CS & IT ENGINEERING

Programming in C

Arrays and Pointers

Lec- 01

By- Pankaj Sharma sir

Arrays and Pointers-1

Arrays & Address

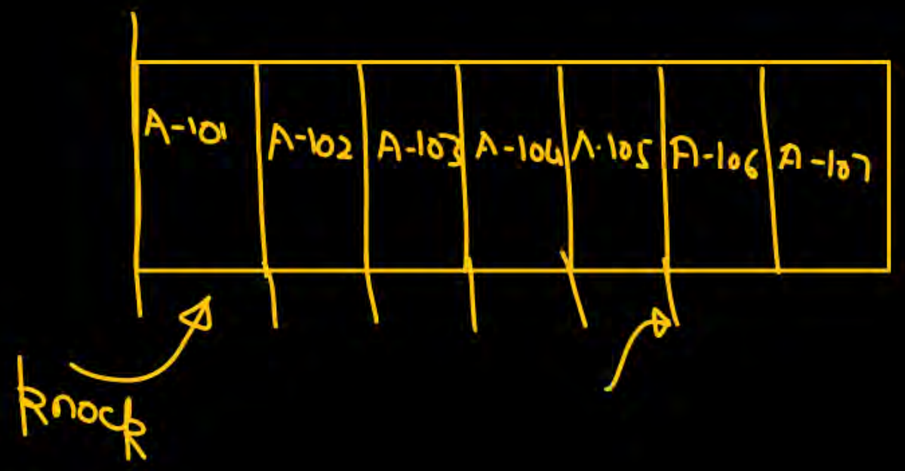
1) Address

Relative Address ⇒ purpose of introducing Array in C lang.

Abs. Address

A-106, Krishna Nagar

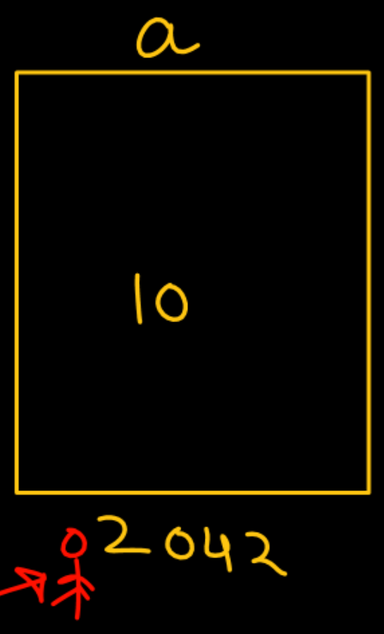
Mathura (U.P.)



2) Address of operator (&)

```
int a = 10;
```

&a = Memory location 2042



3) value at operator (*) :

int a = 10;

&a = Memory location 2042

*( )

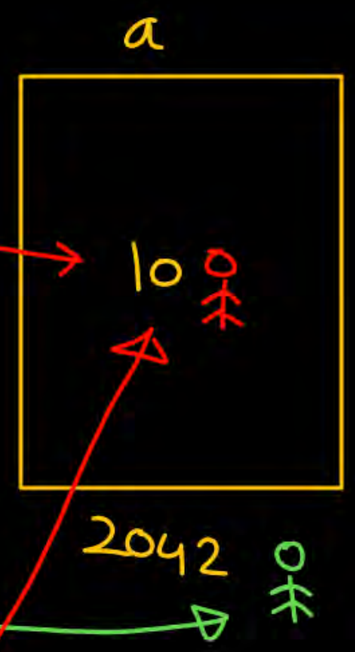
Address

*(&a) = value at (Memory location 2042) = 10

int a = 10;

a ⇒ 10

&a ⇒ 2042



*(&a) ⇒ valueat(2042)

