Introduction to Python

Variables and Data Types

Java

- In Java variables are statically typed. Which means we need to mention the data type while declaring a variable.

```
public class DeclaringVariable{
    public static void main(String[] args){
        int a = 123;
        float b = 1.23f;
        double c = 1.23;
        String d = "Hello World!";
        boolean e = true;
        boolean f = false;
    }
}
```

Python

- Variables are dynammically typed. No need to write data type while declaring a variable.

```
a = 123           #integer
b = 1.23          #float
c = 1 + 2j        #complex
d = "Hello World" #string
e = True          #boolean true
f =  False        #boolean false
```

Printing the variables

Java

```
public class PrintingVariable{
    public static void main(String[] args){
        System.out.println(a);
        System.out.println(b);
        System.out.println(c);
        System.out.println(d);
        System.out.println(e);
        System.out.println(f);
    }
}
```

```
# Output
123
1.23
1.23
Hello World
true
false
```

Python

```
print (a)
print (b)
print (c)         #Bracket for complex number
print (d)
print (e)
print (f)
```

```
# Output
123
1.23
(1+2j)
```

```
Hello World
True
False
```

Checking types

Java

- You cannot check primitive types at runtime because java is statically typed. However for non-primitive data types, you can use `instanceof` .

```
public class CheckType{
    public static void main(String[] args){
        Object a = 123;
        System.out.println(a instanceof Integer);
    }
}
```

```
# Output
true
```

Python

```
c = 1 + 2j

type(c)
```

```
# Output
complex
```

Basic arithmetic operations

Java

- For floor division both numbers should be of `int` datatype.

```
public class PrintingVariable{
    public static void main(String[] args){
        System.out.println(5 + 2);
        System.out.println(5 - 2);
        System.out.println(5 * 2);
        System.out.println(5.0 / 2);
        System.out.println(5 / 2);
        System.out.println(5 % 2);
        System.out.println(Math.pow(2,4));
    }
}
```

```
# Output
7
3
10
2.5
2
1
16
```

Python

```
print (5 + 2)
print (5 - 2)
print (5 * 2)
print (5 / 2)
print(5//2)
print (5 % 2)
print (2 ** 4)
```

```
# Output
7
3
10
2.5
2
1
16
```

Concatination

Java

```
public class Contatination{
    public static void main(String[] args){
        int a = 123;
        int b = 456;
        System.out.println("The current value of a is "+ a);
        System.out.println("The current value of a is " + a +
" and b is" + b);
    }
}
```

```
# Output
The current value of a is 123
The current value of a is 123 and b is 456
```

Python

```
a = 123
b = 456

# Concatenated using comma
```

```
print ("The current value of a is", a)
print ("The current value of a is", a, "and b is", b)

#Concatenated using +
print ("The current value of a is " + a)
print ("The current value of a is " + a + " and b is " + b)
```

```
# Output
The current value of a is 123
The current value of a is 123 and b is 456
The current value of a is 123
The current value of a is 123 and b is 456
```

String

String Concatination

Java

```
public class StrContatination{
    public static void main(String[] args){
        String a = "Welcome to ";
        String b = "CSE330 Lab.";
        System.out.println(a + b);
    }
}
```

```
# Output
Welcome to CSE330 Lab.
```

Python

```
a = "Welcome to "
b = "CSE330 Lab."
z = a + b
print (z)
```

```
# Output
Welcome to CSE330 Lab.
```

String Indexing

Java

```
//Accessing String with character position number. Space is a
lso a character
public class StrIndexing{
    public static void main(String[] args){
        String a = "Welcome to ";
        String b = "CSE330 Lab.";
        System.out.println(a.charAt(0));
        System.out.println(b.charAt(5));
    }
}
```

```
# Output
W
0
```

Python

```
a = "Welcome to "
b = "CSE330 Lab."
```

```
print (a[0])
print (b[5])
```

```
# Output
W
0
```

Length of the String

Java

```
public class StrLength{
    public static void main(String[] args) {
        String a = "Welcome to ";
        String b = "CSE330 Lab.";
        String z = a + b;

        System.out.println(z.length());
    }
}
```

```
#Output
22
```

Python

```
a = "Welcome to "
b = "CSE330 Lab."
z = a + b
print(len(z))
```

```
#Output
22
```

