Arrays

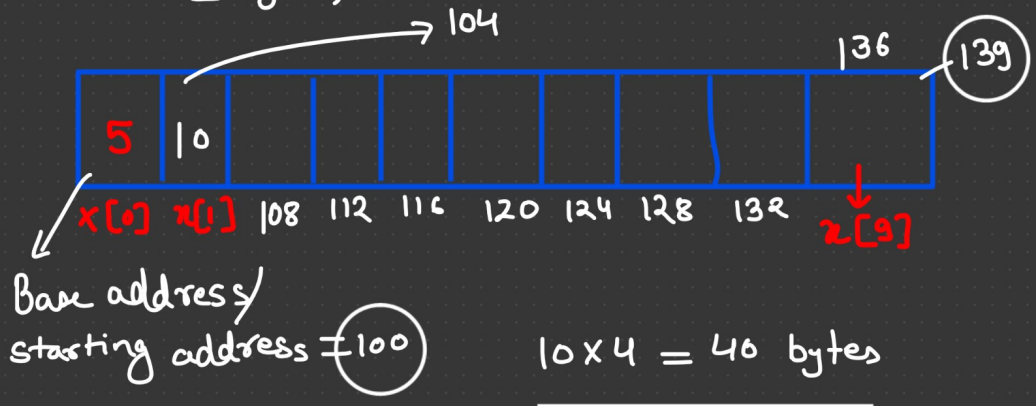
1) How to create an Array?

Data Type variable name [size];

Ex:- int x[10];

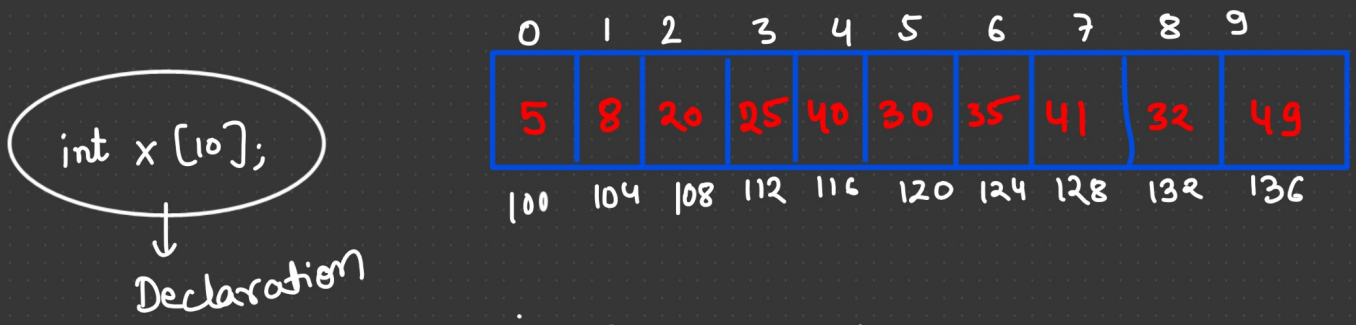
x[0] = 5;
x[1] = 10;
⋮

int = 4 byte



10 × 4 = 40 bytes

* Array is collection of similar type of Elements, stored in Contigous memory allocation.



