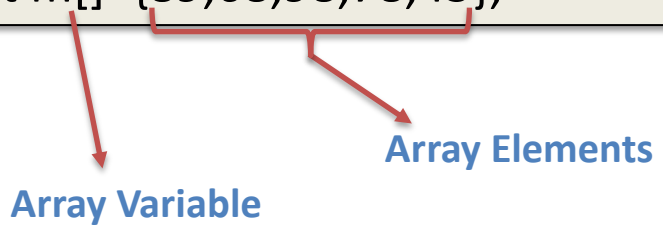
Array

- **Array** is the collection of similar type of data with contiguous memory allocation.

Without Array

int m1=89;
int m2=68;
int m3=98;
int m4=78;
int m5=45;

With Array
int m[]={89,68,98,78,45};



Steps for Array Creation

Step 1- Array Declaration

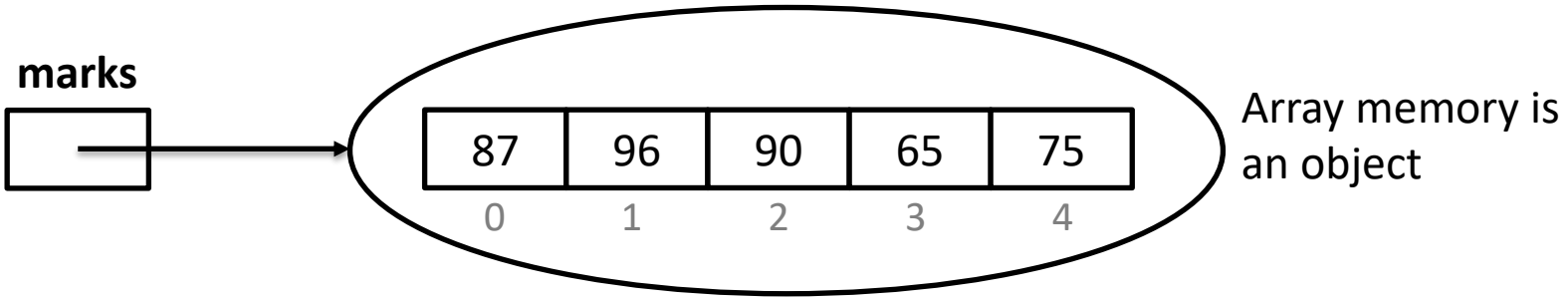
int marks[]; or **int []marks;**

Step 2- Array Memory Creation

marks=new int[5];

Step 3- Array Initialization:

```
marks[0]=87;
marks[1]=96;
marks[2]=90;
marks[3]=65;
marks[4]=75;
```



Note:

Array indexing starts from 0. Because by default array variable refers the first block of the memory. That's why to access first block, we need to add 0 to current address and to access second block, we need to add 1.

Array Creations

incapp

Array Creation:

```
int marks[]= new int[5];
marks[0]=87;
marks[1]=96;
marks[2]=90;
marks[3]=65;
marks[4]=75;
```

Array Creation:

```
int marks[]= new int[ ]{87,96,90,65,75};
```

Array Creation:

```
int marks[]= new int[5]{87,96,90,65,75};
```

Array Creation:

```
int marks[]= {87,96,90,65,75};
```

Array Creation:

```
int marks[];
marks={87,96,90,65,75};
```

Accessing an Array

We can access array using two loop controls:

1. Normal for loop
2. for-each loop

Normal for loop

```
int marks[]={89,68,98,78,45};
for(int i=0;i<marks.length;i++) {
        System.out.println(marks[i]);
}
```

For-Each loop

```
int marks[]={89,68,98,78,45};
for(int x:marks) {
        System.out.println(x);
}
```

Note:
'marks.length' property return the number of elements in array.