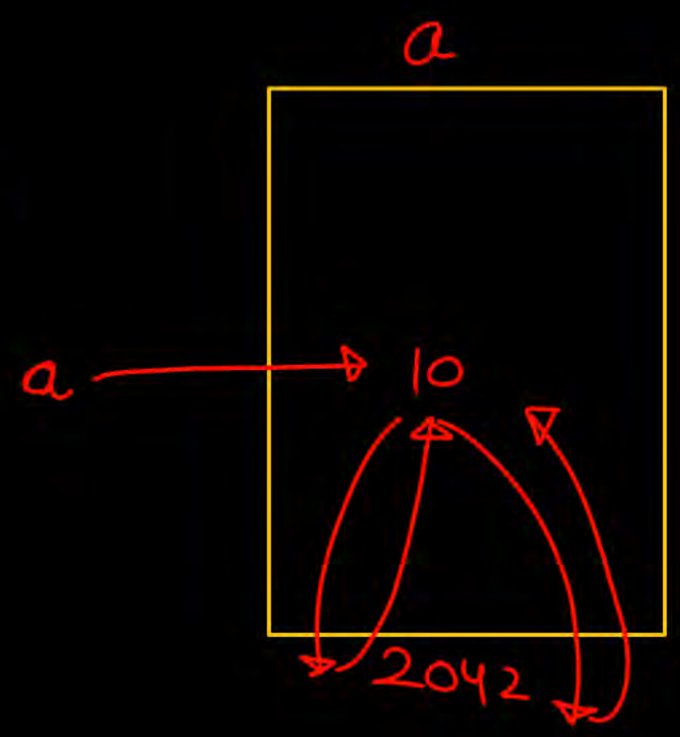
= 10

*&a = a

*&a = a

```
int a = 10;
printf("%d", *&*&*&a);
```

10



same type

```
int m1, m2, m3;
float avg;
```

```
scanf("%d %d %d", &m1, &m2, &m3)
```

Why array?

50 stu

```
int m1, m2, m3, m4, m5, m6,
    m7, m8, m9, m10, m11, m12,
    m13, - - - -
```

int a, b, c;

3 variables

of type

of type

elements

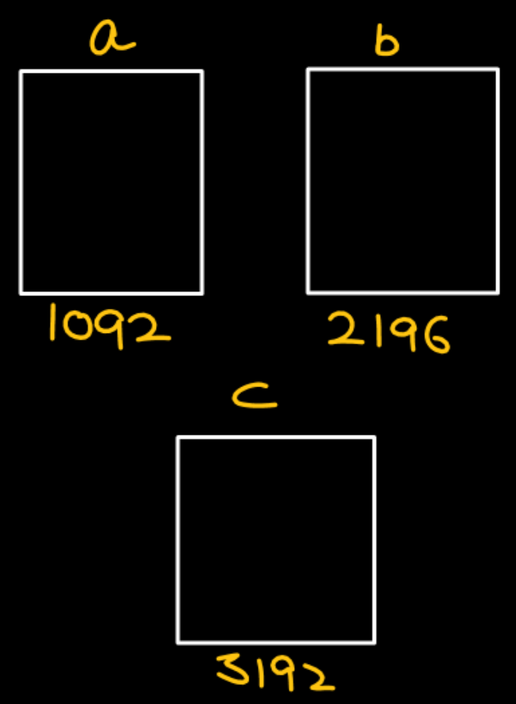
int a [3]

a is a group of

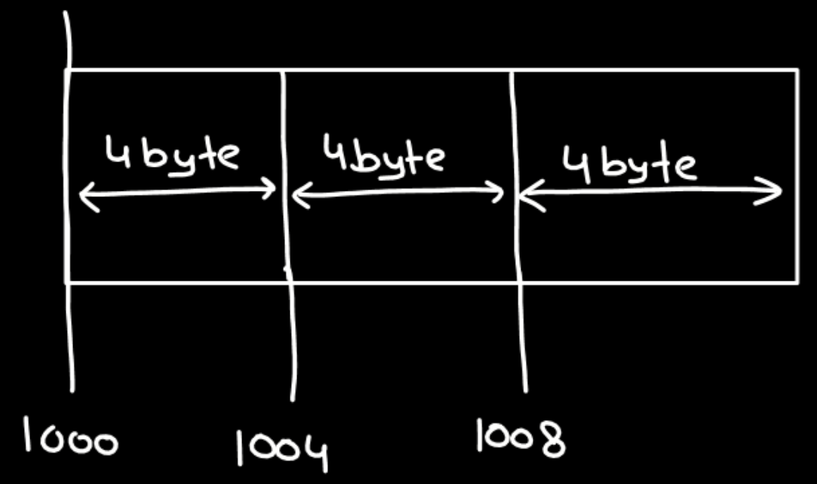
array is collection of (group of) similar type of elements.