Acsessing Last Character

Java

```
public class LastChar{
    public static void main(String[] args) {

        String a = "Welcome to ";
        String b = "CSE330 Lab.";

        System.out.println(b.charAt(b.length() - 1));
    }
}
```

```
#Output
.
```

Python

```
a = "Welcome to "
b = "CSE330 Lab."
print(b[len(b)-1])
print(b[-1])
```

```
#Output
.
.
```

String Slicing

Java

```
public class StrSlice {
    public static void main(String[] args) {

        String s = "Welcome to CSE330 Lab.";

        // using builtin function
        // s[0:5] → substring starting from 0 to 5 (excludin
g)
        System.out.println(s.substring(0, 5));

        // without using any builtin function
        for(int i=0; i<5; i++){
            System.out.print(s.charAt(i));
        }
    }
}
```

```
#Output
Welco
Welco
```

Python

```
s = "Welcome to CSE330 Lab."
print(s[0:5])   # prints characters from index 0 to 4
```

```
#Output
Welco
Welco
```

Java

```
public class StrSlice {
    public static void main(String[] args) {

        String s = "Welcome to CSE330 Lab.";

        // s[:6] → substring starting from 0 to 6 (excludin
g).
        System.out.println(s.substring(0, 6));

        // s[6:] → substring starting from 6 to End.
        System.out.println(s.substring(6));

        // s[-4:] → last 4 characters
        System.out.println(s.substring(s.length() - 4));

        //=========================================

        // without using builtin functions
        for(int i=0; i<6; i++){
            System.out.print(s.charAt(i));
        }
        System.out.println();
        for(int i=6; i<s.length(); i++){
            System.out.print(s.charAt(i));
        }
        System.out.println();
        for(int i=s.length()-4; i<s.length(); i++){
            System.out.print(s.charAt(i));
        }
    }
}
```

```
#Output
Welcom
e to CSE330 Lab.
Lab.
#================
Welcom
e to CSE330 Lab.
Lab.
```

Python

```
s = "Welcome to CSE330 Lab."

print (s[:6])
print (s[6:])
print (s[-4:])
```

```
#Output
Welcom
e to CSE330 Lab.
Lab.
```

User Input

Java

```
import java.util.Scanner;

public class UserInput {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter username: ");
        String username = sc.nextLine();
```

```
        System.out.println("Username is: " + username);
        //====================================
        System.out.println(username + username + username);
        //====================================
        System.out.print("Enter number: ");
        int num = input.nextInt();
        System.out.println(num * 2);
    }
}
```

```
#Output
Enter username:coder
Username is: coder
#====================================
codercodercoder
#====================================
Enter number:4
8
```

Python

```
username = input("Enter username:")
print("Username is: " + username)
#====================================
print(username*3)
#====================================
num = int(input("Enter number:"))
print (num * 2)
```

```
#Output
Enter username:coder
Username is: coder
#====================================
codercodercoder
```

```
#====================================
Enter number:4
8
```

Conditional Statement

Java

```
public class CondStat {
    public static void main(String[] args) {
        int x = 5;
        int y = 6;

        if (x > y) {
            System.out.println("Maximum is x");
        } else {
            System.out.println("Maximum is y");
        }
    }
}
```

```
#Output
Maximum is y
```

Python

```
x = 5
y = 6

if x > y:
  print("Maximum is x")
else:
  print ("Maximum is y")
```

```
#Output
Maximum is y
```

elif

Java

```
// elif
public class CondStat {
    public static void main(String[] args) {
        int x = 5;
        int y = 5;

        if (x > y) {
            System.out.println("Maximum is x");
        } else if (x == y) {
            System.out.println("x and y are equal");
        } else {
            System.out.println("Maximum is y");
        }
    }
}
```

```
#Output
x and y are equal
```

Python

```
#elif
x = 5
y = 5

if x > y:
```

```
  print("Maximum is x")
elif x == y:
  print ("x and y are equal")
else:
  print ("Maximum is y")
```

```
#Output
x and y are equal
```

and

- Both conditions need to be True

Java

```
public class CondStat {
    public static void main(String[] args) {
        int a = 200;
        int b = 10;
        int c = 500;

        if (a > b && c > a) {
            System.out.println("Both conditions are True");
        } else {
            System.out.println("Negative!!");
        }
    }
}
```

```
#Output
Both conditions are True
```

