Summary

1. Purpose of Double Pointers (Pointers to Pointers):

- Double pointers (int **ptr) are used when:

* You want to modify a pointer inside a function (e.g., dynamic memory allocation with malloc inside a funct

* You're working with 2D arrays dynamically.

* You're dealing with arrays of strings (e.g., char **argv in main()).

* Useful in data structures like linked lists, trees, etc., when manipulating pointers to nodes.

Example:

```
void allocate(int **ptr) {
*ptr = malloc(sizeof(int));
}
```

2. Relation Between Pointers, Arrays, and Strings:

- An array name is essentially a pointer to its first element.

- You can access array elements using pointer arithmetic: a[i] == *(a + i)

- A string in C is a character array terminated with a null character '\0'.

- So, char *str = "Hello"; points to the first character 'H' of the string.

Key Idea:

Pointers allow flexible traversal and manipulation of arrays and strings without using array indices.

3. Purpose of Pointer to Function:

- A function pointer stores the address of a function and allows dynamic function calls at runtime.

- Enables callbacks (e.g., in sorting functions like qsort).

- Useful in function tables, plugin systems, or event-driven programming.

Example:

```
int add(int a, int b) { return a + b; }

int (*func_ptr)(int, int) = add;

printf("%d", func_ptr(2, 3)); // Calls add(2, 3)
```