int → 4 bytes

int a, b, c;



int a[3];



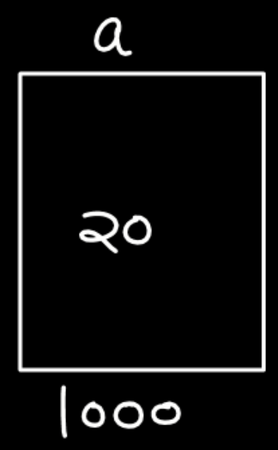
Seq. store

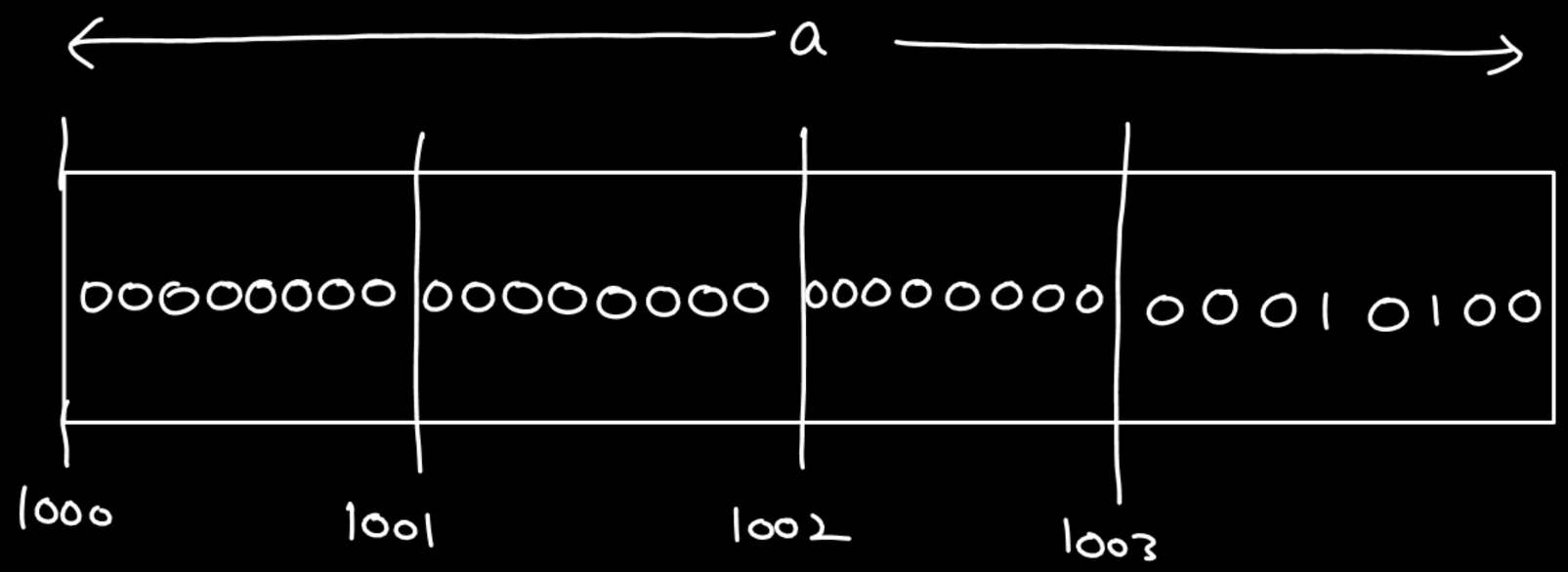
elements are stored one after another

int a = 20;