Python

```
a = 200
b = 10
c = 500
if a > b and c > a:
  print("Both conditions are True")
else:
  print("Negative!!")
```

```
#Output
Both conditions are True
```

or

- Either one of the conditions = True should be enough

Java

```
public class CondStat {
    public static void main(String[] args) {
        int a = 200;
        int b = 10;
        int c = 500;

        if (a > b || a > c) {
            System.out.println("At least one of the condition
s is True");
        } else {
            System.out.println("Negative!!");
        }
    }
}
```

```
#Output
At least one of the conditions is True
```

Python

```
a = 200
b = 10
c = 500
if a > b or a > c:
  print("At least one of the conditions is True")
else:
  print("Negative!!")
```

```
#Output
At least one of the conditions is True
```

Nested if

Java

```
public class CondStat {
    public static void main(String[] args) {
        int x = 50;

        if (x > 10) {
            System.out.println("Above ten!");

            if (x > 20) {
                System.out.println("and also above 20!");
            } else {
                System.out.println("but not above 20!");
            }
```

```
        } else {
            System.out.println("Not even above ten!");
        }
    }
}
```

```
#Output
Above ten!
and also above 20!
```

Python

```
x = 50

if x > 10:
  print("Above ten!")
  if x > 20:
    print("and also above 20!")
  else:
    print("but not above 20!")
else:
  print("Not even above ten!")
```

```
#Output
Above ten!
and also above 20!
```

pass

Java

```
public class PassStat{
    public static void main(String[] args) {
```

```
        int a = 10;
        int b = 20;

        if (b > a) {
            // do nothing (equivalent to pass)
        } else {
            System.out.println("hello");
        }
    }
}
```

Python

```
a = 10
b = 20

if b > a:
  pass
#else:
#print("hello")
```

Python Built-in Data Type

List

- Similar to Java or C array, but not same
- List can store any type of data in one single list.
- It doesn’t have any fixed size like array.
- Requires more memory to store additional information about data types of each element

Python

```
#Initializing
list = [1, 2, 3, 4, 5]
print(list)
```

```
#Output
[1, 2, 3, 4, 5]
```

```
mixed_list = [1, 2.5, "hello", [1, 2, 3]]
print(mixed_list[3])
```

```
#Output
[1, 2, 3]
```

```
#List Methods
print(len(list))
print(4 in list)    #Search

list[2] = "New"     #Update value
print(list)

list.append("World")    #Add new element
print(list)
```

```
#output
5
True
[1, 2, 'New', 4, 5]
[1, 2, 'New', 4, 5, 'World']
```

```
list.remove("New")    #Remove element
print(list)
```

```
list.pop(1)           #Remove using index
print(list)

list2 = [16, 55]
joined_list = list + list2
print(joined_list)
```

```
#Output
[1, 2, 4, 5, 'World']
[1, 4, 5, 'World']
[1, 4, 5, 'World', 16, 55]
```

List Slicing

```
my_list = ['a', 'b', 'c', 'e', 'f', 'd']

print (my_list[3])
print (my_list[1:3])
print (my_list[:3])
print (my_list[3:])
print (my_list[-3:])
```

```
#Output
e
['b', 'c']
['a', 'b', 'c']
['e', 'f', 'd']
['e', 'f', 'd']
```

```
print (my_list[0:4:2])      #index+2 everytime
print (my_list[:])          #All elements
```

```
print (my_list[::2])
print (my_list[::-1])       #Reverse order
```

```
#Output
['a', 'c']
['a', 'b', 'c', 'e', 'f', 'd']
['a', 'c', 'f']
['d', 'f', 'e', 'c', 'b', 'a']
```

```
#Print last 4 elements in reverse
my_list = ['a', 'b', 'c', 'd', 'e', 'f']

print (my_list[-1:-5:-1])

#print 2nd to 5th element reverse order
print (my_list[4:0:-1])
```

```
#Output
['f', 'e', 'd', 'c']
['e', 'd', 'c', 'b']
```

```
#Matrix [List of list] using list
matrix = [[1, 2, 3, 4],
          [5, 6, 7, 8]]

# |1 2 3 4|
# |5 6 7 8|

print (matrix[1][2])
```

```
#Output
7
```

