The Heap

This is the counter-part of the Stack. The heap is an unregulated area of memory, which is both good and bad as we'll learn below.

We'll cover the following

What is Heap?

The heap is a region of your computer's memory that is not managed automatically for you, and is not as tightly managed by the CPU. It is a more free-floating region of memory (and is larger).

Unlike the stack, the heap does not have size restrictions on variable size (apart from the obvious physical limitations of your computer).

Also, unlike the stack, variables created on the heap are accessible by any function, anywhere in your program. Heap variables are essentially global in scope.

However, heap memory is slightly slower to be read from and written to, because one has to use **pointers** to access memory on the heap. We will talk about pointers shortly.

Memory Leak

To allocate memory on the heap, you must use `malloc()` or `calloc()`, which are built-in C functions. Once you have allocated memory on the heap, you are responsible for using `free()` to deallocate that memory once you don't need it any more. If you fail to do this, your program will have what is known as a **memory leak**. That is, memory on the heap will still be set aside (and won't be available to other processes).

As we will see in the debugging section, there is a tool called `valgrind` that can help you detect memory leaks.

So the stack and the heap have their own benefits and dangers in terms of memory allocation. The next lesson compares their properties.