What is Array Decay in C++? How can it be prevented?

Difficulty Level : Medium ● Last Updated : 21 Sep, 2021

What is Array Decay?

The loss of type and dimensions of an array is known as decay of an array.This generally occurs when we pass the array into function by value or pointer. What it does is, it sends first address to the array which is a pointer, hence the size of array is not the original one, but the one occupied by the pointer in the memory.

CPP

```cpp
// C++ code to demonstrate array decay
#include<iostream>
using namespace std;

// Driver function to show Array decay
// Passing array by value
void aDecay(int *p)
{
    // Printing size of pointer
    cout << "Modified size of array is by "
            " passing by value: ";
    cout << sizeof(p) << endl;
}

// Function to show that array decay happens
// even if we use pointer
void pDecay(int (*p)[7])
{
    // Printing size of array
    cout << "Modified size of array by "
            "passing by pointer: ";
    cout << sizeof(p) << endl;
}

int main()
{
    int a[7] = {1, 2, 3, 4, 5, 6, 7,};

    // Printing original size of array
    cout << "Actual size of array is: ";
    cout << sizeof(a) <<endl;

    // Passing a pointer to array
    aDecay(a);

    // Calling function by pointer
    pDecay(&a);

    return 0;
}
```

Output:

```
Actual size of array is: 28
Modified size of array by passing by value: 8
Modified size of array by passing by pointer: 8
```

In the above code, the actual array has 7 int elements and hence has 28 size. But by calling by value and pointer, array decays into pointer and prints the size of 1 pointer i.e. 8 (4 in 32 bit).

How to prevent Array Decay?

A typical solution to handle decay is to pass size of array also as a parameter and not use sizeof on array parameters (See this for details)

Another way to prevent array decay is to send the array into functions by reference. This prevents conversion of array into a pointer, hence prevents the decay.

CPP

```cpp
// C++ code to demonstrate prevention of
// decay of array
#include<iostream>
using namespace std;

// A function that prevents Array decay
// by passing array by reference
void fun(int (&p)[7])
{
    // Printing size of array
    cout << "Modified size of array by "
            "passing by reference: ";
    cout << sizeof(p) << endl;
}

int main()
{
    int a[7] = {1, 2, 3, 4, 5, 6, 7,};

    // Printing original size of array
    cout << "Actual size of array is: ";
    cout << sizeof(a) <<endl;

    // Calling function by reference
    fun(a);

    return 0;
}
```

Output:

```
Actual size of array is: 28
Modified size of array by passing by reference: 28
```

In the above code, passing array by reference solves the problem of decay of array. Sizes in both cases is 28.

This article is contributed by **Manjeet Singh** .If you like GeeksforGeeks and would like to contribute, you can also write an article using write.geeksforgeeks.org or mail your article to review-team@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

GeeksforGeeks

5th Floor, A-118,
Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org