4 byte

2BHK 5BHK





```
int a,b,c;
```

=

```
a = 3;
```

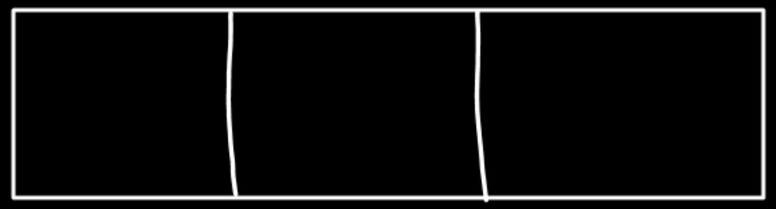
=

```
b = 10;
```

=

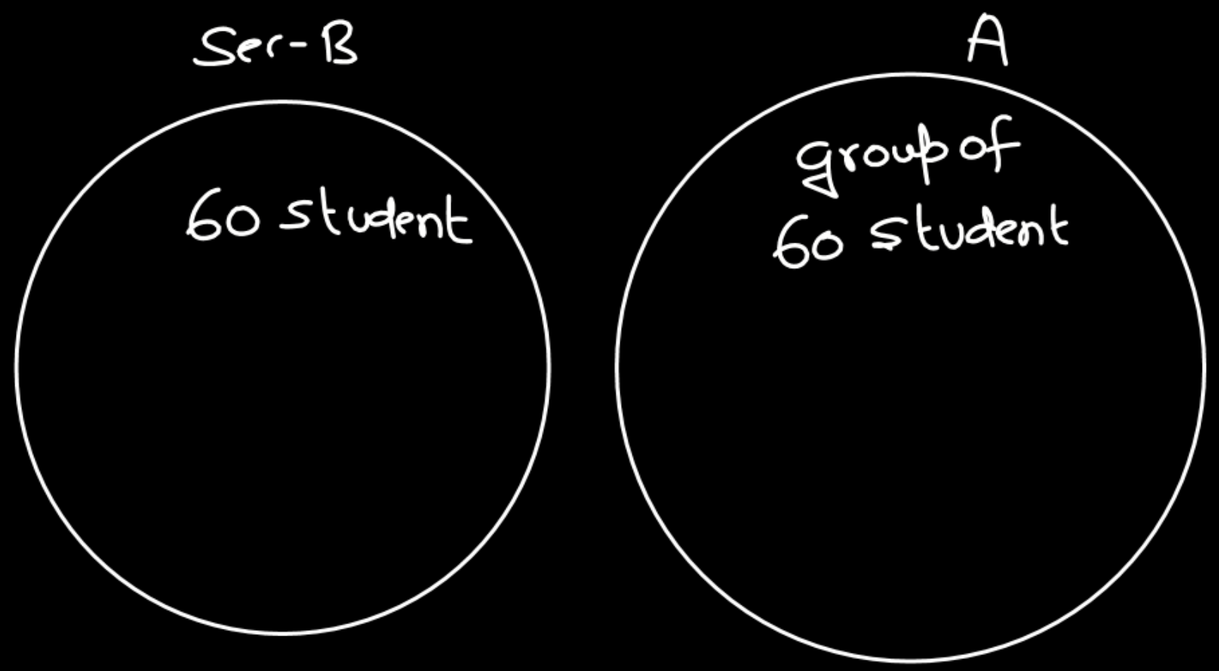
```
c = 100;
```

```
int a[3];
```



All 3 elements are represented by same entity/name a.

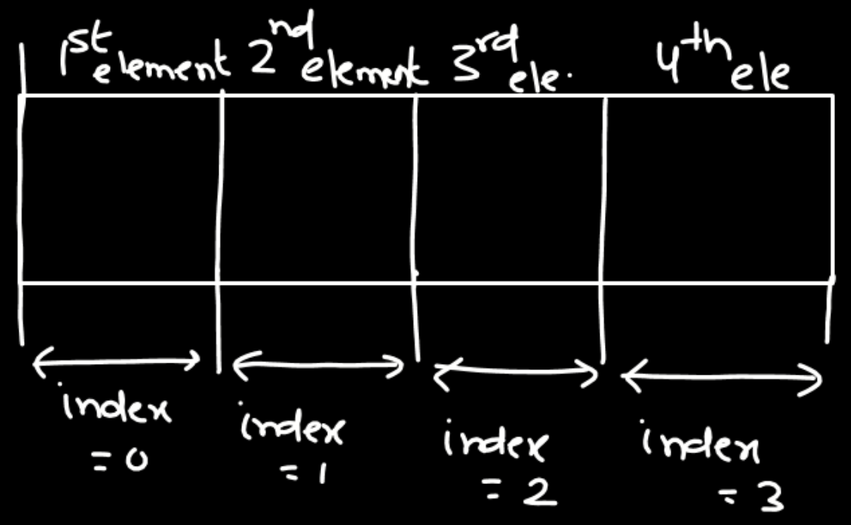
Group name → unique id.



