon |
| end\_fill() | None | It closes the polygon and fills with the current fill color |
| dot() | None | Leaves the dot at the current position |
| stamp() | None | Leaves an impression of a turtle shape at the current location |
| shape() | shapename | Should be ‘arrow’, ‘classic’, ‘turtle’ or ‘circle’ |

Example code

# import turtle library

import turtle

my\_window = turtle.Screen()

my\_window.bgcolor("blue") # creates a graphics window

my\_pen = turtle.Turtle()

my\_pen.forward(150)

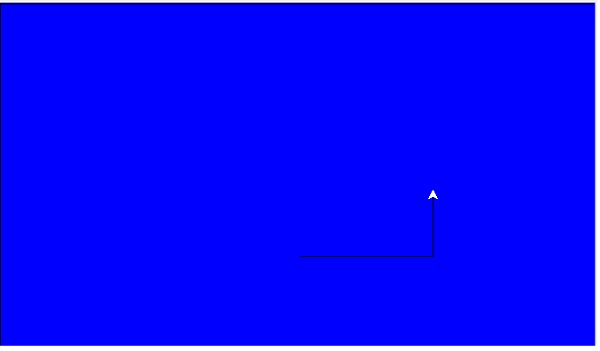
my\_pen.left(90)

my\_pen.forward(75)

my\_pen.color("white")

my\_pen.pensize(12)

Output



## Draw a star

## Example code

# import turtle library

import turtle

my\_pen = turtle.Turtle()

for i in range(50):

my\_pen.forward(50)

my\_pen.right(144)

turtle.done()

## Draw a Hexagon

## Example code

# import turtle library

import turtle

polygon = turtle.Turtle()

my\_num\_sides = 6

my\_side\_length = 70

my\_angle = 360.0 / my\_num\_sides

for i in range(my\_num\_sides):

polygon.forward(my\_side\_length)

polygon.right(my\_angle)

turtle.done()

## Draw a square inside another square box.

## Example code

# import turtle library

import turtle

my\_wn = turtle.Screen()

my\_wn.bgcolor("light blue")

my\_wn.title("Turtle")

my\_pen = turtle.Turtle()

my\_pen.color("black")

def my\_sqrfunc(size):

for i in range(4):

my\_pen.fd(size)

my\_pen.left(90)

size = size - 5

my\_sqrfunc(146)

my\_sqrfunc(126)

my\_sqrfunc(106)

my\_sqrfunc(86)

my\_sqrfunc(66)

my\_sqrfunc(46)

my\_sqrfunc(26)

## Drawing of another pattern

## Example code

# import turtle library

import turtle

my\_wn = turtle.Screen()

turtle.speed(2)

for i in range(30):

turtle.circle(5\*i)

turtle.circle(-5\*i)

turtle.left(i)

turtle.exitonclick()

# **Python Turtle Programming Tutorial**

Python Turtle is a good way to learn basic programming syntax of Python. You can use it to teach programming to kids. It is like a canvas or easel board that enables you to draw different figures by giving commands to turtle.

## Steps involved in drawing using Python Turtle

1. Prior to drawing anything using the turtle, you must start by importing the turtle module.
2. import turtle
3. or

from turtle import \*

1. Call turtle.Turtle() to create a new drawing board and assign it to an object t.

t = turtle.Turtle()

1. Start drawing by calling the different methods of turtle such as forward(), left(), right(). etc.
2. When you are done with the drawing, call turtle.done()

## Important methods that are used in turtle drawing

| **Methods** | **Description** |
| --- | --- |
| Turtle() | Creates a turtle object. |
| forward(distance) | This method moves the turtle forward by a specified distance. |
| backward(distance) | This method moves the turtle backward by a specified distance. |
| left(angle) | It is used to turn the turtle anti-clockwise by a specified angle. |
| right(angle) | It is used to turn the turtle clockwise by a specified angle. |
| goto(x, y) | It moves the turtle to the position (x,y) on the canvas. |
| dot(size) | This method draws a dot of a specified size. |
| circle(radius) | It draws a circle of a provided radius. |
| stamp() | Leaves a mark of turtle shape at the current position. |
| shape('shapename') | Changes the shape of a turtle to the specified shape. You can pass arrow, square, triangle, circle, and turtle. |
| setheading(angle) | It is used to set the orientation of the turtle to the provided angle. |
| penup() or up() | It lifts up the turtle's pen and no drawing after calling this method. |
| pendown() or down() | It puts the turtle's pen on the canvas and drawing can be done after calling this method. |
| color('colorname') or color('#rrggbb') | Sets the fill color and color of the turtle's pen. |
| pencolor('colorname') or pencolor('#rrggbb') | Sets a color of the turtle's pen. |
| fillcolor('colorname') or fillcolor('#rrggbb') | Sets the fill color of the turtle that will fill a polygon. |
| begin\_fill() | When you want to fill a shape with a color, then call this method. This method will remember the initial position of a filled polygon. |
| end\_fill() | This method fills the polygon with the current fill color by closing it between the current position and the initial position. |
| write('text') | Writes a text on the canvas. |

