Initial implementations of Java were all hybrid. Its intermediate form, called byte code, provides portability to any machine that has a byte code interpreter and an associated run-time system. Together, these are called the Java Virtual Machine. There are now systems that translate Java byte code into machine code for faster execution. However, Java applets are always downloaded from the Web server in the form of byte code.

Sometimes an implementor may provide both compiled and interpreted implementations for a language. In these cases, the interpreter is used to develop and debug programs. Then, after a (relatively) bug-free state is reached, the programs are compiled to increase their execution speed.

1.8 Programming Environments

A programming environment is the collection of tools used in the development of software. This collection may consist of only a file system, a text editor, a linker, and a compiler. Or it may include a large collection of integrated tools, each accessed through a uniform user interface. In the latter case, the development and maintenance of software is greatly enhanced. Therefore, the characteristics of a programming language are not the only measure of the software development capability of a system. We now briefly describe several programming environments.

UNIX is an older programming environment, first distributed in the middle 1970s, built around a portable multiprogramming operating system. It provides a wide array of powerful support tools for software production and maintenance in a variety of languages. In the past, the most important feature absent from UNIX was a uniform interface among its tools. This made it more difficult to learn and to use. However, UNIX is now often used through a graphical interface that runs on top of UNIX. One example of such a graphical interface is the Common Desktop Environment (CDE).

Borland JBuilder is a programming environment that provides an integrated compiler, editor, debugger, and file system for Java development, where all four are accessed through a graphical interface. This is a complex and powerful system for creating Java software.

The latest step in the evolution of software development environments is represented by Microsoft Visual Studio .NET. This is a large and elaborate collection of software development tools, all used through a windowed interface. This system can be used to develop software in any one of the five .NET languages, C#, Visual Basic .NET, JScript, J#, or managed C++.

The study of programming languages is valuable for a number of important reasons: It increases our capacity to use different constructs in writing programs, enables us to choose languages for projects more intelligently, and makes learning new languages easier. Computers are used in a wide variety of problem-solving domains. The design and evaluation of a particular programming language is highly dependent on the domain in which it is to be used. Among the most important criteria for evaluating languages are readability, writability, reliability, and overall cost. These will be the basis on which we examine and judge the various language features discussed in the remainder of the book.

The major influences on language design have been machine architecture and software design methodologies.

Designing a programming language is primarily an engineering feat, in which a long list of trade-offs must be made among features, constructs, and capabilities.

The major methods of implementing programming languages are compilation, pure interpretation, and hybrid implementation. Programming environments have become important parts of software development systems, in which the language is just one of the components.

REVIEW QUESTIONS

1. Why is it useful for a programmer to have some background in language design, even though he or she may never actually design a programming language?
2. How can knowledge of programming language characteristics benefit the whole computing community?
3. What programming language has dominated scientific computing over the past 40 years?
4. What programming language has dominated business applications over the past 40 years?
5. What programming language has dominated artificial intelligence over the past 40 years?
6. In what language is UNIX written?
7. What is the disadvantage of having too many features in a language?
8. How can user-defined operator overloading harm the readability of a program?
9. What is one example of a lack of orthogonality in the design of C?
10. What language used orthogonality as a primary design criterion?
11. What primitive control statement is used to build more complicated control statements in languages that lack them?
12. What readability problem is caused by using the same closing reserved word for more than one kind of control statement?
13. What construct of a programming language provides process abstraction?
14. What does it mean for a program to be reliable?
15. Why is type checking the parameters of a subprogram important?
16. What is aliasing?
17. What is exception handling?
18. Why is readability important to writability?
19. How is the cost of compilers for a given language related to the design of that language?
20. What has been the strongest influence on programming language design over the past 45 years?
21. What is the name of the category of programming languages whose structure is dictated by the von Neumann computer architecture?
22. What two programming language deficiencies were discovered as a result of the research in software development in the 1970s?
23. What are the three fundamental features of an object-oriented programming language?
24. What language was the first to support the three fundamental features of object-oriented programming?
25. What is an example of two language design criteria that are in direct conflict with each other?
26. What are the three general methods of implementing a programming language?
27. Which produces faster program execution, a compiler or a pure interpreter?
28. What role does the symbol table play in a compiler?
29. What does a linker do?
30. Why is the von Neumann bottleneck important?
31. What are the advantages in implementing a language with a pure interpreter?

### Problem Set

1. Do you believe our thinking capabilities are influenced by our language? Support your opinion.
2. What are some features of specific programming languages you know whose rationales are a mystery to you?
3. What arguments can you make for the idea of a single language for all programming domains?
4. What arguments can you make against the idea of a single language for all programming domains?
5. Name and explain another criterion by which languages can be judged (in addition to those discussed in this chapter).
6. What common programming language statement, in your opinion, is most detrimental to readability?
7. Java uses a right brace to mark the end of all compound statements. What are the arguments for and against this design?
8. Many languages distinguish between uppercase and lower-case in user-defined names. What are the pros and cons of this design decision?
9. Explain the different aspects of the cost of a programming language.
10. What are the arguments for writing efficient programs even though hardware is relatively inexpensive?
11. Describe some design trade-offs between efficiency and safety in some language you know.
12. What major features would a perfect programming language include, in your opinion?
13. Was the first high-level programming language you learned implemented with a pure interpreter, a hybrid implementation system, or a compiler? (You would not necessarily know this without research.)
14. Describe the advantages and disadvantages of some programming environment you have used.
15. How do type declaration statements for simple variables affect the readability of a language, considering that some languages do not require them?
16. Write an evaluation of some programming language you know, using the criteria described in this chapter.
17. Ada uses the semicolon to separate statements, while Java uses it to terminate statements. Which of these, in your opinion, is most natural and least likely to result in syntax errors? Support your answer.
18. Many contemporary languages allow two kinds of comments, one in which delimiters are used on both ends (for multiple-line comments), and one in which a delimiter marks only the beginning of the comment (for one-line comments). Discuss the advantages and disadvantages of each of these with respect to our criteria.