Unique identification number → Roll no.

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

a


1st element
2nd element
3rd ele.
4th ele
index = 0
index = 1
index = 2
index = 3

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

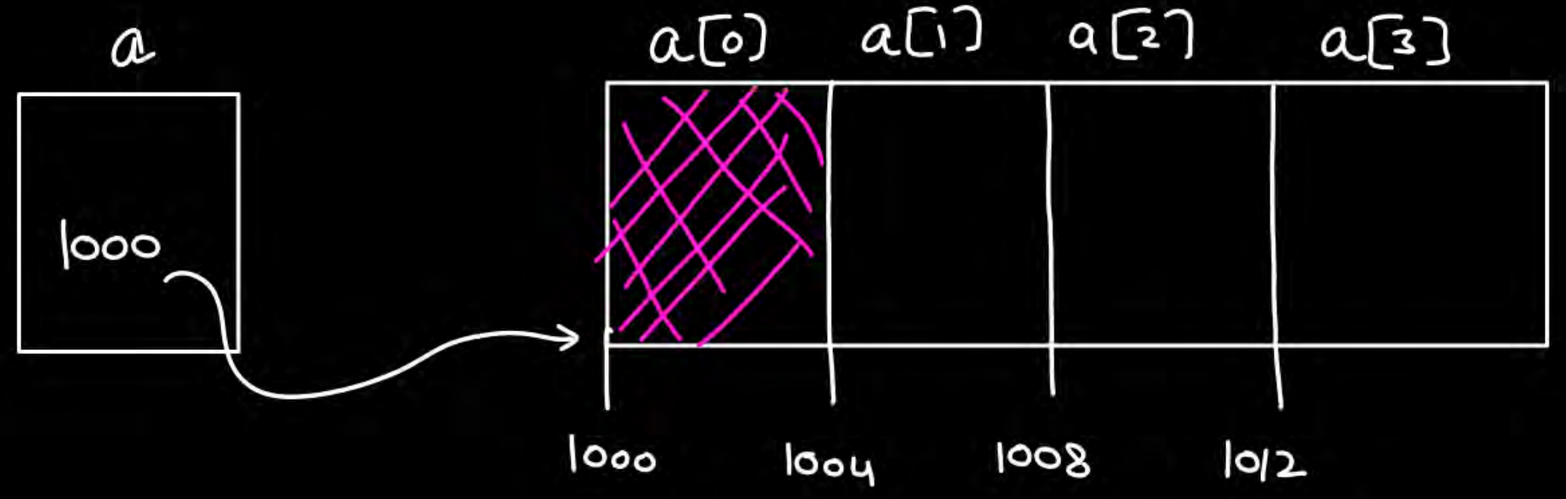
a

a[0]	a[1]	a[2]	a[3]
3		100	
0	1	2	3

```
a[0] = 3;
=
a[2] = 100;
```

3rd element

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



Array-name represent address of its first element.

} constant address

a is as same as &a[0]

Lvalue = Rvalue

- Constant/literal
- var
- expression

Lvalue → can not be a constant

Array-name is a constant

Array-name =  ;

① Array-name can not be Lvalue for assignment statement

(2)

```
++2;
--2;
2--;
2++;
```

All are invalid.

we can not apply inc/dec operators on constant/literals.

```
Array-name++;
++Array-name;
--Array-name;
Array-name--;
```