Using above methods, you can draw numerous shapes and figures. Even you can draw using different colors and fill shapes with your favorite color.

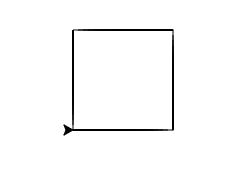
# **How to draw square and rectangle in Python Turtle?**

Two functions that are useful for us for drawing square and rectangle are- forward() and left(). Before we draw any of the shapes, we must know the basic properties of them. Let's start with a square. All the sides of a square are equal. And the angle between two adjacent sides is 90°. Opposite sides are parallel to each other.

Now, we know the basic features of the square. It's time to draw a square in Python Turtle. Let's assume the side of a square is 100 units.

|  |
| --- |
| #Program to draw square in Python Turtle  import turtle    t = turtle.Turtle()  t.forward(100) #Forward turtle by 100 units  t.left(90) #Turn turtle by 90 degree  t.forward(100)  t.left(90)  t.forward(100)  t.left(90)  t.forward(100)  t.left(90) |

Output of the above program



Explanation of the above code-

import turtle

t = turtle.Turtle()

Here, we are importing the turtle module. Then, we have created a new drawing board and assigned it to an object t.

forward(100)

left(90)

We have moved the turtle in the forward direction by 100 units. Because the side of a square is 100 units. Then, we have turned the turtle by 90° as the angle between adjacent sides is 90°. This finishes one side of the square. The same statements are repeated three more times to draw the remaining three sides.

## Using loops to draw square in Turtle

You can see in the above code that we have used the same statements(forward(100) and left(90)) four times. So, instead of writing them again and again, we can use a loop that will run 4 times.

|  |
| --- |
| #Using loop to draw square in Python Turtle  import turtle    t = turtle.Turtle()  for i in range(4): # for loop will run 4 times    t.forward(100) #Forward turtle by 100 units    t.left(90) #Turn turtle by 90 degree |

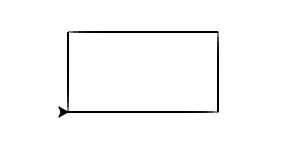
When you run this program, you will the same output.

## Drawing Rectangle in Python Turtle

We know that opposite sides of a rectangle are equal. The angle between two adjacent sides of a rectangle is 90° By keeping these properties in mind, we will draw the rectangle. Suppose the length of the rectangle is 150 units and its breadth is 80 units. Run the below code to obtain rectangle in turtle.

|  |
| --- |
| #Program to draw rectangle in Python Turtle  import turtle    t = turtle.Turtle()  t.forward(150) #Forward turtle by 150 units  t.left(90) #Turn turtle by 90 degree  t.forward(80) #Forward turtle by 80 units  t.left(90) #Turn turtle by 90 degree  t.forward(150) #Forward turtle by 150 units  t.left(90) #Turn turtle by 90 degree  t.forward(80) #Forward turtle by 80 units  t.left(90) #Turn turtle by 90 degree |

Output of the above program



Explanation of the above code-

t.forward(150)

t.left(90)

t.forward(80)

t.left(90)

We have moved the turtle in the forward direction by 150 units. Because the length of a rectangle is 150 units. Then, we have turned the turtle by 90° as the angle between adjacent sides is 90°. This finishes one side of the rectangle. Then we forwarded the turtle by 80 units and turned it by 90° This completes second side of the rectangle. The same statements are repeated one more times to draw the remaining two sides.

## Using loops to draw rectangle in Turtle

Can you think, which set of statements we have to place in for loop? We will place forward(150), left(90), forward(80) and left(90) in a for loop and run it for 2 times.

