Task 3.1 - NUMBERS

1. What are Numbers in Python?

All numbers that are handled will belong to some type. A type is a way to define various data structures and containers and define the functionality associated with them. Types are implemented in Python as classes.

2. Numeric Types: Python supports four different numerical types −

a) int (signed integers) − They are often called just integers or ints, are positive or negative whole numbers with no decimal point.

b) long (long integers) − Also called longs, they are integers of unlimited size, written like integers and followed by an uppercase or lowercase L.

c) float (floating point real values) − Also called floats, they represent real numbers and are written with a decimal point dividing the integer and fractional parts. Floats may also be in scientific notation, with E or e indicating the power of 10 (2.5e2 = 2.5 x 10^2 = 250).

d) complex (complex numbers) − are of the form a + bJ, where a and b are floats and J (or j) represents the square root of -1 (which is an imaginary number). The real part of the number is a, and the imaginary part is b. Complex numbers are not used much in Python programming.

3. Numeric Operations:

General mathematical operations such as addition, subtraction, and multiplication can be done on them.

i. **Operations when the type are the same:** When the type of the input numbers are the same the result will be a number of the same class that the input numbers belong. This is true except the case of division.

```
# assign the integer value 2 to a variable ‘A’

A = 2

# assign the integer value 3 to a variable ‘B’

B = 3

# print the sum of the two variables ‘A and B’

print( A + B )



Output: 5
```

Similarly if we consider two decimal numbers and perform a mathematical operation then we will get:

```
# assign the float value 9.8 to a variable ‘C’
C = 9.8
# assign the float value 5.4 to a variable ‘D’
D = 5.4
# print the subtraction of the two variables ‘C and D’
print( C - D )
```

Output: 4.4

Even, we can perform such simple mathematical operations on complex numbers also.

Example:

```
# assign a complex number to a variable ‘E’
E = ( 1 + 2j)
# assign a complex number to a variable ‘F’
F = ( 3 +5j)
#print sum of the two complex numbers
print( E + F)
```

Output: (4 +7j)

But in the case of division, dividing two integers will always give a float value in as a result.

Example:

```
# divide two integers  and assign it to variable result.
Result  = 4/2
#print the result
print(Result)
```

Output: 2.0

ii. **Operations between numbers of different types:** The operations work when operate of variables are of different numeric types as well. For example, we can add two numbers, one with class int and another with class float. The result will be in float. This is done by changing the integer to a float and then adding them.

```
# add two numbers , one is integer and another is float

Result = 2 +5.1

#print the result

print(Result)


Output: 7.1
```

Other operations like adding int to complex or float to complex works in similar manner. The result of both kind of operations belong to the complex class.

```
# add two numbers belonging to complex and int class

Result = ( 2 + 3j) + 4

# add two numbers belonging to complex and float class

Sum = (2 +3j) + 4.12

#print the result and the sum

print(Result)

print(Sum)


Output: ( 6 + 3j)

        ( 6.12 +3j)
```

4. Number Type Conversion:

- We can convert one type of number into another. This is also known as coercion.
- Python converts numbers internally in an expression containing mixed types to a common type for evaluation.
- But sometimes, you need to coerce a number explicitly from one type to another to satisfy the requirements of an operator or function parameter.
- Operations like addition, subtraction coerce integer to float implicitly (automatically), if one of the operand is float.

Example:

```
1 + 2.0

Output: 3.0
```

- From the above example it can be clearly observed that 1(integer) is coerced into 1.0(float) for addition and the result is also a floating point number.
- The built-in functions like int( ), float( ), complex( ) can also be used to convert between types explicitly.

Example:

```
i.    int(2.5)
      Output : 2

ii.   float(5)
      Output: 5.0

iii.  complex(3)
      Output: (3 + 0j)
```

5. Mathematical Functions:

Python includes several functions that perform mathematical calculations:

Sr. No.	Function	Function Description	Syntax	Example
1.	abs( )	The absolute value of x: the positive distance between x and zero.	abs(x)	abs(-45) Output: 45
2.	cmp( )	It returns the sign of the difference of two numbers: -1 if x<y, 0 if x = y, or 1 if x>y.	cmp(x, y)	cmp(80,100) Output: -1

3.	exp( )	➢ It returns exponential of x: e^x.
		➢ This function isn't directly accessible so we've to import math function first.
			import math
			math.exp(x)
				import math
				math.exp(100.12)
				Output: 3.03e+43

4.	log( )	It returns natural logarithm of x, for x>0.	import math
			math.log(x)
				import math
				math.log(100.12)
				Output: 4.606369

5.	log10( )	It returns base-10 logarithm of x for x>0.	import math
			math.log10(x)
				import math
				math.log10(100.12)
				Output: 2.0005208

