ye dry run kro:

```cpp
#include <iostream>
using namespace std;
int main()
{
    char arr[][4] = {'1', '2', '3', '4', '5', '6', '7', '8', '9'};

    // for(int i = 0; i < 9; i++) {
    //     cout<<(void *)(&arr[0][i])<<endl;
    // }
    // cout<<endl;

    char *ptr = *arr;
    char * *ptr2 = &ptr;

    cout << ptr << endl;
    cout << (void *)ptr << endl;
    cout << *ptr << endl;
    cout << *(++ptr) << endl
         << endl;

    cout << ptr2 << endl;
    cout << (*ptr2) << endl;
    cout << (++*ptr2) << endl
         << endl;

    cout << ++ ++*((ptr += 1) + 1) << endl;
    cout << *ptr2 << endl;
    cout << * *ptr2 << endl;
    cout << * (++ *(ptr2) += 2 + 1) << endl;
    cout << *ptr2 << endl;
}
```

Code Walkthrough

1. Array Initialization

cpp

Copy code

```
char arr[][4] = {'1', '2', '3', '4', '5', '6', '7', '8', '9'};
```

This creates a **2D character array** with 3 rows and 4 columns. However, since we only initialized 9 characters, it looks like this in memory (row-major order):

css

Copy code

arr[0][0] = '1', arr[0][1] = '2', arr[0][2] = '3', arr[0][3] = '4'

arr[1][0] = '5', arr[1][1] = '6', arr[1][2] = '7', arr[1][3] = '8'

arr[2][0] = '9'

---

2. Pointer Initializations

cpp

Copy code

char *ptr = *arr;

- *arr is equivalent to arr[0], which points to the **first element** of the first row ('1').
- Now, ptr points to the first element of arr, i.e., '1'.

cpp

Copy code

char **ptr2 = &ptr;

- &ptr is the address of the pointer ptr.
- Now, ptr2 is a **pointer to the pointer** ptr.

---

3. First Set of Outputs

cpp

Copy code

cout << ptr << endl;

- ptr is a pointer to the first character ('1'), so it prints the **entire string** starting from '1':
  Output: 123456789.

cpp

Copy code

cout << (void *)ptr << endl;

- This outputs the **address** of ptr, converted to void *.
  Output: Address of '1'.

cpp

Copy code

cout << *ptr << endl;

- Dereferencing ptr gives the value it points to, which is '1'.
  Output: 1.

cpp

Copy code

cout << *(++ptr) << endl << endl;

- ++ptr increments the pointer, moving it to the next character ('2').
- Dereferencing ptr gives '2'.
  Output: 2.

---

4. Second Set of Outputs

cpp

Copy code

cout << ptr2 << endl;

- ptr2 holds the address of ptr. This prints the **address of ptr**.
  Output: Address of ptr.

cpp

Copy code

cout << (*ptr2) << endl;

- Dereferencing ptr2 gives ptr. Since ptr points to '2', this prints the **entire string** starting from '2':
  Output: 23456789.

cpp

Copy code

cout << (++*ptr2) << endl << endl;

- *ptr2 is ptr. Incrementing *ptr2 moves ptr one step forward, so it now points to '3'.
- Dereferencing ptr2 gives ptr, which prints the **entire string** starting from '3':
  Output: 3456789.

---

5. Third Set of Outputs

cpp

Copy code

cout << ++ ++ *((ptr += 1) + 1) << endl;

- ptr += 1: Moves ptr to the next character, now pointing to '4'.
- (ptr += 1) + 1: Adds 1 to the updated ptr, so this now points to '5' (start of the next row).
- *((ptr += 1) + 1): Dereferences the pointer to '5', then moves **two characters forward**, landing on '7'.
- ++ ++: Increments '5' twice to '7'.
- **Output**: 7.

cpp

Copy code

cout << *ptr2 << endl;

- *ptr2 gives ptr. Since ptr now points to '4', this prints the **entire string** starting from '4':
  Output: 476789.

cpp

Copy code

cout << * *ptr2 << endl;

- **ptr2 dereferences ptr2 to get ptr, and dereferencing ptr gives '4'.
  Output: 4.

cpp

Copy code

cout << * (++ *(ptr2) += 2 + 1) << endl;

- ++ *(ptr2): Increments *ptr2 (i.e., ptr) to point to the next character ('5').
- += 2 + 1: Moves ptr forward by 3 steps, so ptr now points to '8'.
- Dereferencing ptr gives '8'.
  Output: 8.

cpp

Copy code

cout << *ptr2 << endl;

- *ptr2 gives ptr. Since ptr now points to '8', this prints the **entire string** starting from '8':
  Output: 89.

Final Output

css

Copy code

123456789

<Address of '1'>

1

2

<Address of ptr>

23456789

3456789

7

476789

4

8

89