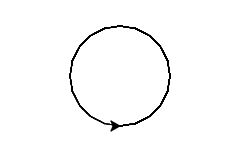
|  |
| --- |
| #Using loop to draw rectangle in Python Turtle  import turtle    t = turtle.Turtle()  for i in range(2):    t.forward(150) #Forward turtle by 150 units    t.left(90) #Turn turtle by 90 degree    t.forward(80) #Forward turtle by 80 units    t.left(90) #Turn turtle by 90 degree |

# **How to draw circle in Python Turtle**

To draw a circle, we will use circle() method which takes radius as an argument.

|  |
| --- |
| #Program to draw circle in Python Turtle  import turtle    t = turtle.Turtle()  t.circle(50) |

Output of the above program



Explanation of the above code

import turtle

t = turtle.Turtle()

You must import turtle module in order to use it. Then, we have created a new drawing board and assigned it to an object t.

t.circle(50)

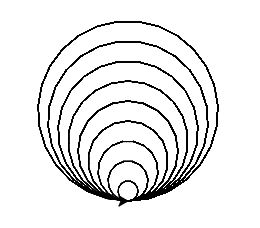
It will draw a circle of radius 50units.

## Draw Tangent Circles in Python Turtle

More than one circle having one point of intersection is called tangent circles.

|  |
| --- |
| #Program to draw tangent circles in Python Turtle  import turtle    t = turtle.Turtle()  for i in range(10):    t.circle(10\*i) |

Output of the above program-



Explanation of the above code

for i in range(10):

t.circle(10\*i)

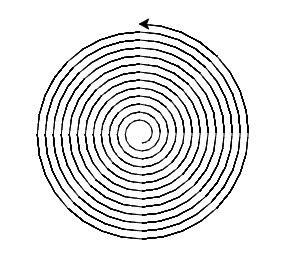
After drawing a circle, turtle reaches the same point from where it has started drawing a circle. So, just by varying the radius we can obtain tangent circles. This thing is repeated 10 times to obtain tangent circles.

## Draw Spiral Circles in Python Turtle

Circles with varying radius are called spiral.

|  |
| --- |
| #Program to draw spiral circles in Python Turtle  import turtle    t = turtle.Turtle()  for i in range(100):    t.circle(10+i, 45) |

Output of the above program-



Explanation of the above code

for i in range(100):

t.circle(10+i, 45)

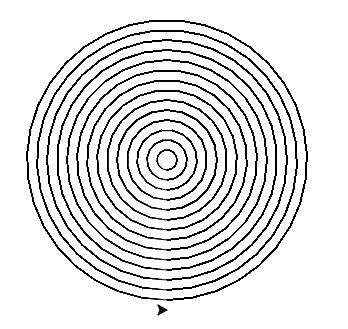
The second argument of circle() method helps in drawing an arc. It controls the measure of the central angle. Here we have passed 45 as a central angle. This thing is repeated 100 times to obtain concentric circles.

## Draw Concentric Circles in Python Turtle

Circles with different radii having a common center are called concurrent circles.

|  |
| --- |
| #Program to draw concentric circles in Python Turtle  import turtle    t = turtle.Turtle()  for i in range(50):    t.circle(10\*i)    t.up()    t.sety((10\*i)\*(-1))    t.down() |

Output of the above program-



Explanation of the above code

for i in range(100):

t.circle(10\*i)

t.up()

t.sety((10\*i)\*(-1))

t.down()

After drawing a circle, we have picked up the turtle's pen and set the y coordinate of turtle's pen to -1 times 10\*i. After which we have put the pen back on the canvas. This thing is repeated 50 times to obtain concentric circles.

# **How to draw color filled shapes in Python Turtle?**

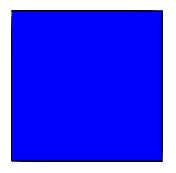
Follow the below steps to draw filled shape with the desired color-

1. Choose the fill color by calling fillcolor() function and pass the color name or color in the #RRGGBB format.
2. After step 1, you must call begin\_fill() and then start drawing using Turtle functions. Once you are done with the drawing, call end\_fill() function to fill the drawn figure with the selected color.

