**Chapter 1 Homework Solutions**

*Review Questions:*

1. It is useful for a programmer to have some background in language design, even though he or she may never actually design a programming language because they will then better understand how the computer will interpret the programs they write, and thus be able to write more efficient programs. Also it gives them an increased capacity to express ideas, an improved background for choosing a language to write with, increased ability to learn new languages, better understanding of the significance of implementation, better use of languages already known, and overall advancement of computing.

13. A program is said to be reliable if it performs to its specifications under all conditions. In other words, it has the ability to encounter errors, recover from them and move on in the program.

17. Readability is important to writability because how easy it is to read and understand a program is greatly affected by how easy it is to write the program. Also the more options there are for writing a certain construct, the harder it will be for the reader to know all such possible ways to write the same thing, and this could affect the reader’s ability to understand the program.

22. Three fundamental features of an object-oriented programming language are data abstraction, inheritance, and polymorphism. Data abstraction encapsulates processing with data objects and it also controls access to data. Data abstraction is useful because it protects information from being changed. Inheritance enhances the potential reuse of existing software and provides a significant increase in software development. Polymorphism allows data types and functions to belong to more generic classes, which in return allows different data types to be processed in an orderly manner.

28. A linker is a systems program that is used in the process of collecting system programs and linking them to user programs, which is known as linking and loading.

*Problem Set:*

2. {Example} Some features of Fortran whose rationales are a mystery to me are: the kind statement and the case statement.

3. Some arguments for having a single language for all programming domains are: It would dramatically cut the costs of programming training and compiler purchase and maintenance; it would simplify programmer recruiting and justify the development of numerous language dependent software development aids.

4. Some arguments against having a single language for all programming domains are: The language would necessarily be huge and complex; compilers would be expensive and costly to maintain; the language would probably not be very good for any programming domain, either in compiler efficiency or in the efficiency of the code it generated. More importantly, it would not be easy to use, because regardless of the application area, the language would include many unnecessary and confusing features and constructs (those meant for other application areas). Different users would learn different subsets, making maintenance difficult.

6. The most common programming language statement that is most detrimental to readability is the “goto” statement.

8. The reasons why a language would distinguish between uppercase and lowercase in its identifiers are: (1) So that variable identifiers may look different than identifiers that are names for constants, such as the convention of using uppercase for constant names and using lowercase for variable names in C, and (2) so that catenated words as names can have their first letter distinguished, as in TotalWords. (Some think it is better to include a connector, such as underscore.) The primary reason why a language would not distinguish between uppercase and lowercase in identifiers is it makes programs less readable, because words that look very similar are actually completely different, such as SUM and Sum.

9. The different aspects of the cost of a programming language are: the cost of training programmers to use it, the cost of writing programs in the language, the cost of compiling programs, the cost of executing programs, the cost of the implementation system, the cost of poor reliability, the cost of maintaining programs.

10. One of the main arguments is that regardless of the cost of hardware, it is not free. Why write a program that executes slower than is necessary. Furthermore, the difference between a well-written efficient program and one that is poorly written can be a factor of two or three. In many other fields of endeavor, the difference between a good job and a poor job may be 10 or 20 percent. In programming, the difference is much greater.

17. The different aspects of the cost of a programming language are: the cost of training programmers to use it, the cost of writing programs in the language, the cost of compiling programs, the cost of executing programs, the cost of the implementation system, the cost of poor reliability, the cost of maintaining programs.