



# Ch 6(L-3) : Pointers III

## Double Pointers

▼ It is a pointer to `int*` data or pointer

```
int a =5;
int* p = &a;
int** q = &p;
```

## Reference Variable

- I can call same memory location with different name
- `int& b = a;` → b is a reference variable and pointing to same memory location of a i.e entry sirf symbol table me gai h koi new block create nhi hua

▼ Why to use reference variable ?

1. Reference variable **cannot be set NULL** but pointer can set as NULL
2. Reference variable **are simpler to use and more readable** while Pointers are difficult to read and write

▼ Reference Variable is always pass by reference

```
#include<iostream>
using namespace std;

void solve(int& val){
    val ++;
}

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

```

    cout << a; // output -> 6
}

```

## ▼ Some Important points

### ▼ How to pass pointer as pass by reference

```

#include<iostream>
using namespace std;

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

int main(){
    int a=5;
    int* p = &a;
    cout << p <<endl; //output -> 2c
    solve(p);
    cout << p; // output -> 30
}

```

### ▼ An array of pointers is an array where each elements is a pointer, while a pointer to an array is a pointer that points to the first element of array

```

int* arr[3]; //array of pointers to int
int nums[3] = {1,2,3}
int* ptr = nums; //pointer to int (points to the first element)

```

### ▼ `int (*ptr)[10]` is a pointer to an array of 10 integers in C++. This means that `ptr` is pointer to the first element of an array that contain 10 integers.

```

int nums[10] = {1,2,3,4,5,,6,7,8,9,10};
int (*ptr)[10] = &nums; //pointer to an array of 10 integers

```

### ▼ Return by pass by reference

### ▼ Output of the code

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