## Draw color filled square in Python Turtle

|  |
| --- |
| #Python program to draw color filled square in turtle programming  import turtle    t = turtle.Turtle()  t.fillcolor('blue')  t.begin\_fill()  for i in range(4):    t.forward(150)    t.right(90)  t.end\_fill() |

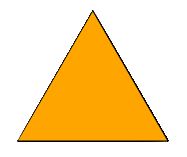
Output of the above program



## Draw color filled triangle in Python Turtle

|  |
| --- |
| #Python program to draw color filled triangle in turtle programming  import turtle    t = turtle.Turtle()  t.fillcolor('#FFA500')  t.begin\_fill()  for i in range(4):    t.forward(150)    t.right(-120)  t.end\_fill() |

Output of the above program



## Draw color filled star in Python Turtle

|  |
| --- |
| #Python program to draw color filled star in turtle programming  import turtle    t = turtle.Turtle()  t.fillcolor('orange')  t.begin\_fill()  for i in range(5):    t.forward(150)    t.right(144)  t.end\_fill() |

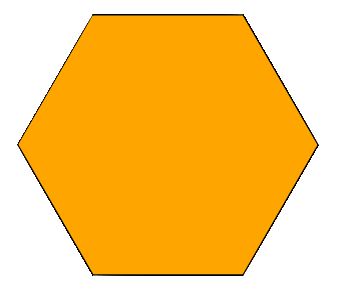
Output of the above program



## Draw color filled hexagon in Python Turtle

|  |
| --- |
| #Python program to draw color filled hexagon in turtle programming  import turtle    t = turtle.Turtle()  t.fillcolor('orange')  t.begin\_fill()  for i in range(6):    t.forward(150)    t.right(60)  t.end\_fill() |

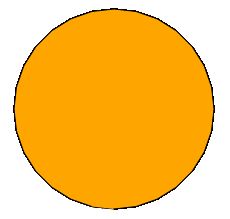
Output of the above program



## Draw color filled circle in Python Turtle

|  |
| --- |
| #Python program to draw color filled circle in turtle programming  import turtle    t = turtle.Turtle()  t.fillcolor('orange')  t.begin\_fill()  t.circle(100)  t.end\_fill() |

Output of the above program

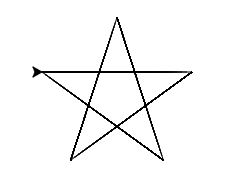


# **How to draw stars in Python Turtle**

When we were in kids school, we were taught of drawing stars. Here also, you will learn how to draw stars using Python programming language.

|  |
| --- |
| #Program to draw stars in Python Turtle  import turtle    t = turtle.Turtle()    for i in range(5):      t.forward(150)      t.right(144) |

Output of the above program



Explanation of the above code-

t.forward(150)

t.right(144)

Here, we will move the turtle forward by 150 units and then turn it towards the right by 144°. Remember, one thing you have to turn the turtle by 144°. IIf you turn it by some other angle then you will not be able to draw the star.

# **How to draw pentagon, hexagon and other polygons in Python Turtle?**

Polygon is a n-sided closed figure. All the sides of a polygon are of equal length. The naming of a polygon depends on how many sides it is having. An exterior angle of a polygon is 360/(number of sides). So, for a pentagon, it will be 72. For a hexagon, it will be 60. And so on.

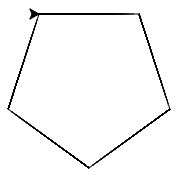
| **Number of Sides** | **Polygon name** | **Exterior Angle** |
| --- | --- | --- |
| 5 | Pentagon | 72 |
| 6 | Hexagon | 60 |
| 7 | Heptagon | 51.42 |
| 8 | Octagon | 45 |
| 9 | Nanogon | 40 |
| 10 | Decagon | 36 |

In order to draw pentagon, hexagon and other polygons, we will use the above-mentioned properties.

## Draw Pentagon in Python Turtle

|  |
| --- |
| #Python programming to draw pentagon in turtle programming  import turtle    t = turtle.Turtle()  for i in range(5):     t.forward(100) #Assuming the side of a pentagon is 100 units     t.right(72) #Turning the turtle by 72 degree |

Output of the above program



Explanation of the above code-

for i in range(5):

t.forward(100)

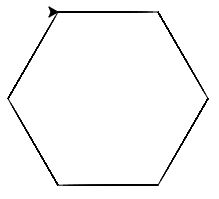
t.right(72)

We are assuming the side of a pentagon is 100 units. So, we will move the turtle in the forward direction by 100 units. And then turn it in the clockwise direction by 72°. Because the exterior angle of a pentagon is 72° These two statements are repeated 5 times to obtain Pentagon.

