007: Libraries

Learning Outcome:	Meaning of libraries, import function, use of turtle, random.

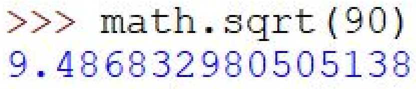
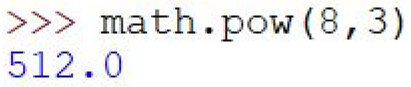
Definitions/Concepts	
Modules	● Module is simply a file containing python code. ● Every python file you create and save can be used as a module. ● Modules can have variables and functions. ● The creators of python have also created a lot of modules which can be used to do different useful tasks. ● The set of all these modules which comes along with Python is called *The Python Standard library*.
import	● Using the keyword import, makes python aware that we are going to use a module in the current program. ● Syntax: *import module_name*
Using functions of a module	● Modules contain functions which can be used in our program. ● Syntax: *module_name.function_name()*

math Module

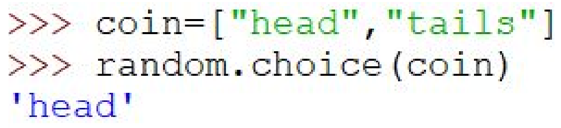
- This module provides access to the mathematical functions in python.
- To use this module we can simply use the keyword **import**
- **Syntax:**

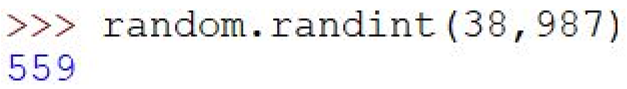
```
import math
```

Let's see some functions of the math Module.

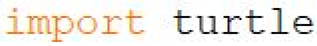
sqrt	● It is used to find the square root of a number. ● Syntax: *math.sqrt(number)*	 `>>> math.sqrt(90)` `9.486832980505138`
pow	● It is used to find the result of a number raised to a power. ● Syntax: *math.pow(number, power)*	 `>>> math.pow(8,3)` `512.0`

random Module	
● To use this module we can simply use the keyword **import** **Syntax:** `import random`	
Let's see some functions of the random Module.	
choice	● To choose a random element from a sequence like list or tuple, we can use the function choice. ● Syntax: *random.choice(sequence)*

 `>>> coin=["head","tails"]`
`>>> random.choice(coin)`
`'head'`	

randint	● This function returns a random integer in a given interval, including the start and stop value of the interval. ● Syntax: *random.randint(start, stop)*
 `>>> random.randint(38,987)` `559`	

turtle Module

- This module helps us draw using a turtle.
- Turtle is like a pointer, which we can control using different commands.
- To use this module we can simply use the keyword **import**
- **Syntax:** 

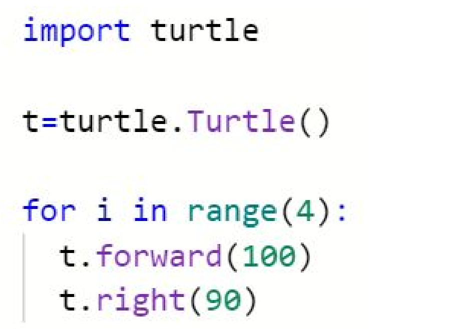
```
import turtle
```

Let's see some functions of the turtle module.

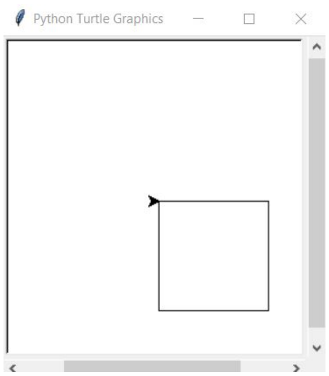
Turtle	● It is used to create and name a turtle. ● We can create as many turtles as we want. Syntax: *t1=turtle.Turtle()* ● Here, t1 is the variable which store the turtle object, now we will refer to this turtle by the name t1 ● turtle.Turtle() follows the syntax of module_name.function_name
forward	● It is used to move the turtle forward by a given length in pixels. Syntax: *turtle_name.forward(length)*
backward	● It is used to move the turtle backward by a given length in pixels. Syntax: *turtle_name.backward(length)*
right	● It is used to make the turtle turn clockwise by a given angle in degrees. Syntax: *turtle_name.right(angle)*
left	● It is used to make the turtle turn anti-clockwise by a given angle in degrees. Syntax: *turtle_name.left(angle)*

