Dangling Pointers and the Null Value

Just like normal variables, pointers are not initialized when declared. If not initialized with a value, they will be undefined. Once the memory a pointer has been pointing to, has been released, a pointer is also undefined. If we reference an undefined pointer, we do not know to what variable or value it is pointing, and cannot predict what our program will compute. This is important when we work with linked data structures (which you will study in your second year).

To guard against inadvertently accessing an undefined pointer, we can set pointers that are undefined to a null value, which means that the pointer is not pointing to anything. We can then test whether or not a pointer has a null value before dereferencing it. A pointer with a null value will evaluate to the Boolean value `false`, and otherwise to `true`.

We can assign a null value to a pointer by using the value 0, e.g.:

```
int* ptr;  //declare pointer ptr
int a = 15; //declare variable a
ptr = 0;  // assign null value to pointer ptr
if (!ptr) //test whether ptr is pointing to anything
    ptr = &a; //if ptr is not pointing to a variable, let it point to
          // the address in memory where variable a is stored.
if (ptr)  //if ptr is pointing to a variable,
    cout << *ptr<< endl; // display the value of that variable
```

We can also set a pointer to the null value by using the preprocessor macro `NULL` inherited from C to indicate that a pointer is undefined. If you want to use `NULL`, you have to include the `<cstddef>` library with a `#include <cstddef>` statement.

```
#include <cstddef>
int *ptr;
ptr = NULL;
```

C++11 has introduced a special keyword `nullptr` to use a null value. This keyword can be used in the same way as you would use 0 to initialise a pointer to the null value. The version of Code::Blocks we use, only allow us to use 0 and the preprocessor macro NULL inherited from C to indicate that a pointer is undefined or dangling.

Here is a website where you can learn more and see more examples:

https://www.learncpp.com/cpp-tutorial/6-7a-null-pointers/

Note that this website use extended initialiser lists for pointers. Extended initialiser lists for pointers is only available with C++11, so you will receive a warning in the prescibed version of Code::Blocks.