## Draw Hexagon in Python Turtle

|  |
| --- |
| #Python programming to draw hexagon in turtle programming  import turtle    t = turtle.Turtle()  for i in range(6):     t.forward(100) #Assuming the side of a hexagon is 100 units     t.right(60) #Turning the turtle by 60 degree |

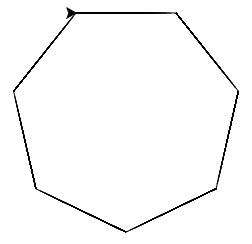
Output of the above program



## Draw Heptagon in Python Turtle

|  |
| --- |
| #Python programming to draw heptagon in turtle programming  import turtle    t = turtle.Turtle()  for i in range(7):     t.forward(100) #Assuming the side of a heptagon is 100 units     t.right(51.42) #Turning the turtle by 51.42 degree |

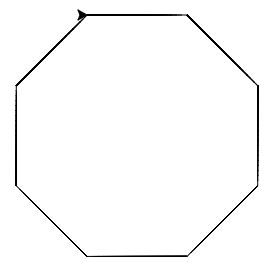
Output of the above program



## Draw Octagon in Python Turtle

|  |
| --- |
| #Python programming to draw octagon in turtle programming  import turtle    t = turtle.Turtle()  for i in range(8):     t.forward(100) #Assuming the side of a octagon is 100 units     t.right(45) #Turning the turtle by 45 degree |

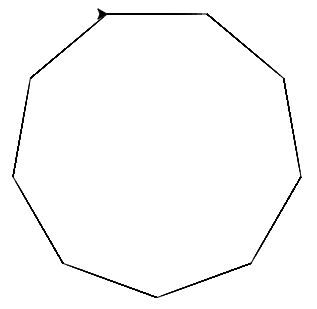
Output of the above program



## Draw polygon in Python Turtle

|  |
| --- |
| #Python programming to draw polygon in turtle programming  import turtle    t = turtle.Turtle()  numberOfSides = int(input('Enter the number of sides of a polygon: '))  lengthOfSide = int(input('Enter the length of a side of a polygon: '))  exteriorAngle = 360/numberOfSides  for i in range(numberOfSides):     t.forward(lengthOfSide)     t.right(exteriorAngle) |

Output of the above program

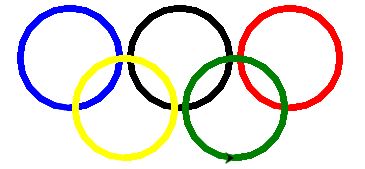
User Input in Python Turtle 

# **How to draw Olympics logo in Python Turtle?**

Olympics logo is made up of five interlaced rings having the following colors- blue, black, red, yellow and green. The basic logic of drawing it is using the proper combination of goto() and circle() function.

|  |
| --- |
| #Python Program to draw Olympics logo in Turtle Programming  import turtle    t = turtle.Turtle()  t.pensize(6) #Set the thickness of the pen to 6  firstRowColors = ["blue", "black", "red"] #firstRowColors is a list of colors that are present in the first row of logo  for i in range(3):    t.penup()    t.pencolor(firstRowColors[i])    t.goto(i\*110, 0)    t.pendown()    t.circle(50)    secondRowColors = ["", "yellow", "", "green"]  for i in range(1, 4, 2):    t.penup()    t.pencolor(secondRowColors[i])    t.goto(i\*55, -50)    t.pendown()    t.circle(50) |

Output of the above program



Explanation of the above code-

import turtle

olympics = turtle.Turtle()

To draw any shape using Turtle, you have to import turtle module. After this, create a new drawing board and assign it to an object t.

firstRowColors = ["blue", "black", "red"]

for i in range(3):

t.penup()

t.pencolor(firstRowColors[i])

t.goto(i\*110, 0)

t.pendown()

t.circle(50)

Colors are stored in a list and are applied to the turtle's pen based on the iteration. Here, we are moving the turtle to the desired position, and then drawing the circle of 50 units. This thing is repeated 3 times to obtain first row of Olympics logo.

secondRowColors = ["", "yellow", "", "green"]

for i in range(1, 4, 2):

t.penup()

t.pencolor(secondRowColors[i])

t.goto(i\*55, -50)

t.pendown()

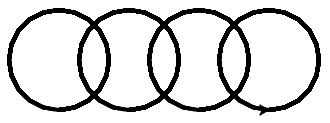
t.circle(50)

The logic of the first row is applied for the second row.

