Chapter 12: Memory Management

Notes

- The standard changes the old way ‘new’ used to work in which it returned NULL if the allocation failed. If ‘new’ fails under the standard, it throws an exception of type ‘std::bad_alloc’, making a NULL check pointless. The standard defines a way to use ‘new’ without the possibility of exception by passing ‘nothrow’: new (nothrow) object; Obviously, pre-standard implementations of C++ do not support ‘nothrow’ so you really can’t win if you want true portability.
- POD (Plain Old Data) objects are those who do not need a constructor to be properly initialized. Such as primitive types or classes with primitive data. They can be used without initializing them. Non-POD objects must be initialized properly before they can be used. This is where ‘new’ excels over ‘malloc’ because it automatically calls the constructor. Of course, if you use ‘malloc’ to allocate memory for **classes** then you’re asking for trouble.
- Three categories of storage: automatic, static, and free (dynamic).
- ‘new’ and ‘delete’ are considered, confusingly, both operators *and* functions. They boil down to implicit implementation-dependant functions.

Introduction