What are Pointers?

This lesson introduces the concept of pointers in C++

We'll cover the following

Definition

> A pointer is a *special* kind of variable that stores the **address** in *memory* of another variable and can be used to manipulate that variable.

In general, whenever a variable is used in C++, it must exist somewhere in the *host computer's memory* and pointers can store the *location of a particular variable*.

What do Pointers Hold?

Pointers associate **two** pieces of information:

- The **memory address**, which is the *"value"* of the pointer itself.
- The **data type** of the variable *pointed to*, which is the *kind* of variable located at that address.

Dereferencing Pointers

Pointers can be **dereferenced** to access the value of the variable at the *pointer's address*.

For example:

```
void f(int* p)
{
    // The code "*p" takes the value of the data at location stored in p
    int n = *p;
}
```

Storage

> **Note:** All the data stored in a program is stored in the computer's memory.

For example, if a program has a *global* variable named `numberOfEmployees` that variable has both **a value**, for example, **120** employees, and **an address**, which is the *actual* location in the computer's *memory* where the value is *stored*.

Declaring pointers

Declaring a *pointer* is similar to declaring a *regular* variable, although the name is *preceded* by an **asterisk**:

```
int *ptr;
struct coord *pCrd;
void *vp;
```

In the example above

- `ptr` is a *pointer* to an **integer**.
- `pCrd` is a *pointer* to a **structure** named `coord`.
- `vp` is a `void` *pointer*, which does not require a specific data type.

> **Note:** Unlike references, *pointers* are not guaranteed to be *initialized*. As such, they should only be used when they are known to point to an *existing* object.

That completes our discussion on pointers. Next, we'll discuss *arrays* in C++.