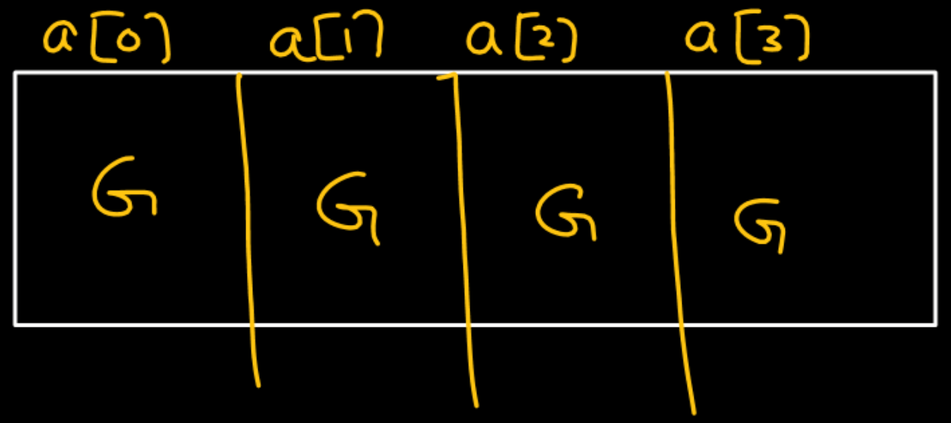
All are invalid

```
void main()
{
 int a ;
 printf("%d",a);
}
```

