Characteristics of Type Systems\textsuperscript{1}

\textit{Type systems} for programming languages are typically designed to provide several important functions.

\textbf{Safety:} Type checking of programs should prevent (either at compile or run time) the execution of certain illegal operations. The first is a type error because that operation should never be applied to two operands, one of which is a string and the other of which is an integer. The second is not a type error because division is an operation that is normally applied to pairs of integers. However, when the operation is applied to certain combinations of values from those types, an error results. Thus, information on the types of the operands is not sufficient to determine whether the operation will be erroneous.

\textbf{Optimization:} Type checking can provide useful information to a compiler or interpreter. This information can be used to allocate storage for values, select appropriate code to execute (\textit{e.g.}, for overloaded operations), and support various optimizations.

\textbf{Documentation:} Type annotation (or, to a lesser extent, inference) provides documentation on constructs that make it easier for programmers to determine how the constructs can or should be used. Of course, the programmer should provide more than just type information as documentation, but our experience is that omission of type information significantly impacts the comprehensibility of code.

\textbf{Abstraction:} The ability to name types, and, even more importantly, the ability to hide the implementation of types, allows (even forces) the programmer to think at a higher level of abstraction in programming. This hiding of details allows more straightforward modeling of the problem domain, while making it possible to change the implementation of a type and its operations without impacting the correctness of programs using the implementation. Of course, an important reason for changing an implementation is to improve some aspect of the behavior of the program, but correctness of the program should be dependent only on the specification of the provided operations.

\textsuperscript{1}Foundations of Object Oriented Languages by Kim Bruce MIT Press © 2002 with slight modifications