## How to draw Audi logo in Python Turtle?

|  |
| --- |
| #Python Program to draw Audi logo in Turtle Programming  import turtle    t = turtle.Turtle()  t.pensize(4)    for i in range(4):    t.penup()    t.goto(i\*70, 0)    t.pendown()    t.circle(50) |

Output of the above program



Explanation of the above code-

for i in range(4):

t.penup()

t.goto(i\*70, 0)

t.pendown()

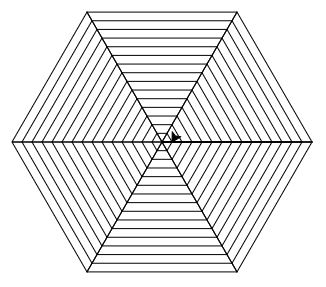
t.circle(50)

# **How to draw spiderweb in Python Turtle?**

Spiderweb consists of radical and spiral threads. First of all, we will build radical thread and then spiral threads.

|  |
| --- |
| #Python program to draw spider web in turtle programming  import turtle    t = turtle.Turtle()  t.speed(0)    #Code for building radical thread  for i in range(6):    t.forward(150)    t.backward(150)    t.right(60)    #Code for building spiral thread  side = 150  for i in range(15):    t.penup()    t.goto(0,0)    t.pendown()    t.setheading(0)    t.forward(side)    t.right(120)    for j in range(6):      t.forward(side)      t.right(60)    side = side - 10 |

Output of the above program



Explanation of the above code-

side = 150

for i in range(15):

t.penup()

t.goto(0,0)

t.pendown()

t.setheading(0)

t.forward(side)

t.right(120)

for j in range(6):

t.forward(side)

t.right(60)

side = side - 10

There are two for loops- Outer for loop will control the building of spiral threads and inner for loop will build a single spiral thread. In each iteration, we will reduce the length of side by 10.

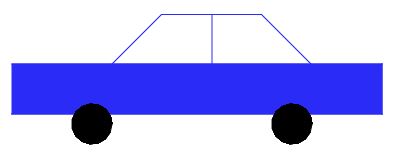
## Recommended Posts

# **How to draw car in Python Turtle?**

To draw a car in Python turtle, you must think it in terms of particular shapes. Tyres can be drawn using circle() function. The upper body of a car can be think of as a rectangle. And roof and window are similar to trapezoid.

|  |
| --- |
| #Python program to draw car in turtle programming  import turtle    car = turtle.Turtle()    #Below code for drawing rectanglura upper body  car.color('#2B2BF6')  car.fillcolor('#2B2BF6')  car.penup()  car.goto(0,0)  car.pendown()  car.begin\_fill()  car.forward(370)  car.left(90)  car.forward(50)  car.left(90)  car.forward(370)  car.left(90)  car.forward(50)  car.end\_fill()    #Below code for drawing window and roof  car.penup()  car.goto(100, 50)  car.pendown()  car.setheading(45)  car.forward(70)  car.setheading(0)  car.forward(100)  car.setheading(-45)  car.forward(70)  car.setheading(90)  car.penup()  car.goto(200, 50)  car.pendown()  car.forward(49.50)    #Below code for drawing two tyres  car.penup()  car.goto(100, -10)  car.pendown()  car.color('#000000')  car.fillcolor('#000000')  car.begin\_fill()  car.circle(20)  car.end\_fill()  car.penup()  car.goto(300, -10)  car.pendown()  car.color('#000000')  car.fillcolor('#000000')  car.begin\_fill()  car.circle(20)  car.end\_fill()    car.hideturtle() |

Output of the above program



# **How to draw snowman in Python Turtle?**

Snowman consists of circular shaped snowballs having different radius. The body of the snowman is made up of three snowballs placed one over the other. Eyes, nose, and buttons are also circular in shape. Keeping this thing in mind, we will draw snowman.

|  |
| --- |
| import turtle    t = turtle.Turtle()    def draw\_circle(color, radius, x, y):    t.penup()    t.fillcolor(color)    t.goto(x,y)    t.pendown()    t.begin\_fill()    t.circle(radius)    t.end\_fill()    #Below three statements for drawing snowman body  draw\_circle("#ffffff", 30, 0, -40)  draw\_circle("#ffffff", 40, 0, -100)  draw\_circle("#ffffff", 60, 0, -200)    draw\_circle("#ffffff", 2, -10, -10) #Drawing left eye  draw\_circle("#ffffff", 2, 10, -10) #Drawing right eye  draw\_circle("#FF6600", 3, 0, -15) #Drawing nose    #Below three statements for drawing buttons  draw\_circle("#ffffff", 2, 0, -35)  draw\_circle("#ffffff", 2, 0, -45)  draw\_circle("#ffffff", 2, 0, -55)    #Code for drawing left arm  t.penup()  t.goto(-15,-35)  t.pendown()  t.pensize(5)  t.goto(-75, -50)  #Code for drawing right arm  t.penup()  t.goto(15,-35)  t.pendown()  t.pensize(5)  t.goto(75, -50)    #Code for drawing hat  t.penup()  t.goto(-35, 8)  t.color("black")  t.pensize(6)  t.pendown()  t.goto(35, 8)    t.goto(30, 8)  t.fillcolor("black")  t.begin\_fill()  t.left(90)  t.forward(15)  t.left(90)  t.forward(60)  t.left(90)  t.forward(15)  t.end\_fill() |

