In C++, an array of pointers is an array where each element is a pointer to a memory location, while a pointer to an array is a pointer that points to the first element of an array.

For example, consider the following code:

```
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 of nums)

// assigning pointers to elements in the array of pointers

arr[0] = &nums[0];

arr[1] = &nums[1];

arr[2] = &nums[2]; // accessing elements using array notation

cout << *arr[0] << endl; // prints 1

cout << *arr[1] << endl; // prints 2

cout << *arr[2] << endl; // prints 3

// accessing elements using pointer notation

cout << *ptr << endl; // prints 1

cout << *(ptr + 1) << endl; // prints 2

cout << *(ptr + 2) << endl; // prints 3
```

In this example, `arr` is an array of pointers to `int`, and `ptr` is a pointer to the first element of the `nums` array. The `arr` array is initialized to point to the elements of the `nums` array. We can access the elements of the `arr` array using array notation, or we can access the elements of the `nums` array using pointer notation.

In summary, an array of pointers is an array where each element is a pointer, while a pointer to an array is a pointer that points to the first element of an array.

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

For example, consider the following code:

```
int nums[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

int (*ptr)[10] = &nums; // pointer to an array of 10 integers

// accessing elements of the array using pointer notation

cout << (*ptr)[0] << endl; // prints 1
```

```
cout << (*ptr)[1] << endl; // prints 2

cout << (*ptr)[2] << endl; // prints 3
```

In this example, `nums` is an array of 10 integers, and `ptr` is a pointer to the first element of `nums`. The parentheses around `*ptr` are necessary because the dereference operator `*` has a lower precedence than the array subscript operator `[]`.

We can access the elements of the array using pointer notation, but we need to use the parentheses to first dereference the pointer and then apply the subscript operator to the array.

In summary, `int (*ptr)[10]` is a pointer to an array of 10 integers in C++.