Making pattern using turtle Module

Square

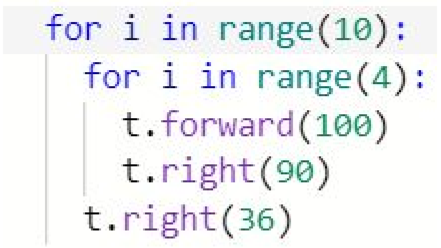


```
import turtle

t=turtle.Turtle()

for i in range(4):
    t.forward(100)
    t.right(90)
```

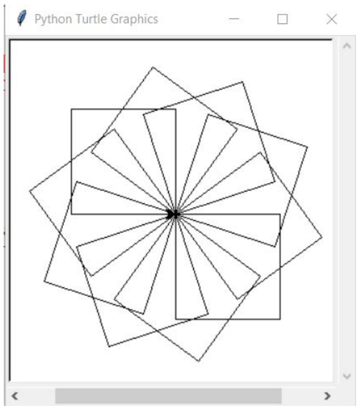


- The turtle moves 100 pixels ahead and then turns 90 degree clockwise and this gets repeated 4 times by the loop so we get a square.

Square Pattern



```
for i in range(10):
    for i in range(4):
        t.forward(100)
        t.right(90)
    t.right(36)
```

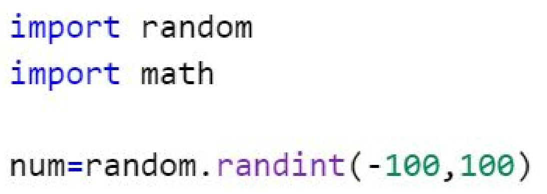


- Here, we have a nested loop: a for loop inside another for loop.
- The inner loop is the same as the loop used to make a square.
- The outer loop repeats the process of making squares **10 times**.
- And the **t.right(36)** turns the turtle by 36 degrees each time so that the squares don't overlap and the pattern is formed.
- We get a perfect circular pattern as the turtle turns 36 degrees 10 times, which makes the total turn angle to be 360 degrees, the angle in a circle.

Activity links and Solutions

Student Activity 1: Standard Library

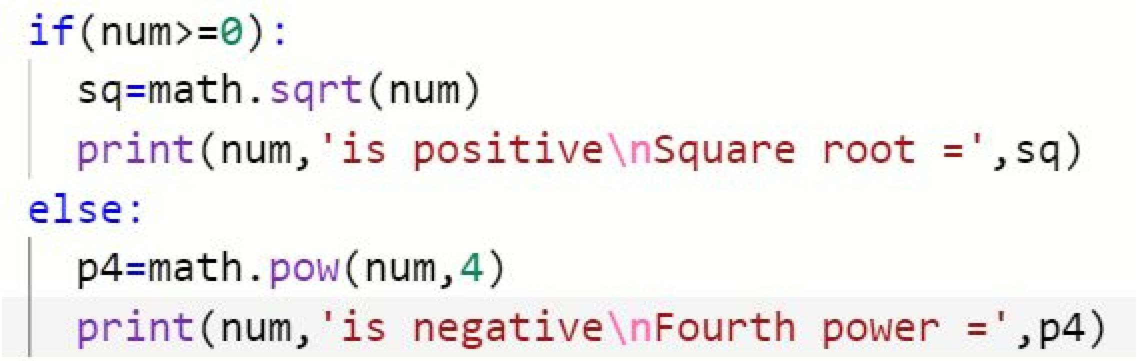
#Problem: Write a program to take a random number between -100 to 100, if the number is positive, display its square root, if the number is negative, display its 4th power.(Hint: You have to use two modules to solve this question.)



```
import random
import math

num=random.randint(-100,100)
```

- To take a random number between a given range, we need the **random** module.
- To find square root and fourth power we need the **math** module.

