

1 byte = 8 bit so 32 byte = 4 bit

## Pointer & array :- Difference

1) `int arr[10];`  
`sizeof(arr);` → 40 byte  
(`sizeof` pure array ka size return karta)  
- tot space taken by array

1) `int *p = arr;`  
`sizeof(p) = 8`

- tot space taken by pointer

2) `arr = arr + 1` ✗

eg. 

100	1	2	3
-----	---	---	---

1) `p = p + 1` ✓

$$\begin{aligned} \text{arr} &= 100 + 3 \times 4 \\ &= 112 \end{aligned}$$

## Char :-

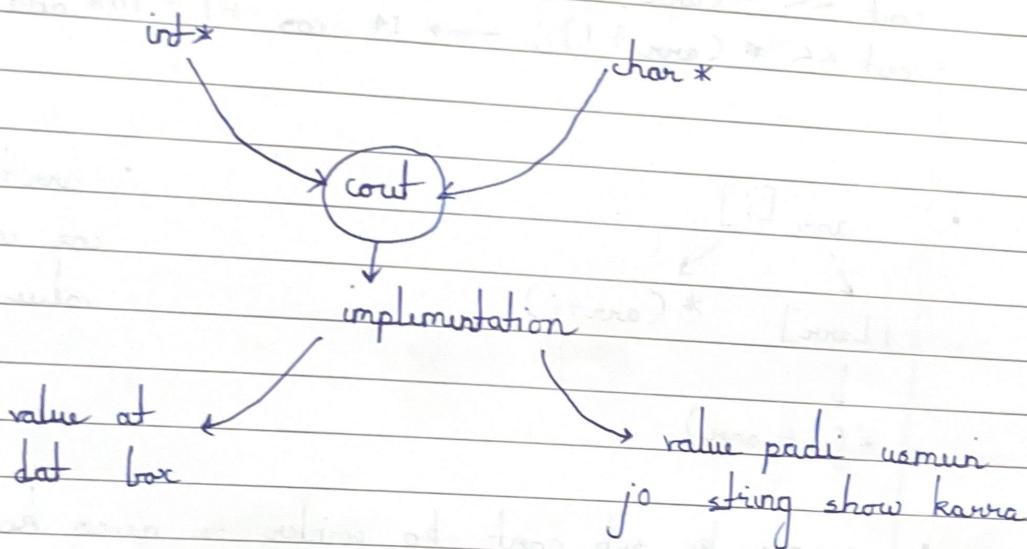
`char ch[10] = "Babbar";`

`char *ch = ch;`

- `cout << c;`

Babbar

cout khr diff for char



• `*c = *(c+0)`  
`= c[0]`

Bad pract

coz temp store kr kab tak



so 32 byte = 4 bit

111

## - Difference

1)  $\text{int} * p = \text{arr}$   
 $\text{sizeof}(p) = 8$

- tot space taken by pointer

1)  $p = p + 1$  ✓

3 x 4

cout like diff  
for char

Badhan

char \*

value padi usmein  
string show karva

eg.  $\text{char name}[10] = \text{"Shur Bano"};$   
 $\text{char} * \text{cptr} = \&\text{name}[0];$

1)  $\text{name} \rightarrow \text{Shur Bano}$

2)  $\text{cptr} + 2 \rightarrow \text{ur Bano}$  ✓ 2 as last index

2)  $\&\text{name} \rightarrow 104$

8)  $*\text{cptr} \rightarrow \text{cptr}[0] \rightarrow \text{S}$

3)  $(\text{name} + 3) \rightarrow \text{name}[3] \rightarrow \text{r}$

9)  $\text{cptr} + 8 \rightarrow \text{empty}$

4)  $\text{cptr} \rightarrow \text{Shur Bano}$

5)  $\&\text{cptr} \rightarrow 216$

6)  $*(\text{cptr} + 3) \rightarrow \text{cptr}[3] \rightarrow \text{r}$

• faruga

$\text{char ch} = 'k';$

~~cout~~  $\text{char} * c = \&\text{ch};$

$\text{cout} << c;$

→ OP = k garbage value

coz null single I/P mein add nahi hota for  
termination

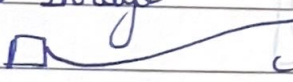
•  $\text{char name}[10] = \text{"Sneha"}; \rightarrow \text{Sneha}$

working:- 1) temp storage  $\rightarrow \text{"Sneha"}$

2) memory change  $\rightarrow$  copy to name array ki storage

$\text{char} * c = \text{"Sneha"}; \rightarrow \text{Sneha}$

working:- 1) temp storage  $\rightarrow \text{"Sneha"}$

2)  $c$    $c$  pointer points to S first letter

Bad pract

coz temp storage

dk kab tak nahuga



## Pointers with function :-

- when we pass array in function so pointer pass hota so pass by reference

func :- solve ( int arr [] {  
arr[0] = 50;  
↓  
\*(arr+0)  
∴ actual arr mein chg hota