Tuple

- Immutable (unchangable) - Items can not be modified
- Collection of elements of multiple data types
- Commonly used in programming (like Python) and databases to store a fixed sequence of related data

Python

```
tuple = (1, 2, 3, 4, 5)
tuple[0]
#====================
tuple[1] = "hello"
```

```
#Output
1
#======================
TypeError                                 Traceback (most rec
ent call last)
<ipython-input-41-f3de6cd25f9c> in <cell line: 0>()
----> 1 tuple[1] = "hello"

TypeError: 'tuple' object does not support item assignment
```

Set

- Immutable - Items can not be modified
- Unordered - No index
- No duplicates allowed

Python

```
set = {"apple", "banana", "cherry", "banana"}
print(set)
```

```
#====================
set.add("orange")
print(set)

set.remove("banana")
print(set)
```

```
#Output
{'apple', 'cherry', 'banana'}
#====================
{'apple', 'cherry', 'orange', 'banana'}
{'apple', 'cherry', 'orange'}
```

Dictionary

- Values stored as key : value pair
- Unordered, mutable
- Key must be unique and immutable types (e.g., strings, numbers, tuples).

Python

```
drivers = {
    16: "Charles",
    1: "Max",
    55: "Carlos",
    "Extra": "Liam"
}

print(drivers)
```

```
#Output

```

```
{16: 'Charles', 1: 'Max', 55: 'Carlos', 'Extra': 'Liam'}
```

```
print(drivers["Extra"])      #Accessing elements

drivers["Extra"] = "Ollie"
print(drivers)

drivers.update({81:"Oscar"})
print(drivers)

drivers.pop("Extra")
print(drivers)
```

```
#Output
Liam
{16: 'Charles', 1: 'Max', 55: 'Carlos', 'Extra': 'Ollie'}
{16: 'Charles', 1: 'Max', 55: 'Carlos', 'Extra': 'Ollie', 81:
'Oscar'}
{16: 'Charles', 1: 'Max', 55: 'Carlos', 81: 'Oscar'}
```

Loops

While loop

Java

```
public class WhileLoop{
    public static void main(String[] args) {
        int i = 0;

        while (i < 10) {
            System.out.println(i);
            i = i + 1;
        }
```

```
    }
}
```

Python

```
i = 0
while i < 10:
    print(i)
    i = i + 1
```

```
#Output
0
1
2
3
4
5
6
7
8
9
```

For loop

Java

```
public class ForLoop{
    public static void main(String[] args) {
        for (int i = 0; i < 10; i += 2) {
            System.out.println(i);
        }
    }
}
```

Python

```
for i in range(0, 10, 2):
    print(i)
```

```
#Output
0
2
4
6
8
```

Iterate over the list

Java

```
//Iterate over the list
public class ArrayIteration{
    public static void main(String[] args){
        String[] list = {"a","b","c","d","e"};
        for(int i = 0; i < list.length; i++){
            System.out.println(list[i]);
        }
        System.out.println("------------------");
        for(int i = 0; i < list.length; i++){
            System.out.println(i);
        }
        System.out.println("------------------");
        for(int i = 0; i < list.length; i++){
            System.out.println("Index = "+i+", Value = "+list
[i]);
        }
```

```
    }
}
```

```
#Output
a
b
c
d
e
------------------
0
1
2
3
4
------------------
Index = 0, Value = a
Index = 1, Value = b
Index = 2, Value = c
Index = 3, Value = d
Index = 4, Value = e
```

Python

```
#Iterate over the list
list = ['a', 'b', 'c', 'd', 'e']

for i in list:
  print (i)
print("------------------")
for i in range(len(list)):
  print(i)
print("------------------")
```

```
for i in range(0, len(list)):
    print("Index =", i, "Value =", list[i])
```

```
# OUTPUT
a
b
c
d
e
------------------
0
1
2
3
4
------------------
Index = 0, Value = a
Index = 1, Value = b
Index = 2, Value = c
Index = 3, Value = d
Index = 4, Value = e
```