Now after storing a random number in the variable **num,** we need to use the **if** statement to check if it is positive or negative and then accordingly we will display the output.



```
if(num>=0):
  sq=math.sqrt(num)
  print(num,'is positive\nSquare root =',sq)
else:
  p4=math.pow(num,4)
  print(num,'is negative\nFourth power =',p4)
```

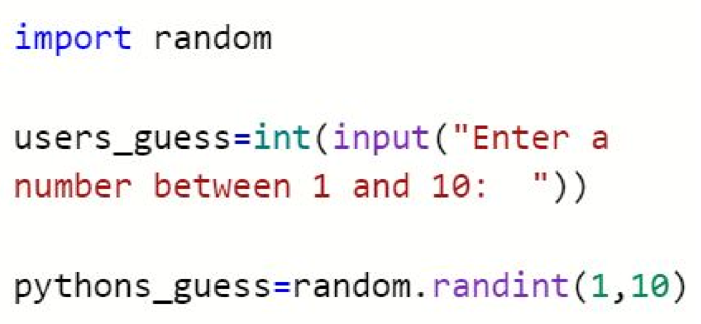
Here, the variables **sq** and **p4** store square root and fourth power of the random no.

Do you remember the escape sequence **\n?**
It is used to create a new line in output.

Student Activity 2: Random

#Activity 1: Take a number as input from user in a given interval. Also take a random number from the computer in the same interval. If the user input is greater than random number then display "You Won" otherwise display "You lose"

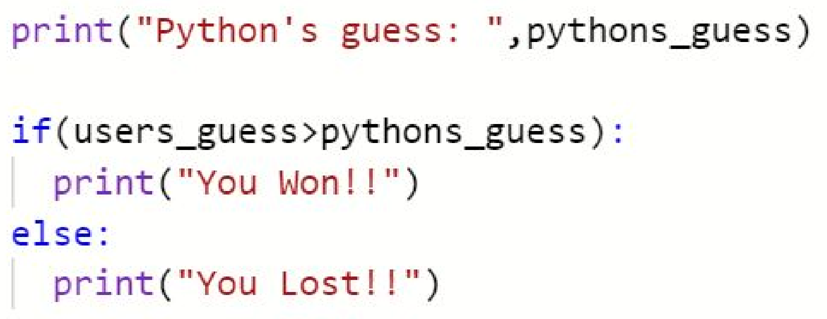
- Let’s take the input interval to be 1 to 10.
- We need two variables, one to store the user’s guess and the other to store the Python’s guess.
- User’s guess will be an input from the user between 1 to 10
- Python’s guess will be a random integer from 1 to 10



```
import random

users_guess=int(input("Enter a number between 1 and 10:  "))

pythons_guess=random.randint(1,10)
```

- We used the **random** module for the function **randint**
- We used the **int** function to convert the user input from String to integer.

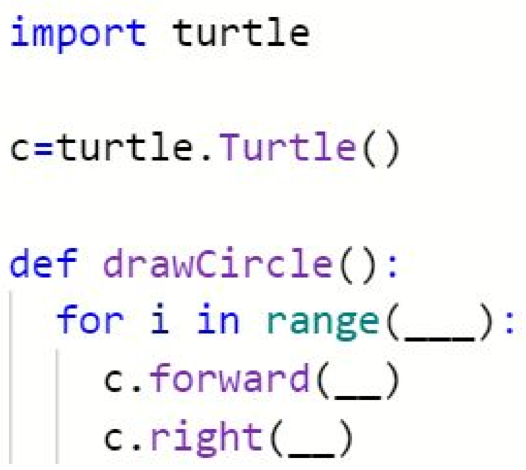
- Now, we will display the Python’s guess and then display the result of the game according to the rules using the if statement.



```
print("Python's guess:  ",pythons_guess)

if(users_guess>pythons_guess):
  print("You Won!!")
else:
  print("You Lost!!")
```

