Heap

Heap is the data model used by virtually all C++ compilers to implement the free-store. Free-store is an abstract term referring to unused memory that is available for dynamic allocations.

In practice, however, the distinction between heap and free-store is roughly equivalent to the differences between the memory models of C and C++, respectively. In C, you allocate memory dynamically using **malloc()**, and release the allocated memory using **free()**. The allocated memory is not initialized. Dynamic allocations in C are said to take place on the heap. In C++, you perform dynamic allocations with new and release the allocated memory with delete. Operator new automatically initializes the allocated memory (by calling the object's constructor), and delete automatically destroys the object before releasing its memory. In C++, dynamic allocations are said to take place on the free-store.

Note that the heap and free-store may reside on distinct physical memory regions and they might be controlled by different underlying memory managers.

Heap memory

The heap is a different thing altogether. Different PC's have different amounts of memory so it would be silly to have to allocate all the memory at compile time. If a particular computer has a lot of RAM, why not make use of it? Therefore the heap consists of memory that can be allocated at run time. The amount of heap memory is different for different machines and a well written program will take account of this (allocating heap memory sparingly - and not assuming that everyone will have massive amounts of it). Heap memory is most often used for 'large' things (e.g. the contents of a file) - or for things that need to 'stay around' in memory so that different functions can share the data (in other words, for NON-local variables). It's also used very frequently to allocate memory for things whose size is not known until run time. For example, if you had to open a file and read in the data, you don't know how much data exists until run time. In contrast, the size of objects allocated on the stack, needs to be known at compile time.

Whereas your program takes care of stack memory (allocating it & cleaning it up) it's the programmer's job to deal with heap memory. The size of objects allocated on the heap is often not known to the compiler - therefore the programmer must allocate and release the memory specifically.