Break Statement

Java

- Break statement - Out of the loop

```
public class BreakStatement{
    public static void main(String[] args){
        String[] fruits = {"apple", "banana", "cherry"};
        for(int i = 0; i < fruits.length; i++){
            System.out.println(fruits[i]);
            if(fruits[i].equals("banana")){
                break;
```

```
            }
        }
    }
}
```

```
# OUTPUT
apple
banana
```

Python

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)
  if x == "banana":
    break
```

```
# OUTPUT
# OUTPUT
apple
banana
```

Continue Statement

Java

- Continue statement - Stop the current iteration of the loop and continue with the next

```
public class ContinueStatement{
    public static void main(String[] args){
        String[] fruits = {"apple", "banana", "cherry"};
        for(int i = 0; i < fruits.length; i++){
            if(fruits[i].equals("banana")){
```

```
                continue;
            }
            System.out.println(fruits[i]);
        }
    }
}
```

```
# OUTPUT
apple
cherry
```

Python

```
fruits = ["apple", "banana", "cherry"]
for i in fruits:
  if i == "banana":
    continue
  print(i)
```

```
# OUTPUT
apple
cherry
```

Nested Loop

Java

```
public class NestedLoop{
    public static void main(String[] args){
        int[] num_arr = {1,2,3};
        String[] alphabet_arr = {"a","b","c"};
        for(int i = 0; i < num_arr.length; i++){
            System.out.println(num_arr[i]);
```

```
            for(int j = 0; j < alphabet_arr.length; j++){
                System.out.println(alphabet_arr[j]);
            }
        }
    }
}
```

```
# OUTPUT
1
a
b
c
2
a
b
c
3
a
b
c
```

Python

```
#Nested loop
num_list = [1 , 2, 3]
alphabet_list = ['a', 'b', 'c']

for number in num_list:
    print(number)
    for letter in alphabet_list:
        print(letter)
```

```
# OUTPUT
1
```

```
a
b
c
2
a
b
c
3
a
b
c
```

Functions or Methods

Java

- We cannot create a java method with default value but we can use method overloading (same method name but different numbers of parameters).

```
public class JavaMethod{
    public static void hello_method(){
        System.out.println("Hello");
    }
    public static void name_method(String fname, String lnam
e){
        System.out.println(fname+" "+lname);
    }
    public static int sum_numbers(int num1, int num2){
        return num1 + num2;
    }
    public static int increment(int number) {
    return increment(number, 1); // default value
    }
    public static int increment(int number, int by) {
    return number + 2 * by;
```

```
    }
    public static void main(String[] args){
        hello_method();
        name_method("Cristiano", "Ronaldo");
        int x = sum_numbers(2, 3);
        System.out.println(x);
        System.out.println(increment(2,5));
        System.out.println(increment(2));
    }
}
```

```
# OUTPUT
Hello
Cristiano Ronaldo
5
4
12
```

Python

```
def hello_function():
  print("Hello")

def name_function(fname, lname):
  print(fname, " ", lname)

def sum_numbers(number1, number2):
  result = number1+number2
  return result

def increment(number, by=1):
    return number + 2*by

hello_function()
```

```
name_function("Cristiano", "Ronaldo")
x = sum_numbers(2, 3)
print(x)
print(increment(2))
print(increment(2, 5))
# Keyword arguments
print(increment(by=1, number=2))
```

NumPy

Java

```
import java.util.Arrays;
public class JavaArr{
    public static void main(String[] args){
        String[] arr = {"1", "2", "3", "2.45", "hello"};
        System.out.println(Arrays.toString(arr));
        System.out.println("Arr Length: " + arr.length);
    }
}
```

```
# OUTPUT
[1, 2, 3, 2.45, hello]
Arr Length: 5
```

Python

- A Python library
- Used for working with arrays
- Short form of "Numerical Python”
- Aims to provide an array object that is up to 50x faster than traditional Python lists
- Provides a lot of supporting functions

```
# As arrays can store only one type of data, therefore it con
verts everything to String
# We cannot perform any kind of calculations
import numpy as np

arr = np.array([1 , 2, 3, 2.45, "hello"])
print(arr)
print(type(arr))
print("Arr Length:",len(arr))
print(arr.shape)
```