Output of the above program



Explanation of the above code-

def draw\_circle(color, radius, x, y):

t.penup()

t.fillcolor(color)

t.goto(x,y)

t.pendown()

t.begin\_fill()

t.circle(radius)

t.end\_fill()

This function takes 4 arguments- color and radius of the circle, x and y coordinates where we want to move the turtle.

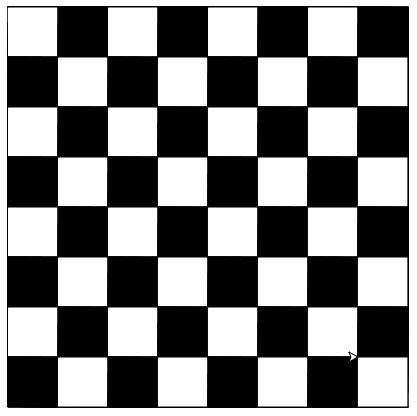
# **How to draw chessboard in Python Turtle?**

Chess is one of the most popular indoor game that is played on a chessboard. The chessboard has a squared shape and consists of sixty-four subdivisions. Here, subdivisions are painted with alternate colors. Keeping this thing in mind, we will draw chessboard in turtle.

Several functions from the turtle library will be used. These are- forward(), right(), penup(), pendown(), goto(), fillcolor(), begin\_fill() and end\_fill().

|  |
| --- |
| #Program to draw chessboard in Python Turtle  import turtle  chessboard = turtle.Turtle()  chessboard.speed(8) #for speedily drawing chessboard  for i in range(4): #for loop will run 4 times      chessboard.forward(800) #forward turtle by 800 units      chessboard.right(90) #turn turtle clockwise by 90 degree  a = 0 #for controlling alternate colors in a row  b = 0 #for controlling alternate colors in a column  for i in range(8): #for loop will run 8 times as there are 8 rows in the chessboard      if(b == 0):          a=1      else:          a=0      for j in range(8): #for loop will run 8 times as there are 8 columns in the chessboard          chessboard.penup()          chessboard.goto(j\*100, i\*100\*(-1))          chessboard.pendown()          if(a==0):              chessboard.fillcolor('black')              a=1          else:              chessboard.fillcolor('white')              a=0          chessboard.begin\_fill()          for k in range(4):              chessboard.forward(100)              chessboard.right(90)          chessboard.end\_fill()      if(b==0):          b=1      else:          b=0 |

Output of the above program-



Explanation of the above code-

import turtle

chessboard = turtle.Turtle()

To work with turtle module, you have to import it. After importing this module, you have to create a new drawing board and assign it to an object chessboard.

for i in range(4):

chessboard.forward(800)

chessboard.right(90)

We have moved the turtle in the forward direction by 800 units. Because the length of the outer boundary of the chessboard is 800 units. Then, we have turned the turtle by 90° as the angle between adjacent sides is 90°. This finishes one side of the chessboard. These same statements are repeated 3 times to obtain the remaining boundary of the chessboard.

for i in range(8):

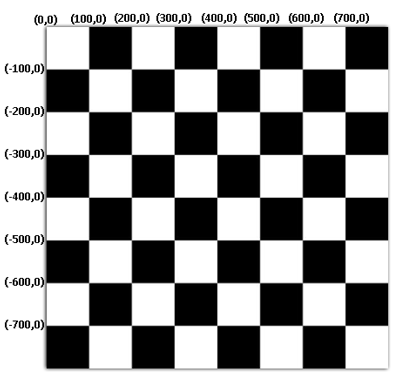
if(b == 0):

a=1

else:

a=0

b variable will be used to control the color of first square of each row.



for j in range(8):

chessboard.penup()

chessboard.goto(j\*100, i\*100\*(-1))

chessboard.pendown()

if(a==0):

chessboard.fillcolor('black')

a=1

else:

chessboard.fillcolor('white')

a=0

chessboard.begin\_fill()

for k in range(4):

chessboard.forward(100)

chessboard.right(90)

chessboard.end\_fill()

The above for loop will control the drawing of squares in a row. goto() will move the turtle to a specific left corner of a square. If the value of a is equal to 0 then we have set the fillcolor to black and otherwise we have set the fillcolor to white. After this, we have called begin\_fill() to start filling the square with the selected fillcolor. Then, we have drawn the square of 100 units side.