Garbage


```
void main(){
 int a[4] ;
 printf("%d",a[0]);
}
```

Garbage



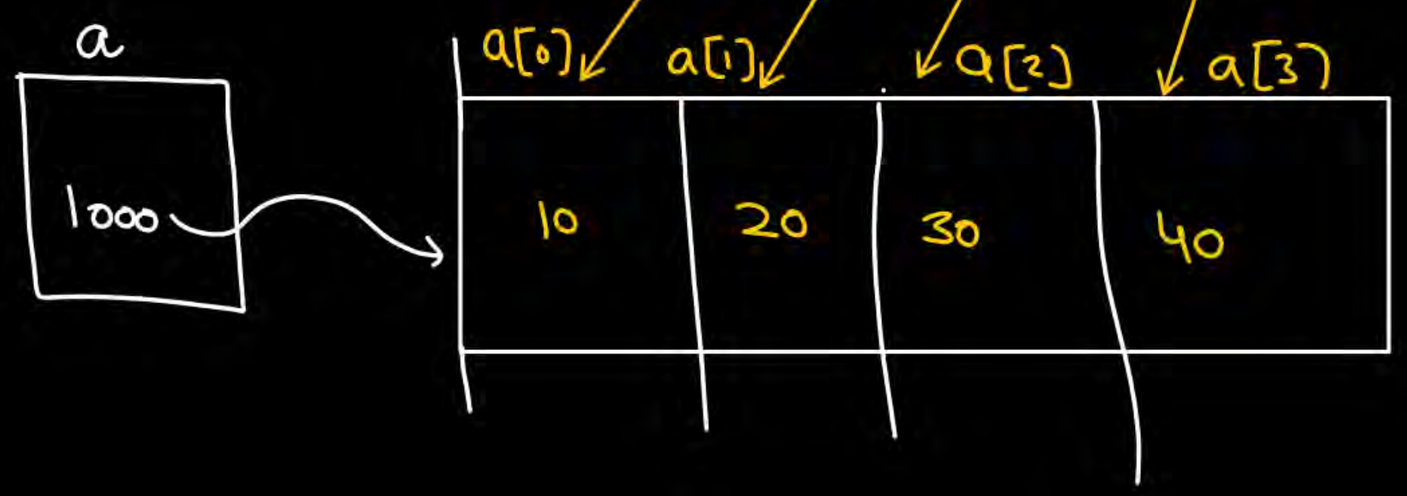
int a = 1; valid

lvariable lvalue

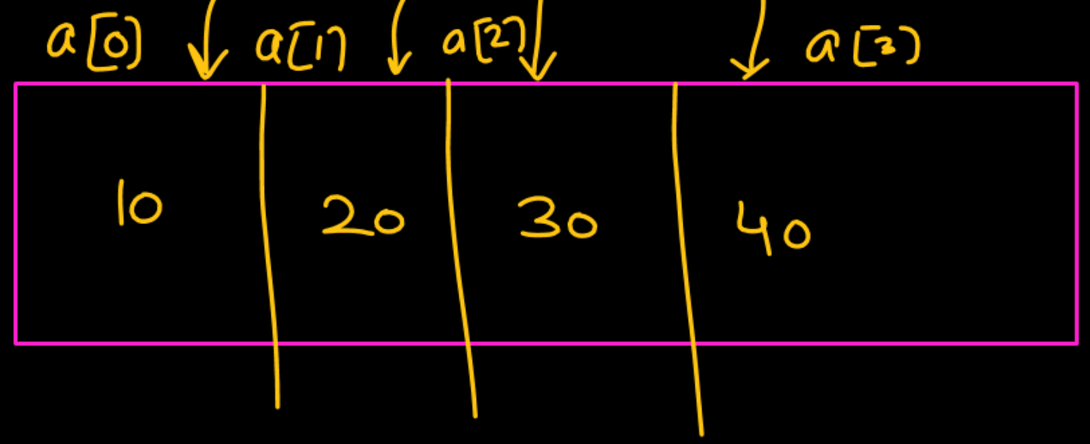
int a[4] = { 10, 20, 30, 40 };

Collection of multiple variable

a[0], a[1], a[2], a[3]



```
int a[4] = {10, 20, 30, 40}
```



```
int a[4] = {10, 20};
```

a[0]	a[1]	a[2]	a[3]
10	20	0	0

Online Compiler

int a[3] = 4; ⇒ run

a[0]

unsigned int

logically

① int a[ ]; Compiler Ud ke baat Marega

Vidyut

Tiger

② int x = {20}; ✓

int x = 20;

③ int a[ ] = {10, 20, 30}; valid ✓ ✓ ✓

3 size ✓

④ int a[size]

index 0 to size-1

5.)

```
#define SIZE 8
int a[2];
int a[2+3];
int a[2x3+2];
int a[SIZE];
int a[-2];
int a[0];
```

Illegal

String args

int a[unsigned int];

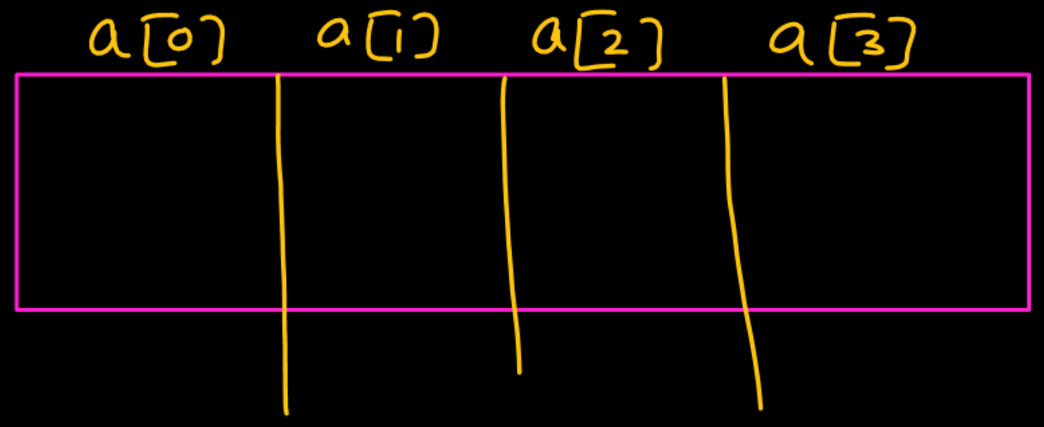
declaration

नोट पूरी

C → malloc, calloc, realloc, free

0 3

int a[4];



a[4] = 100; // behaviour is undefined

```
void main(){
    int a;
    a=10;
    =
}
```

```
void main(){
    int a[4];
    a = {1,2,3,4};
    =
}
```

Invalid

Lvalue

Error

```
void main(){
    int a;
    int a;
}
```

invalid

9 PM

9:00 – 10:30

NIC


THANK YOU GW
SOLDIERS !