```
# OUTPUT
['1' '2' '3' '2.45' 'hello']
<class 'numpy.ndarray'>
5
(5,)
```

Append

Java

```
import java.util.Arrays;
public class JavaAppend{
    public static void main(String[] args){
        // We need to create another array (y), where the siz
e is greater than the previous
        // array (x). Then copy all values from x to y and fi
nally insert the new value (70)
        int[] x = {1,2,3,4,5};
        int[] y = new int[x.length+1];
        for(int i = 0; i < x.length; i++){
            y[i] = x[i];
        }
```

```
        y[y.length-1] = 70;
        System.out.println(Arrays.toString(y));
        System.out.println(Arrays.toString(x));
    }
}
```

```
# OUTPUT
[1,2,3,4,5,70]
[1,2,3,4,5]
```

Python

```
import numpy as np

x = np.array([1, 2, 3, 4, 5])
y = np.append(x, [70])
print(y)
print(x)    #No change in main array
```

```
[ 1  2  3  4  5 70]
[1 2 3 4 5]
```

Delete

Java

```
import java.util.Arrays;

public class JavaDelete {
    public static void main(String[] args) {
        int[] x = {1, 2, 3, 4, 5};
        // We need to create another array (y), where the siz
e is less than the previous
```

```
                // array (x). Then copy all values from x to
y while checking the value's index you want to remove and the
n insert the other values other than the removed value.
        int deleteValue = 3;   // value we want to remove

        int[] y = new int[x.length - 1];

        int index = 0;
        for (int i = 0; i < x.length; i++) {
            if (x[i] != deleteValue) {
                y[index] = x[i];
                index++;
            }
        }

        System.out.println(Arrays.toString(y));
        System.out.println(Arrays.toString(x));
    }
}
```

```
#Output
[1, 2, 4, 5]
[1, 2, 3, 4, 5]
```

Python

```
x = np.array([1, 2, 3, 4, 5])
y = np.delete(x, 3)   #Delete using index
print(x)
print(y)
```

```
#Output
[1, 2, 4, 5]
```

```
[1, 2, 3, 4, 5]
```

Sorting

Java

```
import java.util.Arrays;

public class SortArrayAscending {
    public static void main(String[] args) {
        int[] numbers = {5, 2, 8, 1, 9};

        Arrays.sort(numbers);  //Ascending

        //Descending - Reverse the Ascending order sorted arr
ay
        int[] revArr = new int[numbers.length];
        int s = 0;
        for(int i=numbers.length-1; i>=0; i--){
            revArr[s++]=numbers[i];
        }

        System.out.println(Arrays.toString(numbers));
        System.out.println(Arrays.toString(revArr));
    }
}
```

```
#Output
[1, 2, 5, 8, 9]
[9, 8, 5, 2, 1]
```

Python

```
x = np.array([7, 8, 3, 10, 5])
sorted_x = np.sort(x)
print(sorted_x)     #Ascending
print(sorted_x[::-1]) #Descending - Reverse the Ascending ord
er sorted array
```

```
#Output
[1, 2, 5, 8, 9]
[9, 8, 5, 2, 1]
```

2-D Array

Java

```
import java.util.Arrays;

public class Java2DArrayExample {
    public static void main(String[] args) {

        int[][] arr_2d = {
            {1, 2, 3, 7, 55},
            {4, 5, 6, 99, 10}
        };

        System.out.println(Arrays.deepToString(arr_2d));
    }
}
```

```
#Output
[[1, 2, 3, 7, 55], [4, 5, 6, 99, 10]]
```

Python

```
arr_2d = np.array([[1, 2, 3, 7, 55],
                   [4, 5, 6, 99, 10]])
print(arr_2d)
```

```
#Output
[[ 1  2  3  7 55]
 [ 4  5  6 99 10]]
```

Shape & Dimention of a Matrix

Java

```
public class Java2DArrayInfo {
    public static void main(String[] args) {

        int[][] arr_2d = {
            {1, 2, 3, 4, 5},
            {6, 7, 8, 9, 10}
        };

        int rows = arr_2d.length;
        System.out.println("Number of rows: " + rows);

        int cols = arr_2d[0].length;
        System.out.println("Shape: (" + rows + ", " + cols +
")");

        int ndim = 2;  // all 2D arrays in Java are 2-dimensi
onal
        System.out.println("Number of dimensions: " + ndim);
    }
}
```