X = Base address = 100
X[0] = Value at 100. = 5

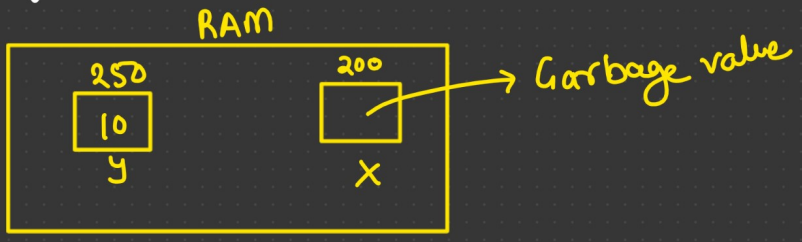
X+1 = 104
X[1] = value at 104 = 8

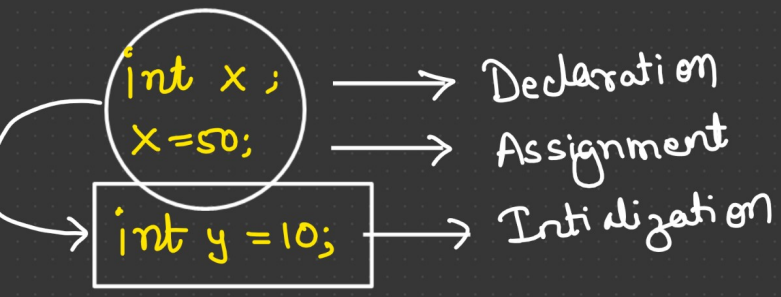
X+6 = 124
X[6] = value at 124 = 35

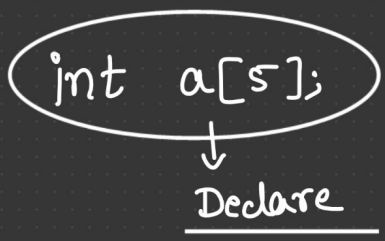
x[0] = x + 0X4 = 100 + 0 = 100
x[2] = 100 + 2X4 = 108
x[6] = 100 + 6X4 = 124

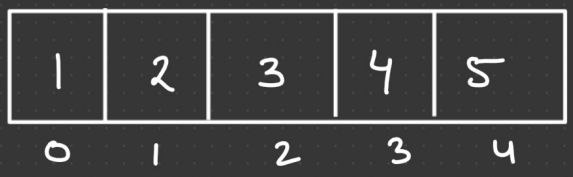
Address = Base Address + (index x size)

Intilialization of an Array:-










```
int a[5] = {1,2,3,4,5};  // Intialization
int a[5] = {1,2,3};      // 1, 2, 3, 0, 0
int a[ ] = {1,2,3,4,5};  //size = 5.
```

---

2) Accessing Array Elements:-

```
for(i=0 ;i<n ;i++)
{
    printf(" %d", a[i]);

}
```

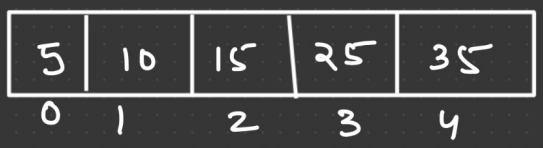
i = 0 , a[0] = 5

i = 1 , a[1] = 10

i = 2 , a[2] = 15

i = 3 , a[3] = 25

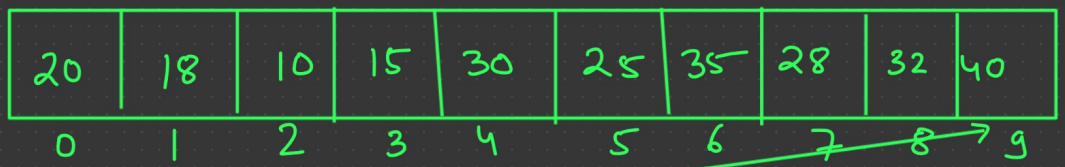
i = 4 , a[4] = 35

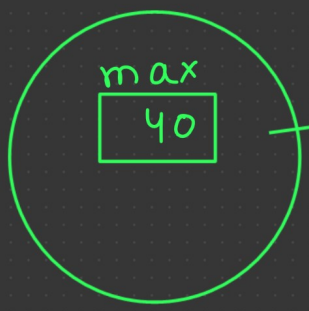


Q:- WAP to find maximum element from the array.

↓

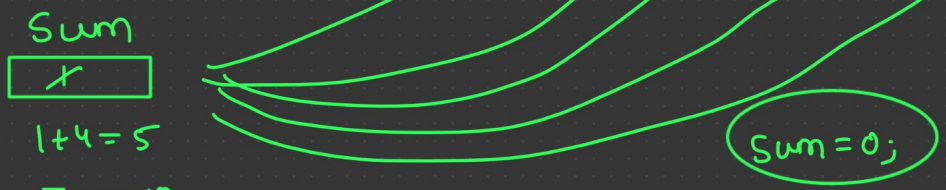
Write a program






```
max = a[0];
for (i= 1 ; i< 10 ; i++)
{
    if (max < a[i])
        max = a[i];
}
printf(" Maximum is %d", max);
```

Q:- Find the sum of all the elements in the given array.