# **How to draw tally marks in Python Turtle?**

Tally marks are generally used for counting numbers. You can also use it in counting scores of games. While counting numbers, you draw a vertical line for each number and every fifth number is drawn diagonally that passes through the previous four vertical lines. For example, tally marks for the number 9 is

Tally Marks for number 9

## Logic for drawing tally marks

We will draw tally marks in the form of square. And taking the side of the square as 30 units. Here, the distance between each vertical line is 10 units. So, four vertical lines will form a square of 30 units side. Every fifth number is drawn diagonally and its length equal to 30x√2

|  |
| --- |
| #Python program to draw tally marks in turtle programming  import turtle  import math    tallymarks = turtle.Turtle()  number = int(input("Enter a number: ")) #Asking user to enter a number  tallymarks.right(90)  x = 0  for i in range(1,number+1):    if(i%5 == 0): #For every fifth number, it will draw diagonal line      tallymarks.right(135)      tallymarks.forward(30\*math.sqrt(2))      tallymarks.right(225)    else: #For other numbers, it will draw vertical line      tallymarks.penup()      tallymarks.goto(x\*10,0)      tallymarks.pendown()      tallymarks.forward(30)    x = x + 1 |

Output of the above program

Assuming that user has entered 34. So, we will get tally marks accordingly.

Tally Marks in Python Turtle

Explanation of the above code-

if(i%5 == 0):

tallymarks.right(135)

tallymarks.forward(30\*math.sqrt(2))

tallymarks.right(225)

For every fifth number, we will move the turtle 30x√2 units in such a way that it will be drawn as a diagonal.

else:

tallymarks.penup()

tallymarks.goto(x\*10,0)

tallymarks.pendown()

tallymarks.forward(30)

x = x + 1

For other numbers, we will draw a vertical line by moving the turtle to the specific position using goto() function.

…………………………………

1. Prior to drawing anything using the turtle, you must start by importing the turtle module.
2. import turtle
3. or

from turtle import \*

1. Call turtle.Turtle() to create a new drawing board and assign it to an object t.

t = turtle.Turtle()

1. Start drawing by calling the different methods of turtle such as forward(), left(), right(). etc.
2. When you are done with the drawing, call turtle.done()

# **How to draw color filled shapes in Python Turtle?**

Follow the below steps to draw filled shape with the desired color-

1. Choose the fill color by calling fillcolor() function and pass the color name or color in the #RRGGBB format.
2. After step 1, you must call begin\_fill() and then start drawing using Turtle functions. Once you are done with the drawing, call end\_fill() function to fill the drawn figure with the selected color.

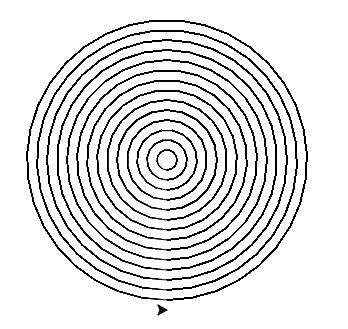
## Draw color filled square in Python Turtle

|  |
| --- |
| #Python program to draw color filled square in turtle programming  import turtle    t = turtle.Turtle()  t.fillcolor('blue')  t.begin\_fill()  for i in range(4):    t.forward(150)    t.right(90)  t.end\_fill() |

Output of the above program

|  |
| --- |
| #Program to draw concentric circles in Python Turtle  import turtle    t = turtle.Turtle()  for i in range(50):    t.circle(10\*i)    t.up()    t.sety((10\*i)\*(-1))    t.down() |

Output of the above program-



Explanation of the above code

for i in range(100):

t.circle(10\*i)

t.up()

t.sety((10\*i)\*(-1))

t.down()

After drawing a circle, we have picked up the turtle's pen and set the y coordinate of turtle's pen to -1 times 10\*i. After which we have put the pen back on the canvas. This thing is repeated 50 times to obtain concentric circles.