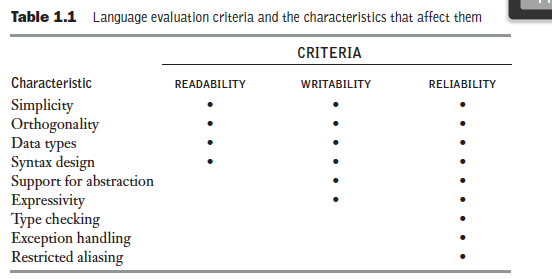
Felix Singerman – 7970742

Assignment 1

1. **Describe the main programming language evaluation criteria and its tradeoffs (0.5).**



That table provided in the textbook is a good overview of the evaluation criteria. I will briefly talk about readability, write-ability, and reliability. I will also talk about some of the trade-offs between them.

Readability: The ease which programs can be read and understood. Maintenance is very important in terms of cost. Because ease of maintenance is determined in large part by the readability of programs, readability became an important measure of the quality of programs and programming languages. Examples are: Simplicity, Orthogonality, Data types, and Syntax Design.

Write-ability: How easily a language can be used to create programs for a chosen problem domain. Most of the language characteristics that affect readability also affect write-ability. Because the process of writing a program requires the programmer to reread the part of the program that is already written. Write-ability must be considered in the context of the target problem domain of a language (i.e. C vs Visual basic for graphical user programs – Visual basic is better). Examples are: Support for abstraction, and Expressivity.

Reliability: Performs to its specifications under all conditions. Both readability and write-ability influence reliability. A program written in a language that does not support natural ways to express the required algorithms will necessarily use unnatural approaches. Unnatural approaches are less likely to be correct for all possible situations. The easier a program is to write, the more likely it is to be correct. Examples are: Type checking, Exception handling, Restricted aliasing.

Two criteria that conflict are reliability and cost of execution. For example, “Java demands that all references to array elements be checked to ensure that the index or indices are in their legal ranges. This step adds a great deal to the cost of execution of Java programs that contain large numbers of references to array elements. C does not require index range checking, so C programs execute faster although java is more reliable.” (Textbook)

The conflict between write-ability and reliability is another common one.  Example: “The pointers of C++ can be manipulated in a variety of ways, which supports highly flexible addressing of data. Because of the potential reliability problems with pointers, they are not included in Java.” (Textbook)

1. **Write an overview of the compilation process illustrated in the figure below, explain in general terms the lexical and syntax analyzers (1).**

The language that a compiler translates is called the source language. The process of compilation and program execution takes place in several phases, the most important are shown in the figure given to us.

The lexical analyzer gathers the characters of the source program into lexical units. The lexical units of a program are identifiers, special words, operators, and punctuation symbols. It ignores comments since the compiler has no use for them.

The syntax analyzer takes the lexical units from the lexical analyzer and uses them to construct hierarchical structures called parse trees. The parse trees represent the syntactic structure of the program.

The symbol table serves as a database for the compilation process. The contents of the symbol table are the type and attribute information of each user-defined name in the program. This information is all placed by the lexical and syntax analyzers. It is used by the semantic analyzer and code generator.

The intermediate code generator produces a program in a different language, at a in-between level of the source program and the final output of the compiler. These languages look like assembly languages. The semantic analyzer checks for errors, such as type errors, that are very hard to detect during analysis.

Optimization is an optional part which improves programs by making them smaller and/or faster.

The code generator translates the optimized intermediate code version of the program into an equivalent machine language program.

**Source code for Questions 3, 4, 5, 6 are attached.**

**For the prolog questions I used Sublime Text 2 for writing/editing the code and GNU Prolog in Terminal.app to run it. For scheme I used Dr. Racket for both editing and running.**