```
#Output
Number of rows: 2
Shape: (2, 5)
Number of dimensions: 2
```

Python

```
print(len(arr_2d))
print(arr_2d.shape)     #row, column
print(arr_2d.ndim)      #Dimension
```

```
#Output
2
(2, 5)
2
```

Indexing

Java

```
//1st & last element of an array
import java.util.Arrays;

public class SortArrayAscending {
    public static void main(String[] args) {
        int[] numbers = {5, 2, 8, 1, 9};


        System.out.println(numbers[0]);
        System.out.println(numbers[numbers.length-1]);
    }
}
```

```
#Output
5
9
```

Python

```
arr = np.array([5, 2, 8, 1, 9])

print(arr[0])
print(arr[-1])
```

```
#Output
5
9
```

Java

```
import java.util.Arrays;

public class Java2DArrayAccess {
    public static void main(String[] args) {

        int[][] arr = {
            {1, 2, 3, 4, 5},
            {6, 7, 8, 9, 10}
        };

        System.out.println(Arrays.deepToString(arr));

        System.out.println("4th element on 2nd row: " + arr
[1][3]);
    }
}
```

```
[[1, 2, 3, 4, 5], [6, 7, 8, 9, 10]]
4th element on 2nd row: 9
```

Python

```
arr = np.array([[1,2,3,4,5],
                [6,7,8,9,10]])
print(arr)

print('4th element on 2nd row: ', arr[1, 3])
```

```
#Output
[[ 1  2  3  4  5]
 [ 6  7  8  9 10]]
4th element on 2nd row:  9
```

Slicing 2-D Array

Java

```
import java.util.Arrays;

public class Java2DSlicing {
    public static void main(String[] args) {

        int[][] arr = {
            {1, 2, 3, 4, 5},
            {6, 7, 8, 9, 10}
        };
                //From the second row, slice elements from in
dex 1 to index 4 (not included)
        int[] slice1 = Arrays.copyOfRange(arr[1], 1, 4);
        System.out.println(Arrays.toString(slice1));
```

```
                //From both rows, slice index 1 to index 4 (n
ot included)
        int[][] slice2 = new int[2][3];

        for (int i = 0; i < 2; i++) {
            slice2[i] = Arrays.copyOfRange(arr[i], 1, 4);
        }

        System.out.println(Arrays.deepToString(slice2));
    }
}
```

```
#Output
[7, 8, 9]
[[2, 3, 4], [7, 8, 9]]
```

Python

```
arr = np.array([[1, 2, 3, 4, 5],
                [6, 7, 8, 9, 10]])

#From the second row, slice elements from index 1 to index 4
(not included)
print(arr[1, 1:4])

#From both rows, slice index 1 to index 4 (not included)
print(arr[0:2, 1:4])
```

```
#Output
[7 8 9]
[[2 3 4]
 [7 8 9]]
```

Taking last column and row

Java

```
import java.util.Arrays;

public class JavaColumnRowLoop {
    public static void main(String[] args) {

        int[][] arr = {
            {1, 2, 3, 4, 5},
            {6, 7, 8, 9, 10}
        };

        int[] column0 = new int[arr.length];
        for (int i = 0; i < arr.length; i++) {
            column0[i] = arr[i][0];
        }
        System.out.println(Arrays.toString(column0));

        int[] row0 = new int[arr[0].length];
        for (int j = 0; j < arr[0].length; j++) {
            row0[j] = arr[0][j];
        }
        System.out.println(Arrays.toString(row0));
    }
}
```

```
#Output
[1 6]
[1 2 3 4 5]
```

Python

```
print(arr[:, 0])    #column
print(arr[0, :])    #row
```

```
#Output
[1 6]
[1 2 3 4 5]
```

Zero Matrix

Java

```
import java.util.Arrays;

public class JavaZeros {
    public static void main(String[] args) {

        // Create a 4×2 2D array filled with zeros
        int[][] y = new int[4][2];