Sr. No.	Function	Function Description	Syntax	Example
6.	sqrt( )	It returns the square root of x for x>0.	import math math.sqrt(x)	import math math.sqrt(100) Output: 10.0
7.	round( )	It returns x rounded to n digits from the decimal point.	round(x, n)	round(80.23456,2) Output: 80.23
8.	pow( )	It returns x to the power of y. if the third argument z is given, it returns x to the power of y modulus z, i.e. pow (x, y) % z.	import math math.pow(x, y,[z])	import math math.pow(100,2) Output: 10000.0
9.	min( )	It returns the smallest o its arguments.	min(x, y, z,…..)	min(80,100,1000) Output: 80
10.	max( )	It returns the largest of its arguments.	max(x, y, z,….)	max(80,100,1000) Output: 1000

Note : Wherever " import math" is written that function isn't directly accessible so we've to import math function first.

6. **Random Number Functions**: Random numbers are used for games, simulations, testing, security, and privacy applications. Python includes following functions that are commonly used:

Sr No	Function	Description	Syntax	Example
1.	choice( )	It returns a random item from a list, tuple or string.	import random choice(seq)	import random random.choice([1,2 ,3,5,9]) Output: 2
2.	randrange( )	It returns a randomly selected element from range(start, stop, step).	import random randrange ([start], stop, [step])	import random random.randrange (100, 1000, 2) Output: 976
3.	seed( )	It sets the integer starting value used in generating random numbers.	import random( ) seed([x])	import random random.seed(10) Output: 0.57140259469
4.	shuffle( )	It randomizes the items of a list in place.	import random random.shuffle (lst)	import random list = [20, 16, 10, 5] random. shuffle(list) Output: reshuffled list: [16, 5, 10, 20]

5.	uniform( )	It returns a random float r, such that x is less than or equal to r and r is less than y.	import random random.uniform(x ,y)	import random random.uniform(5, 10) Output: 5.52615217015

Note: These functions are not accessible directly, so we need to import random module and then we need to call this function using random static object.

7. **Trigonometric Functions:** Python includes several functions used for trigonometric calculations.

Sr. No.	Function	Description	Syntax	Example
1.	acos( )	It returns the arc cosine of x, in radians.	import math acos(x)	import math math.acos(0) Output: 1.570
2.	asin( )	It returns the arc sine of x, in radians	import math asin(x)	import math math.asin(0) Output: 0.0
3.	atan( )	Return the arc tangent of *x*, in radians.	import math atan(x)	import math math.atan(0) Output: 0.0
4.	atan2( )	Return atan(y/x) in radians. The result is between –pi and pi.	import math atan2(y,x)	import math math.atan2(5,5) Output: 0.7853
5.	cos( )	Returns the cosine of x radians.	import math cos(x)	import math math.cos(3) Output: -0.9899
6.	Hypot (*coordinates)	Returns the Euclidean norm, sqrt(x*x + y*y).	import math hypot(x,y)	import math math.hypot(3,2) Output: 3.6055
7.	sin( )	Returns the sine of x radians.	import math sin(x)	import math math.sin(3) Output: 0.1411
8.	tan( )	Returns the tangent of x radians.	import math tan(x)	import math math.tan(3) Output: -0.1425

Note: These functions are not accessible directly, so we need to import math module and then we need to call this function using math static object.

8. **Angular Conversion:** In Python, there are some of the built-in functions which are defined in math module-they can be used for angular conversion i.e. to convert angle values, there are two angular conversion functions:

Sr. No.	Function	Description
1.	math.degrees( )	It is used to convert angle value from radians to degrees.
2.	math.radians( )	It is used to convert angle value from degrees to radians.

Example:

```
#importing math module
import math
# angle x in radians
x = 10.25
print("x in radians: ",x)
#converting to degrees
x = math.degrees(x)
print("x in degrees: ",x)
#now again converting to radians
x = math.radians(x)
print("x in radians: ",x)
```

Output:

```
x in radians: 10.25
x in degrees: 587.2817400090938
x in radians: 10.25
```

9. Mathematical Constant:

- pi: this is inbuilt constant that outputs the value of pi(3.141592).
- e: this is inbuilt constant that outputs the value of e(2.718281).

```
# importing “math” for mathematical operation
import math
#  returning the value of constant pi
print(“the value of constant pi is:”, end=””)
print(math.pi)
#  returning the value of constant e
print(“the value of constant e is:”, end=””)
print(math.e)
```

Output:

```
The value of constant pi is: 3.141592653589793
The value of const. e is : 2.718281828459045
```

References:

- https://docs.python.org/3/library/numbers.html
- https://docs.python.org/3/library/math.html
- https://www.geeksforgeeks.org/mathematical-functions-in-python-set-4-special-functions-and-constants/
- https://www.programiz.com/python-programming/numbers
- https://www.w3schools.com/python/python_numbers.asp
- https://www.tutorialspoint.com/python/python_numbers.htm

… Happy Learning ! ...