**References**

A computer’s memory is a sequence of bytes. We can number the bytes from 0 to the last one, with each number representing the address, or the location in the memory.

1. **References**

In C++, a *reference* variable is an alias for something else.

We MUST initialize the reference when declared.

Suppose we already have an **int** variable called **Minh**, we can create an alias to it by using the **&** sign in the declaration:

int &Michael = Minh;

🡪 Anything we do to Michael/Minh will also be done to the other, and Aliases cannot be changed to alias something else.

1. **Pass-By-Reference**

This means passing parameters to a function by using references 🡪 When called, the function can change the value of the outside variable (if we reassign the argument)

🡪 This will avoid the creation of copies of variable/objects for performance reasons.

Text

Description automatically generated

Shape

Description automatically generated with medium confidence

1. **Pass-By-Reference with Const**

Recall: using the const keyword tells the compiler that we won’t change something (will not reassign)

🡪 If we pass by reference a const variable to a function (int &i), the compiler will throw error

\*\* Sometimes, we still use const in the function parameter to write a function where the parameter will not be reassigned.

int triple(int const i) { return i \* 3; }

Therefore, to save computational cost for a function that doesn’t modify the parameter values, we can also do:

int triple(int const &i) { return i \* 3; }

\*\*\*Notice that since we are passing by reference WITH the const keyword, the variable value outside still would not be changed. However, doing this will maximize the computational cost of making a copy of the argument.

1. **Memory Address**

The & also means “address of”. Therefore, this operator is used to get the memory address of an object.

Make sure to note that:

* When & is used in a declaration, it is a reference operator.
* When & is not used in a declaration, it is an address operator.

*(declaration === left side of assignment statement)*