Working with Arrays

- An array is memory allocation which enable you to store homogeneous mixer of elements and share the variable name
- In general an ordinary variable can store a value. For ordinary variable memory allocation are different in places thus reading the data from ordinary variable is little slow
- For Array variable memory allocations consequent locations thus reading the data from the array variable is faster than an ordinary variable
- In Java Array are Classified into 3 types
  - Single dimension Array [SDA - 1DArray]
  - Double dimension Array [DDS - 2D Array]
  - Jagged Array [JDA – 3D Array]
- In Java For Array Variable memory is allocated in the Heap
- For Every Array Variable an extra block is created by JVM i.e length is always holding the size of an Array
- As we know that array index values will be starts from 0 to n-1
- While working with Array
  - We need to declare the Array
    - Specifying the type of Array you wish to work with
  - We need to define the Array
    - Specifying no.of.elements required to strore

Single Dimension Array [SDA]:

- It is memory allocation which enable you to store the data in a liner fashion

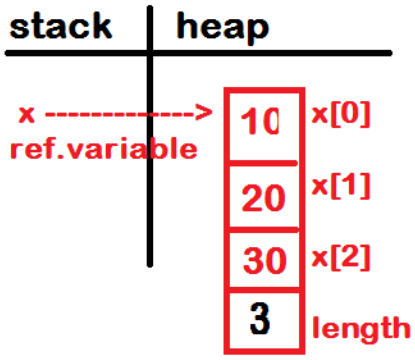
Syn For SDA :

```
[Modifiers] <datatype> <vname>[ ];
            int x[ ]; or int [ ]x; or int[ ] x;
```

Syn: <vname> = new <datatype>[size];

Eg: `x = new int[3];`

`int[ ] x = new int[3];`



Assigning values to an Array:

```
int[ ] x={10,20,30};
int[ ] x=new int[3];
       x[0]=10; x[1]=20; x[2]=30;
```

Printing Values:

```
S.out.println(x[0]); //10
S.out.println(x[1]); //20
S.out.println(x[2]); //30
S.out.println(x.length); //3
```

App-2:

```
for(int i=0; i<3; i++)
    { S.o.pln(x[i]); }
```

App-3

```
for(int i=0; i<x.length;i++)
    { S.o.pln(x[i]); }
```

Example :

```
//SDADemo.java
class SDADemo
{
    public static void main(String args[])
    {
            int[ ] x=new int[3];
            x[0]=10; x[1]=20; x[2]=30;

            //App-1
            System.out.println(x[0]); //10
            System.out.println(x[1]); //20
            System.out.println(x[2]); //30
            System.out.println(x.length); //3

            //App-2
            System.out.println("====================");
            for(int i=0; i<3; i++)
              { System.out.println(x[i]); }

            //App-3
            System.out.println("====================");
                    for(int i=0; i<x.length; i++)
                    {  System.out.println(x[i]); }
    }
}
```

Enhanced For Loop:

- It is specially designed to read the values From Collections
- It will read all the values from the Collection from the beginning to till end
- It doesn't allow to read the values From specified position

```
Syn: for(<dtype> <variable> :Collection )
        {  statement(s);  }

     int[ ] a={1,2,3,4,5};    a-->[1 | 2 | 3 | 4 | 5 ]

     for(int  i : a)
       { S.o.pln(i); } //1 | 2 | 3 | 4 | 5
```

```
for(int i:a)
  S.o.pln(i);
```

```
int[ ] a={1,2,3,4,5};
for(int i=0; i<a.length;i++)
{ S.o.pln(a[i]); }

int s=0;
for(int i=0; i<a.length;i++)
    { s=s+a[i]; }
S.o.pln("Sum is : "+s); //15
```

```
for(int i:a)
  { S.o.pln(i); }

int s=0;
for(int i:a)
{ s=s+i; }
S.o.pln("Sum is : "+s);
```


Example

```
class EFLoop
{
    public static void main(String args[ ])
    {
        int[ ] a={10,20,30,40,50};
        for(int i:a)
        {System.out.println(i); }

        int s=0;
```

```
            for(int j:a)
            { s=s+j; }
            System.out.println("Sum is : "+s);
        }
}
```

