

Pointer to Array

Array of Pointers

Syntax:-

`int *arr[5]`

- * arr is an array containing 5 pointers to int
- * stores 5 addresses.
- * Each pointer points to different integer / element

Example of array of pointers:-

`int *arr[5];` →

100	1008	200	204	308
0th	1st	2nd	3rd	4th

 8B 8x5 = 40B

`int a = 10; int b = 20;`
`int c = 30; int d = 40;`
`int e = 50;`

a	10
b	20
c	30
d	40
e	50

100 1008 200 204 308
0th 1st 2nd 3rd 4th

⇒ 4 byte x 5 = 20

① `arr[0] = &a;`
`arr[1] = &b;`
`arr[2] = &c;`
`arr[3] = &d;`
`arr[4] = &e;`

② `int *arr[5] = {&a, &b, &c, &d, &e};`

Access all variables value via pointer Array

```
for (int i = 0; i < 5; i++) // 0-4
{
    printf("%d", *arr[i]); // arr[0] = *100 → 10
}

10 20 30 40 50
```

//

60B

1x5 saved 32B

// use array instead of using 5 variables

`int arr[5] = {10, 20, 30, 40, 50};`
 `arr` →

10	14	18	22	26
10	20	30	40	50

 ⇒ 4x5 = 20B

`int *arr_ptr[5] = {arr[0], arr[1], arr[2], arr[3], arr[4]};`

```
for (int i = 0; i < 5; i++)
{
    printf("%d", *arr_ptr[i]);
}
```

Output 10 20 30 40 50

Pointer to Array

`int (*arr)[5]`

- * arr is a pointer to an array of 5 integers
- * store on address (the address of an entire array)

`int array[5] = {10, 20, 30, 40, 50};`

`int (*ptr)[5] = &array;`

10	20	30	40	50
100	104	108	112	116

 = 20

`ptr`

100

 8 bytes

Access all array elements using pointer to array.

```
for (int i = 0; i < 5; i++) // 0-4
{
    printf("%d", (*ptr)[i]);
```

}

output ⇒ 10 20 30 40 50

this will give address of array i.e 100

total = 28 bytes
28B ..