SEMtech!
Student Engagement & Mentoring in Technology

Chap 26 Programming languages

Compare and contrast programming language categories

Software developers write code and use programming languages to create software that meets an organization's business objectives. These programming languages each have different commands and formatting, and focus on different types of development right now we'll learn the following:

- interpreted
- compiled programming languages
- query languages
- assembly language

Programming languages categories

Computers execute code in different ways depending on the type of programming language used. They may either interpret the code directly or first compile it into a format that can be executed.

In either case program is developed applications by writing instructions in a language that looks similar to English. Figure below shows an example of code written in the R programming language which is commonly used for statistical and machine learning applications. Don't worry about the content of the code at this point I just want you to have an idea of what the code looks like.

```r
# Create a numeric vector
data <- c(4, 7, 1, 8, 9, 2, 5, 6, 3)

# Calculate basic statistics
mean_value <- mean(data)
median_value <- median(data)
standard_deviation <- sd(data)

# Print the results
print(paste("Mean:", mean_value))
print(paste("Median:", median_value))
print(paste("Standard Deviation:", standard_deviation))

# Create a histogram
hist(data,
     main = "Histogram of Data",
     xlab = "Values",
     col = "lightblue",
     border = "black")
```

Above: Code written in R Language. The difference between interpreted and compiled code occurs when it comes to executing code.

Interpreted code

When you use interpreted code, the computer reads the actual instructions written by the developer as it executes the code. The computer does this by using software called an interpreter that is designed to understand a specific language. There are two subcategories of interpreted languages: scripting languages and markup languages.

Scripting languages

Scripting languages or scripted languages are often used by administrators to automate actions on the computer and for a variety of general programming tasks. Here are some examples of scripting languages:

- Perl
- R
- Python
- JavaScript
- VB script

markup languages

The second category of interpreted language is the markup language. These are languages that provide tags that you can use to mark up text documents. The two most common examples of these are the Hypertext Markup Language (HTML), which is used to create web pages, and the Extensible Markup Language (XML) which is used to exchange structured data between systems.

Below shows an example of a red page written in DML. If you notice all of the text highlighted in blue these are the tags that represent different types of formatting that should be applied to the text these tags usually appear in pairs the first tag includes the instruction inside less than and greater than brackets and the second tag is exactly the same except that there is a/ after the less than symbol any text that appears between the two tags is given the formatting indicated by the tag. Below is a Webpage written in HTML.

<HTML>
<HTTM>

This is a <B>bold text</B>.

For more certification information, visit <A HREF=https://www.certmike.com the CertMike website </A>