        System.out.println(Arrays.deepToString(y));
    }
}
```

```
#Output
[[0, 0], [0, 0], [0, 0], [0, 0]]
```

Python

```
#zero matrix
y = np.zeros((4,2))     #row, column
print(y)
```

```
#Output
[[0. 0.]
 [0. 0.]
 [0. 0.]
 [0. 0.]]
```

Summation & Product of all elements

Java

```
import java.util.Arrays;

public class JavaArraySumProd {
    public static void main(String[] args) {

        int[] a = {3, 4, 5};

        int sumA = 0;
        int prodA = 1;
        for (int i = 0; i < a.length; i++) {
            sumA += a[i];
            prodA *= a[i];
        }
        System.out.println("Sum of a: " + sumA);
        System.out.println("Product of a: " + prodA);
     }
}
```

```
#Output
Sum of a: 12
Product of a: 60
```

Python

```
a = np.array([3, 4, 5])

print(np.sum(a))
print(np.prod(a))
```

```
#Output
12
60
```

Column & Row Wise Operation

Java

```
import java.util.Arrays;

public class JavaArraySumProd {
    public static void main(String[] args) {

        int[][] b = {
            {1, 2, 3},
            {4, 5, 6}
        };
        int sumB = 0;
        for (int i = 0; i < b.length; i++) {
            for (int j = 0; j < b[i].length; j++) {
                sumB += b[i][j];
            }
        }
        System.out.println("Sum of b: " + sumB);

        int[] colSum = new int[b[0].length];
        for (int j = 0; j < b[0].length; j++) {
            for (int i = 0; i < b.length; i++) {
```

```
                colSum[j] += b[i][j];
            }
        }
        System.out.println("Column-wise sum: " + Arrays.toStr
ing(colSum));

        int[] rowSum = new int[b.length];
        for (int i = 0; i < b.length; i++) {
            for (int j = 0; j < b[i].length; j++) {
                rowSum[i] += b[i][j];
            }
        }
        System.out.println("Row-wise sum: " + Arrays.toString
(rowSum));
    }
}
```

```
#Output
Sum of b: 21
Column-wise sum: [5, 7, 9]
Row-wise sum: [6, 15]
```

Python

```
b = np.array([[1,2,3],
              [4,5,6]])

print(np.sum(b))
print(np.sum(b, axis=0))
print(np.sum(b, axis=1))
```

```
#Output
21
```

```
[5 7 9]
[ 6 15]
```

Iterating Row by Row

Java

```
public class Java2DArrayLoopOnly {
    public static void main(String[] args) {

        int[][] arr = {
            {1, 2, 3},
            {4, 5, 6}
        };

        for (int i = 0; i < arr.length; i++) {
            for (int j = 0; j < arr[i].length; j++) {
                System.out.print(arr[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

```
#Output
1 2 3
4 5 6
```

Python

```
arr = np.array([[1, 2, 3],
                [4, 5, 6]])
```

```
for x in arr:
  print(x)
```

```
#Output
[1 2 3]
[4 5 6]
```

Print Element by Element

Java

```
public class Java2DArrayNestedLoop {
    public static void main(String[] args) {

        int[][] arr = {
            {1, 2, 3},
            {4, 5, 6}
        };
        for (int i = 0; i < arr.length; i++) {
            for (int j = 0; j < arr[i].length; j++) {
                System.out.println(arr[i][j]);
            }
        }
    }
}
```

```
#Output
1
2
3
4
5
6
```

Python

```
arr = np.array([[1, 2, 3], [4, 5, 6]])
for x in arr:
  for y in x:
    print(y)
```

```
#Output
1
2
3
4
5
6
```

Linspace

- Create a random array, same difference between the numbers

Java

```
public class JavaLinspace {
    public static void main(String[] args) {

        double start = 0;
        double end = 100;
        int n = 4;

        double[] x = new double[n];
        double step = (end - start) / (n - 1);

        for (int i = 0; i < n; i++) {
            x[i] = start + i * step;
        }
```

```
        for (int i = 0; i < n; i++) {
            System.out.print(x[i] + " ");
        }
    }
}
```

```
#Output
0.0 33.333333333333336 66.66666666666667 100.0
```

Python

```
x = np.linspace(0, 100, 4)     #start, end, how many numbers
print(x)
```

```
#Output
[  0.          33.33333333  66.66666667 100.        ]
```