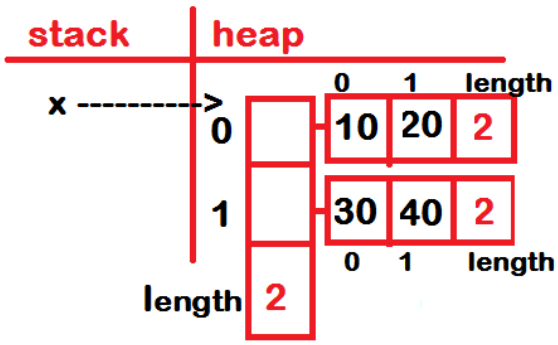
Double dimension Arrays

- It a memory allocation which enable you to store the data in the combination of both rows and columns . simply in the form of matrix

DDA

```
Syn: [modifiers] <datatype>  <vname>[ ][ ];
Eg:   int x[ ][ ]; or int  [ ][ ]x;  or  int[ ][ ]  x; or int[ ]  [ ]x;

Syn:  <vname> = new <datatype>[size][size];
Eg:                x= new int[2][2];
       int[ ][ ]  x = new int[2][2];
```




```
Assign values :
    int[ ][ ] x=new int[2][2];
    x[0][0]=10;  x[0][1]=20;
    x[1][0]=30;  x[1][1]=40;

Reading Values From Array:
S.o.pln(x[0][0]); //10
S.o.pln(x[0][1]); //20
S.o.pln(x[1][0]); //30
S.o.pln(x[1][1]); //40
```

```
for(int r=0; r<x.length; r++)
{
 for(int c=0; c<x[0].length; c++)
   {
       S.o.pln(" "+x[r][c]);
   }
 S.o.pln(" ");
}
```

Example:

```
//DDADemo.java
class DDADemo
{
    public static void main(String args[ ])
    {
            int[ ][ ] x=new int[2][2];

            //Code for assigning values to an Array
            x[0][0]=10;  x[0][1]=20;
            x[1][0]=30;  x[1][1]=40;

            //Reading Values From Array
            System.out.println("No.of.Rows : "+x.length);
            System.out.println
                    ("No.of.Cols in each row : "+x[0].length);

            for(int r=0; r<x.length; r++)
            {
          for(int c=0; c<x[0].length; c++)
                    {
                        System.out.print(" "+x[r][c]);
                    }
                    System.out.println(" ");
            }
    }
}
```

Jagged Arrays:

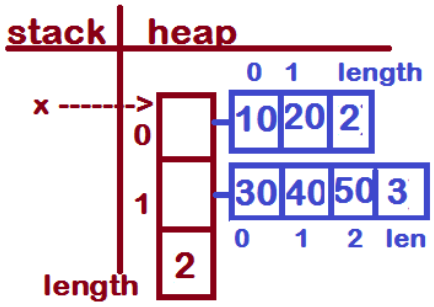
- **It will work similar with double dimension Array. But the difference is in double dimension Array No.of.Columns in each row is same, where as in Jagged Arrays No.of.Columns are not Same**
- **<u>In Simple Words Jagged Array is an Array of Array's</u>**

Syn For JDA:

```
[modifier] <datatype>  <vname>[ ][ ];
int x[ ][ ];  or int   [ ][ ]x;  or int[  ]  [ ]x;   or int[ ][ ]  x;
```

Defining JDA:

```
Syn: <vname> = new <datatype>[size][ ];
  Eg:            x = new int[2][ ];
```




```
x[0] = new int[2];
x[0][0]=10;  x[0][1]=20;
x[1] = new int[3];
x[1][0]=30; x[1][1]=40;
x[1][2]=50;
```

```
for(int r=0; r<x.length;r++)
{
  for(int c=0; c<x[r].length  ;c++)
   {
        S.o.p(" "+x[r][c]);
   }
  S.o.pln(" ");
}
```

Example:

```
//JDADemo.java
class JDADemo
{
	public static void main(String args[ ])
	{
		int[ ][ ] x=new int[2][ ];

		x[0]=new int[2];
		x[0][0]=10; x[0][1]=20;

		x[1]=new int[3];
		x[1][0]=40; x[1][1]=50; x[1][2]=60;

		System.out.println("No.of.Rows : "+x.length);
		System.out.println
			("No.of.cols in x[0]th row : "+x[0].length);
		System.out.println
			("No.of.cols in x[1]th row : "+x[1].length);


		//Reading the values From the JDArray
		for(int r=0; r<x.length; r++)
		{
			for(int c=0; c<x[r].length; c++)
			{
				System.out.print(" "+x[r][c]);
			}
			System.out.println(" ");
		}
	}
}
```