</BODY>

</HTML>

The next figure shows an example of a document written in HTML. This document provides a set of data elements that are going to be exchanged between systems. HTML uses tags just like those in HTML to provide names for each data element. Remember that scripting and markup languages are both examples of interpreted languages.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<students>
    <student id="101">
        <name>Jane Doe</name>
        <age>20</age>
        <major>Computer Science</major>
    </student>
    <student id="102">
        <name>John Smith</name>
        <age>22</age>
        <major>Mathematics</major>
    </student>
    <student id="103">
        <name>Linda Park</name>
        <age>21</age>
        <major>Cybersecurity</major>
    </student>
</students>
```

Compiled code in languages that use compiled code the programmer runs a tool called a compiler on their program to produce an executable file. This executable file contains instructions in machine language that carry out the programmer's instructions. When a user wishes to run the program they launch an executable file rather than the programmer's original source code examples of compiled languages are as follows:

- C and C++
- Java Go
- Go
- Julia
- FORTRAN

Exam tips: You won't find any exam questions requiring you to know how to write code in any specific language. The questions you encounter will either test you on general concepts or ask you to read pseudo-code that is written in a generic way.

In an interpreted language the computer directly executes the source code written by the developer using a program called an interpreter in a compiled language a program called a compiler must first be used to convert the source code into an executable file. Compiled code normally runs faster than interpreted code.

Specialized languages, there are two other categories of languages that we need to talk about, but for special use cases.

Interpreted languages are executed from source code using an interpreter. Interpreted languages may be either scripting languages such as Perl or Python Ruby JavaScript and vbscript or mockup languages such as HTML and XML.

Compiled languages use a compiler to convert source code into executable machine language. Compiled languages include C, C++, Java, Go, Julia, and FORTRAN.

Structured Query Language is used to interact with relational databases assembly language is written to work on a specific hardware processor.

Assembly languages

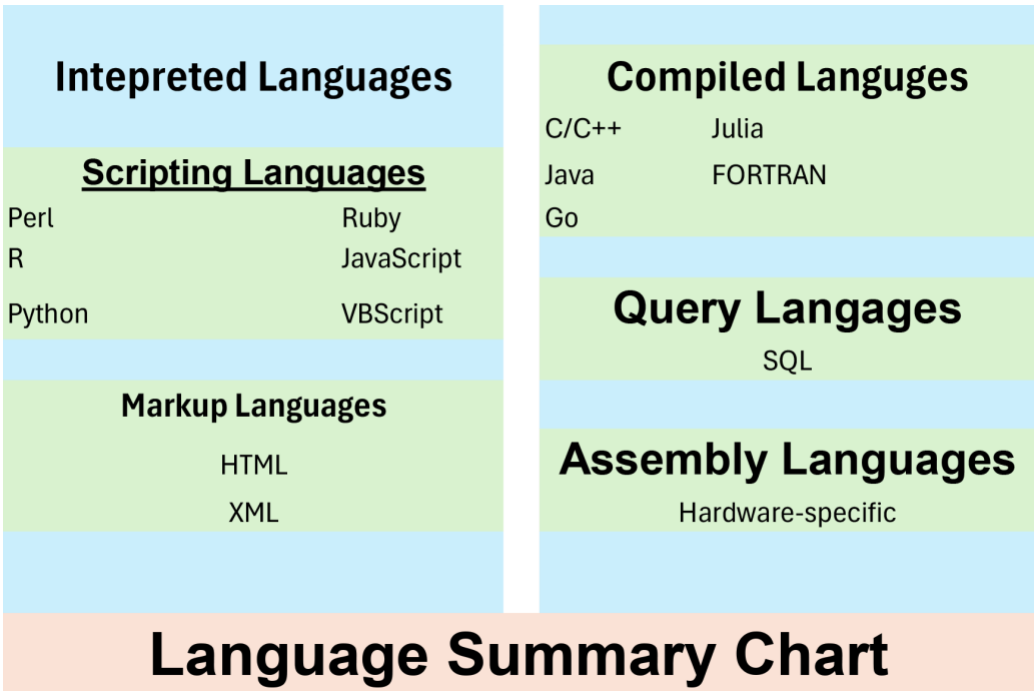
Assembly language allows programmers to write code that works directly with the hardware, bypassing compiled or interpreted languages. Each type of processor you might use has a different assembly language, but it is fairly rare to write code in assembly language for this reason (and because it is pretty challenging to use) . Assembly language isn't quite the same as machine language, but they're very close. Today, the only time you see people writing assembly language is when they're working on specialized hardware, such as embedded devices or Internet of Things devices.

Query languages

Query languages are used to ask questions of databases, and the main language in this category is the Structured Query Language, or SQL. We will cover this later in detail.

Identifying languages

We covered a lot of different categories and languages, and you need to know which language fits in which category for the exam. The summary chart below will help you remember the major languages that you might see on an exam.



Practice question 1

Which of the following languages requires A compiler to create executable code?

This question asks you to recognize which category each programming language falls into. From the knowledge you gain in this chapter, you should know that C and C++ are compiled languages. In these languages, the developer writes source code in that language and then uses a compiler to create a machine language file that can be executed.

Python and Perl are interpreted scripting languages. In those languages, the code written by a software developer is directly executed by the language's interpreter. SQL is a query language used to send commands to a relational database. The correct answer is (C) C++

Practice question 2

You are upgrading the hardware and operating system on a server that runs custom code that was created by the software development team in your organization. Which of the following types of code is most likely to need to be written for the new server?

Code that is written in assembly language is written to work on specific hardware processor. A server upgrade that changes the process is likely to require a new assembly language code that is designed to work on a new

processor. Software written in compiled languages does not need to be written. Instead the developer can take the original source code and recompile it to work on a new processor.

Similarly interpreted languages do not need to be rewritten. The System Administrator simply needs to ensure that an appropriate interpreter is installed on the new server.

Query languages such as structured query language are not hardware specific, and as long as the same database server is installed on the server, existing SQL commands should continue to execute in the same way that they did on the old server. The correct answer is **D, assembly language**

SEMtech!
Student Engagement & Mentoring in Technology

Study Guide: Chapter 26 – Programming Languages

Overview

Programming languages enable developers to write software that meets organizational goals. Different types of programming languages exist, each with unique features, execution models, and use cases.

Categories of Programming Languages

1. Interpreted Languages

- **Executed** line-by-line using an **interpreter**.
- Good for rapid development and scripting.

Subtypes:

- **Scripting Languages**
  - Automate system tasks, support general-purpose programming.
  - Examples: **Perl, Python, R, JavaScript, VBScript**
- **Markup Languages**
  - Used to **format** text or exchange data (not to perform computations).
  - Examples: **HTML, XML**

