

- Example

An example of implementing nullptr.

WE'LL COVER THE FOLLOWING ^

- Example
- Explanation

Example

```
// nullptr.cpp
#include <iostream>
#include <string>

std::string overloadTest(char*){
    return "char*";
}

std::string overloadTest(int){
    return "int";
}

std::string overloadTest(long int){
    return "long int";
}

std::string test2(char*){
    return "char*";
}

int main(){

    std::cout << std::endl;

    int* pi = nullptr;    // OK
    // int i= nullptr;    // cannot convert 'std::nullptr_t' to 'int'
    bool b{nullptr};      // OK. b is false.

    std::cout << std::boolalpha << "b: " << b << std::endl;

    // calls int
    std::cout << "overloadTest(0) = " << overloadTest(0) << std::endl;

    // calls char*
    std::cout << "overloadTest(static_cast<char*>(0)) = " << overloadTest(static_cast<char*>(0))
```

```

std::cout << "overloadTest(static_cast<char*>()) << "overloadTest(static_cast<char*>('0'))
std::cout<< "test2(0)= " << test2(0) << std::endl;

// calls char*
std::cout << "overloadTest(nullptr)= " << overloadTest(nullptr)<< std::endl;

// call of overloaded 'overloadTest(NULL)' is ambiguous
// std::cout << "overloadTest(NULL)= " << overloadTest(NULL) << std::endl;

std::cout << std::endl;
}

```



Explanation

- The `nullptr` can be used to initialize a pointer of type `int` (line 25). However, the `nullptr` cannot be used to initialize a variable of type `int` (line 26).
- The null pointer constant behaves like a `boolean` value, which is initialized with `false` (line 27). If the `nullptr` has to decide between a `long int` and a pointer, it will result in a pointer (line 39).

Simple rule to remember: Use `nullptr` instead of 0 or `NULL`.

For further information, visit [nullptr](#).

The next lesson will introduce you to the user-defined literals.