- pointer pass hota , pointer pass hone se if chg anything using dot pointer so actual arr chg so pass by ref.

eg. 1

main ( )

int arr [10] = {1, 2, 3, 4};

// print = 1, 2, 3, 4, 0, 0, 0, 0, 0, 0

solve (arr)

// print 50, 2, 3, 4, 5, 0, 0, 0, 0, 0

}

solve (int arr [])

{

// size - 8

// arr - 104

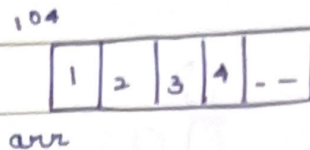
// 8 arr - 104 104 104

arr[0] = 50;

new pointer  
has been created  
stores add of  
array

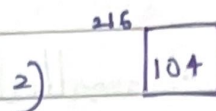
(main n func arr not same, alage se dabra on)

exp:- `int arr[10] = {---}`



`solve(int arr[])`

1) pointer aaya



arr (name rakha)

8 byte space

3) `arr[0] = 50`

$\rightarrow * (arr + 0)$

$\rightarrow * (104 + 0)$

$\rightarrow * (104)$

eg2.

`main()`

```
{  
  int a = 5;  
  int * p = &a;  
  solve(p)  
}
```

`solve(int * ptr)`

$* ptr = * ptr + 10$

$* (104) = * (104) + 10$

$= 5 + 10$

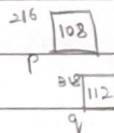
$= 15$

p copied in ptr  
(int \* ptr = p)

eg3.

`main()`

```
{  
  int arr[4] = {10, 20, 30, 40};  
  int * p = &arr[1];  
  int * q = &arr[2];  
  update(p, q);  
}
```



`update(int * a, int * b)`

{

$* a = 100 * (108) = 100$

$* b = 200; * (112)$

$= 200$

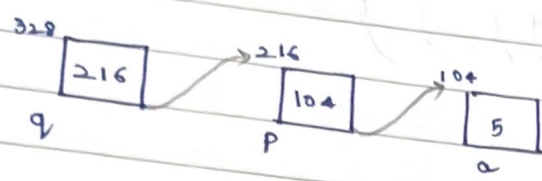
}



# Video 4

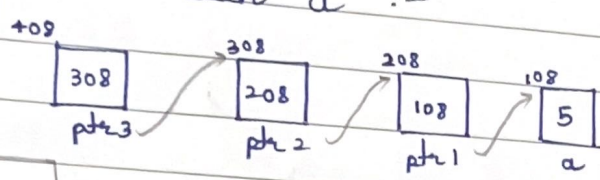
Double pointer =  $\text{int}^{**}q;$

eg.  $\text{int } a = 5;$   
 $\text{int } *p = \&a;$   
 $\text{int } **q = \&p;$  ( $q$  is a pointer  $\text{int}^*$  data)



•  $\text{cout} \ll **q \ll \text{endl}; \rightarrow 5$

• ways to reach a :-



ways :-  $a, *ptr1, **ptr2, *ptr3$   
 (itself also)

•  $\text{main}() \{$

$\text{int } a = 5;$   
 $\text{int } *p = \&a;$

$// \text{print } a, p, *p$   
 $5, 104, 5$

$\text{while}(p)$

$// \text{print } a, p, *p$   
 $5, 104, 5$

$\text{while}(\text{int } *p)$

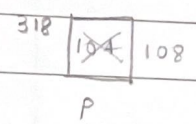
$p = p + 1;$

}

or

$*p = *p + 1$

$* (104) = *(104) + 1$   
 $= 6$



job tak difference nahi karunge tab tak no value change. eg.  $*ptr = *ptr + 1$  (not util  $(int * p)$ )

→ **Reference variable** :- (can't set on null) (simple to use)  
 → same memory location - diff name

`int a = 5;` b is a reference variable & pointing  
`int &b = a;` to d same memory location of a

S.T
a → 104
b → 104

(only entry in S.T for b no diff <sup>block</sup>)

eg. pass by ref

(int)

```
main()
{
  int a = 5;
  solve(a);
  cout << a;
}
```

reference

```
solve(int &value) / (int a)
{
  value++;
}
↓
OP = 6
```

pass by value  
 ↓  
 OP = 5

(pointer)

```
main {
  int a = 5;
  int * p = &a;
  solve(p);
  cout << p;
}
```

```
void solve(int * &p)
{
  p = p + 1;
}
```



→ V. popular of :-

```
int * solve ()  
{
```

```
    int a = 5;
```

```
    int * ans = &a;
```

```
    return ans;  
}
```

→ O/P = 8a

→ Return by reference :-

↘ when v return address

- Do week 6 assignment for clarity

- ```
int arr[10];
```

- ```
int (* ptr)[10] = &arr;
```

- ```
p[i] → *(p+i)
```

- ```
(*p)[i] → *(*p+i)
```

} connect vid