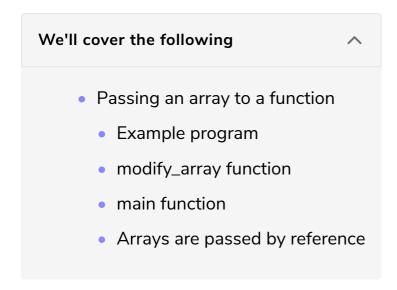
Arrays and Functions

Let's get into the details of passing an array to a function.



Passing an array to a function

To pass an array to a function, we just have to specify the array type, followed by an array name and square brackets in the function parameters.

```
ReturnType FunctionName ( int ArrayName []) {

/// Function Body

int main ( ) {

/// Function Body

FunctionName ( ArrayName )

}
```

Example program

Let's write a program that takes an array in its parameters.

In the program, we will traverse the array elements. If the value of an array

element is less than 50, then we will update the value at that index to -1.

Press the **RUN** button and see the output!

```
#include <iostream>
using namespace std;
// print_array function will print the values of an array
void print_array(int number[], int size) {
  for (int i = 0; i < size; i++) {
    cout << number[i] << " ";</pre>
  cout << endl;</pre>
}
// modify_array function
void modify_array(int number[], int size) {
  // Traverse array
  for (int i = 0; i < size; i++) {
    // If value less tha 50 set it to -1
    if (number[i] < 50)</pre>
      number[i] = -1;
  cout << "Values of array inside the function:" << endl;</pre>
  // Call print_array function
  print_array(number, size);
// main function
int main() {
  // Initialize size of an array
  int size = 8;
  // Initialize values of array
  int number[size] = {67, 89, 56, 43, 29, 15, 90, 67};
  cout << "Values of array before function call:" << endl;</pre>
  // Call print_array function
  print_array(number, size);
  // Call modify_array function
  modify_array(number, size);
  cout << "Values of array after function call:" << endl;</pre>
  // Call print_array function
  print_array(number, size);
```

In the code above:

modify_array function #

modify_array function takes an array of type int and int value in its input parameters.

Line No. 16: Uses for loop to traverse array from i = 0 to i = size-1

Line No. 18: Checks if the value of the number[i] is less than **50**. If true, then it executes **Line No. 19**.

Line No. 19: Sets number[i] to -1

Line No. 23: Calls the print_array function to print the values of an array inside the function

main function #

Line No. 29: Initializes the size of an array

Line No. 31: Initializes the values of the array number

Line No. 35: Calls print_array function to print the values of an array

Line No. 37: Calls the modify_array function

Line No. 40: Calls print_array function to print the values of an array after calling the modify_array function

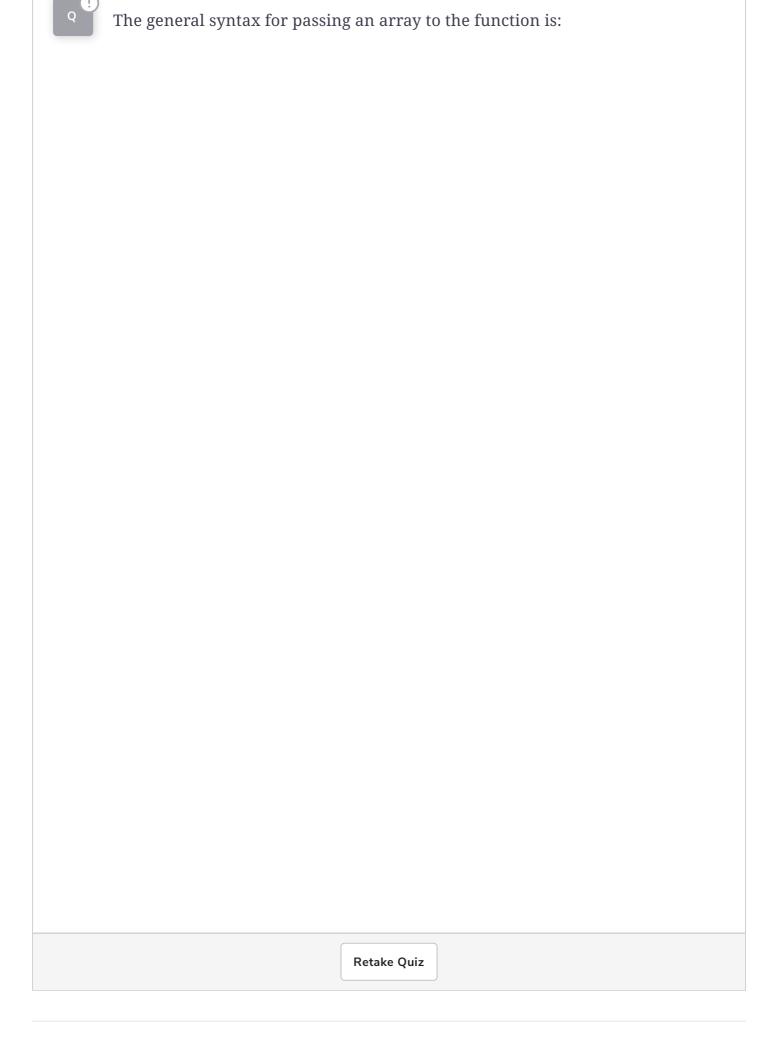
Arrays are passed by reference

In the code above, did you notice that any change made in the elements of an array inside the modify_array function are reflected in the main function?

This was not the case with variables because, by default, variables are passed by value.

When we pass an array to the function, we don't need to specify the size of an array in square brackets. This is because we need the size of an array when we are creating a new array. However, when we pass an array in the function, we are just passing an original array to the function. This means if we made any changes inside the function, we would see that changes outside the function. That is why we can say by default arrays are passed by reference.

/ N	٦	٦	п	\neg
U	ι	J	л	L
T	-	-	_	



That's all about one-dimensional arrays. Let's dive right in and see the implementation of two-dimensional arrays in C++.