C++ NULL POINTERS

It is always a good practice to assign the pointer NULL to a pointer variable in case you do not have exact address to be assigned. This is done at the time of variable declaration. A pointer that is assigned NULL is called a **null** pointer.

The NULL pointer is a constant with a value of zero defined in several standard libraries, including iostream. Consider the following program:

```
#include <iostream>

using namespace std;

int main ()
{
   int  *ptr = NULL;

   cout << "The value of ptr is " << ptr ;

   return 0;
}
```

When the above code is compiled and executed, it produces the following result:

```
The value of ptr is 0
```

On most of the operating systems, programs are not permitted to access memory at address 0 because that memory is reserved by the operating system. However, the memory address 0 has special significance; it signals that the pointer is not intended to point to an accessible memory location. But by convention, if a pointer contains the null *zero* value, it is assumed to point to nothing.

To check for a null pointer you can use an if statement as follows:

```
if(ptr)     // succeeds if p is not null
if(!ptr)    // succeeds if p is null
```

Thus, if all unused pointers are given the null value and you avoid the use of a null pointer, you can avoid the accidental misuse of an uninitialized pointer. Many times, uninitialized variables hold some junk values and it becomes difficult to debug the program.