

Problem 11.7

7. Explain the dangers of C's approach to encapsulation.

The primary means of encapsulating a data type in C is to place the implementation code in a source code file and the public interface declarations in a header file. The source code file can then be compiled and only an object code file given to the user, along with the header file. The user then includes the header file in their code and links to the object file. The major problem with this approach is that there is no means to ensure that the header file is properly matched to the object code file. It is entirely possible for the user to be using the updated version of one file while using a non-updated version of the other. Furthermore, there is nothing to prevent the user from simply copying and pasting the contents of the header file into their own code which almost ensures that even if they get updated copies of both files that their code will be working with outdated header information.

Problem 11.8

8. Why didn't C++ eliminate the problems discussed in Problem 7?

Since C++ is, nominally, a superset of the C language, it inherits (no pun intended) all of the warts of the C language. Furthermore, the C++ compiler uses (at least the original versions) the C linker, which also forces it to be exposed to many of the same problems as C programs are.

Problem 11.12

12. Why are destructors rarely used in Java but essential in C++?

The driving difference is that Java uses automatic garbage collection while C++ does not. As a consequence, the programmer has virtually no control over when, or even if, a destructor would be called in Java. The C++ programmer, on the other hand, not only has complete control over when the destructor is called, but this is the primary mechanism by which the memory resources can be reclaimed.

Problem 12.15

15. Explain why naming encapsulations are important for developing large programs.

Naming encapsulations, generally known as "namespaces" act like area codes for telephone numbers. Telephone numbers only need to be deconflicted with other telephone numbers within the same area code and no concern needs to be given regarding the assignment of numbers to telephones in other area codes. Similarly, functions, methods, constants, and other resources in one namespace can be duplicates of those in other namespaces and there is no conflict between them because each is accessible via its own namespace modifier, just as any phone number is distinguishable from other, identical, phone numbers via the use of the area code.