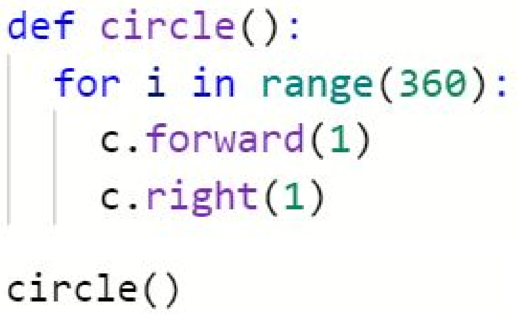
Student Activity 2: Turtle

# Activity 1: Complete the below code to create a function to draw a circle.



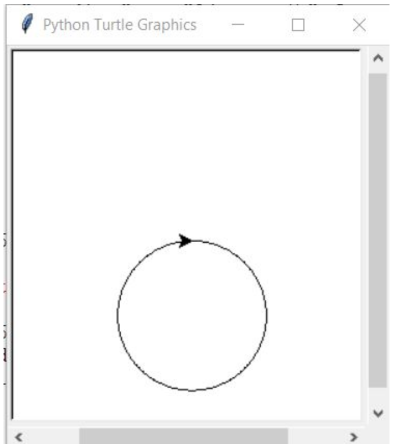
```python
import turtle

c=turtle.Turtle()

def drawCircle():
  for i in range(___):
    c.forward(__)
    c.right(__)
```

- Here, we have a turtle named **c**
- We have to create a function to draw a circle.
- The central angle of a circle is 360 degrees.
- So the loop must run 360 times
- The turtle will move forward by 1 pixel and turn in either clockwise or anti-clockwise direction by 1 degree.

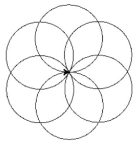


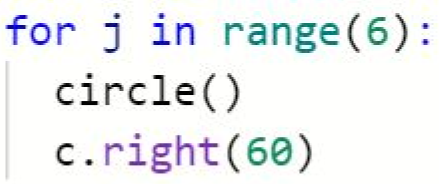
```python
def circle():
  for i in range(360):
    c.forward(1)
    c.right(1)

circle()
```

Here, we have called the function to see the output once.



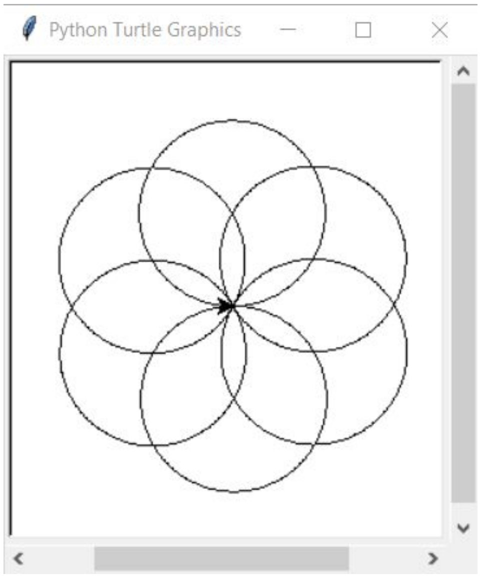
# Activity 2: Use the function drawCircle to create the pattern shown in the image file named CirclePattern (Hint: there are 6 circles in the circular pattern so guess the angle between them)





```python
for j in range(6):
  circle()
  c.right(60)
```

- We have to draw 6 circles so the outer loop has to run 6 times.
- To make the whole pattern circular, we need the turtle to make a total turn of 360 degrees

Output:  	- Since the turn will happen 6 times, so we used the statement **c.right(60)** as 6 times 60 will form 360 degrees.

Fun-Fact

To explore more functions of these modules you can simply write *help("module_name")* in python. It will give you the list of functions with their descriptions and syntax.