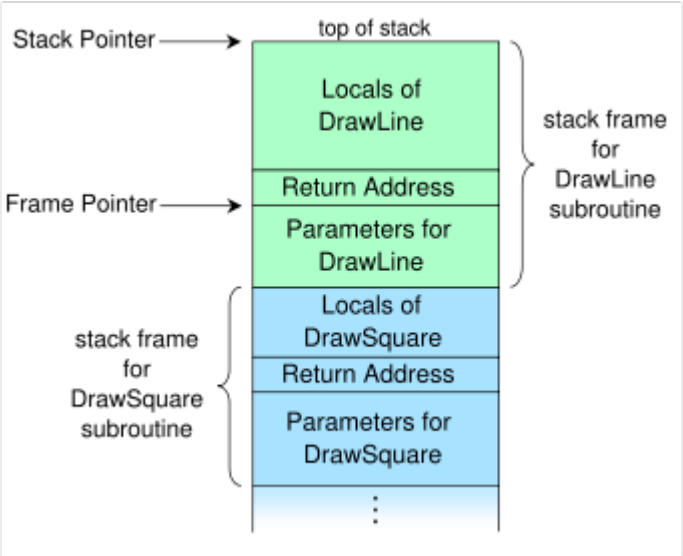
The Stack versus the Heap

The memory involved in a function call, including the values of the function's parameters and local variables, is allocated from the part of memory that is called the ***Stack***.

- Each function ***call*** pushes a new ***stack frame*** onto the Stack. The frame holds the parameters, the local variables and the address in the code to which to return when the function exits.
- Each ***return*** from a function pops the top stack frame off of the Stack and returns control to the address stored in the popped stack frame.



The memory involved in the construction of a new object is achieved by a call to ***malloc***, which allocates rom the part of memory that is called the ***Heap***. The memory manager keeps track of where everything is on the heap.

- In Python, a ***garbage collector*** runs in the background. When an object (place in memory) has no variable referring to it, the garbage collector tells the memory manager that that space is now available to be re-allocated.
- In C, de-allocating space is the programmer's responsibility. It is accomplished by a call to ***free***. Failing to de-allocate space leads to a ***memory leak***.