1+4 = 5

5+5 = 10

10+3 = 13

13+4 = 17

```
for(i=0 ; i<5 ; i++)
{
    sum = sum + a[i];
}
```

Q:- WAP to sort the given array in increasing order

{ 2, 6, 4, 9, 8 } ⟶ { 2, 4, 6, 8, 9 }

a[0]

4	6	8	10	12	9	14
0	1	2	3	4	5	6

swap(10,6);	swap(10,6);
swap(6,4);	swap(10,8);

i=0 [1, 2, 3, 4, 5, 6]

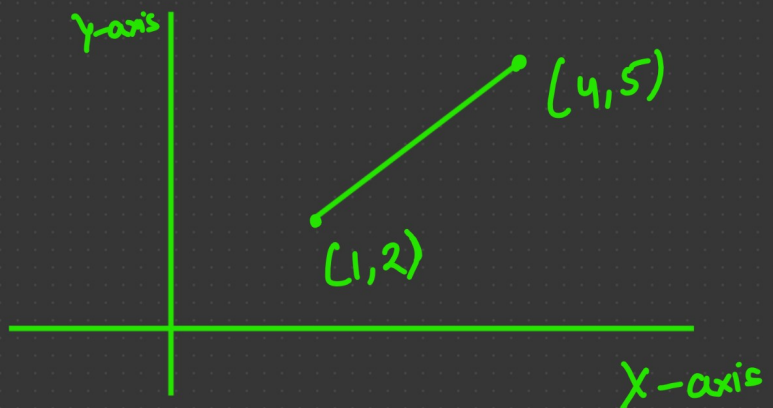
i=1 [2, 3, 4, 5, 6]

i=2 [3, 4, 5, 6]

```
for(i=0 ; i<6 ; i++)
{
   for(j=i+1 ; j<7 ; j++)
   {
      if(a[i] > a[j])
         swap(a[i], a[j]);
   }
}
```

2D Array :-

row	0	1	2	3 (Columns)
→ 0	00	01	02	03
→ 1	10	11	12	13
→ 2	20	21	22	23
→ 3	30	31	32	33



Matrix = 4 x 4 = 16 cells

(Row, Column)

int a[4][4];

Row (0-3)

Column (0-3)

```
int a[5][4];
```

→ 5 rows

→ 4 columns

(0-4) (0-3)

	0	1	2	3
0	0	1	2	3
1	1	2	3	4
2	2	3	4	5
3	3	4	5	6
4	4	5	6	7

← i

↑ j

```
a[0][0] = 1;
a[0][2] = 4;
a[2][2] = 6;
```

```
for(i=0; i<=4; i++)
{
    for(j=0; j<=3; j++)
    {
        a[i][j] = i+j;
    }
}
```

i = 0 1

j = 0 1 2 3

Memory Representation of 2D Array -

```
int a[4][4];
```

	0	1	2	3	
100 → 0	1	2	3	4	→ 16 byte
116 → 1	5	6	7	8	→ 16 byte
132 → 2	9	10	11	12	→ 16 byte
3	13	14	15	16	

4×4

Row 0 | Row 1 | Row 2 | Row 3

100	104	108	112	116	120	124	128	132	136	140	144				
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3

$a[1][2] = 100 + [(4 \times 4) \times 1] + (4 \times 2) = 124$

$a[2][3] = 100 + [(4 \times 4) \times 2] + (4 \times 3) = 144$

(no. of column) Sizeof(int)

int a[R][C];

a[i][j]

$$\text{Address} = \text{Base Address} + [C \times (\text{sizeof Data Type}) \times i] + (\text{sizeof DT}) \times j$$