Note: Interpreted languages **do not require compilation** before execution.

---

2. Compiled Languages

- Code is **translated by a compiler** into an executable machine language file before execution.
- Tend to **run faster** than interpreted code.

Examples:

- **C, C++, Java, Go, Julia, FORTRAN**

---

3. Query Languages

- Used to **interact with databases**, especially **relational databases**.
- Not hardware-dependent.

Example:

- **SQL (Structured Query Language)**

---

4. Assembly Language

- Low-level language that **interacts directly with hardware**.
- Hardware-specific; often used for **embedded systems and IoT**.
- Must be rewritten for different processors.

Key Differences Summary Table

Category	Execution Model	Examples	Hardware Dependency
Interpreted	Line-by-line via interpreter	Python, R, Perl, HTML, XML	❌
Compiled	Pre-compiled to executable	C, C++, Java, Go, Julia	❌
Query Language	Database commands	SQL	❌
Assembly Language	Direct hardware interaction	Varies by processor	✅

Visual Reference

- The **R code example** in the image shows a **scripting language**.
- It calculates mean, median, and standard deviation, and creates a histogram.
- R is an **interpreted scripting language**, ideal for statistical tasks.

SEMtech!
Student Engagement & Mentoring in Technology

Exam Tips

- You **won't need to code**, but you **must identify** language types.
- Understand the **execution model** (compiled vs interpreted).
- Know which languages are **hardware dependent** (like assembly).
- **Focus on application** rather than syntax.

Practice Questions Recap

Q1: Which language requires a compiler?

- ✅ **Answer: C++**
- Why? It's a **compiled language**.

Q2: Which code must be rewritten for new hardware?

- ✅ **Answer: Assembly Language**
- Why? Assembly is **hardware-specific**.

SEMtech Insights

Drawing from Marshall McLuhan's notion that *"the medium is the message"*, understanding programming languages is key not just to what software does—but *how* it communicates with machines, and *how* that shapes the developer's logic and outcomes.

Final Tips for Mastery

- Match each **language** with its **category** and **use case**.
- Remember: Interpreters **execute**, compilers **translate**.
- Think in terms of **abstraction layers**: from high-level (e.g., Python) to low-level (e.g., Assembly).
- Apply the **Postman principle**: Don't just learn tools—understand their **cultural and operational impact**.

SEMtech!
Student Engagement & Mentoring in Technology

Study Review Questions – Chapter 26: Programming Languages

Multiple Choice (20 Questions)

Choose the **best answer** for each question.

1. Which of the following is a compiled programming language?

A. Python
B. C++
C. HTML
D. XML

2. What is the purpose of a compiler?

A. To run SQL queries
B. To translate code into machine language before execution
C. To interpret scripts line-by-line
D. To manage database transactions

3. Which language would you use to develop a webpage?

A. Python
B. C++
C. HTML
D. SQL

4. Which of the following is a scripting language?

A. C
B. Go
C. Perl
D. FORTRAN

5. Which programming language is commonly used for statistical analysis and machine learning?

A. HTML
B. Java

C. R
D. C++

6. What is the correct category for SQL?

A. Scripting language
B. Query language
C. Compiled language
D. Assembly language

7. What is the main characteristic of interpreted languages?

A. Code is converted to an executable file
B. Code runs directly through an interpreter
C. Code is used for web formatting only
D. Code cannot interact with hardware

8. Which of the following is NOT an interpreted language?

A. Python
B. JavaScript
C. C
D. VBScript

9. Which language is best suited for writing software that interacts directly with the hardware?

A. SQL
B. Java
C. Assembly language
D. JavaScript

10. What are HTML and XML examples of?

A. Query languages
B. Compiled languages
C. Scripting languages
D. Markup languages

11. Which language is compiled and used frequently in system-level programming?

A. Python
B. HTML

C. C
D. XML

12. What is a key benefit of compiled languages over interpreted ones?

A. They are easier to read
B. They run faster
C. They require fewer lines of code
D. They support more languages

13. What must be installed on a server to run interpreted code?

A. Operating system
B. Compiler
C. Interpreter
D. Processor firmware

14. Which language allows for automated actions and general programming tasks?

A. SQL
B. HTML
C. Java
D. Scripting languages

15. What kind of code would need to be rewritten if a server's processor is upgraded?

A. Compiled language code
B. Query language code
C. Assembly language code
D. Interpreted language code

16. Which of these is not a scripting language?

A. Perl
B. JavaScript
C. C++
D. Python

17. What does XML primarily do?

A. Format graphics
B. Perform calculations

C. Exchange structured data
D. Compile code

18. Which tool transforms source code into an executable file?

A. Interpreter
B. Compiler
C. Parser
D. Debugger

19. Java is unique because:

A. It cannot be compiled
B. It is a markup language
C. It is a compiled language with interpreted capabilities
D. It is only used in databases

20. Which of these is an interpreted language used in browsers?

A. SQL
B. Java
C. JavaScript
D. Go

Short Answer (5 Questions)

21. What is the primary difference between compiled and interpreted programming languages?

22. Why is assembly language rarely used in most software development projects today?

23. Name two markup languages and explain their use.

24. Explain the role of a query language in software development.

25. Describe a scenario where a scripting language is the best choice for solving a problem.

SEMtech!
Student Engagement & Mentoring in Technology

Vocabulary – Chapter 26: Programming Languages

Term	Definition
Programming Language	A set of rules and syntax used by software developers to write instructions that a computer can understand and execute.
Interpreted Language	A type of programming language in which the source code is executed line-by-line by an interpreter, without prior compilation.
Interpreter	A software program that reads and executes code directly from the source file, translating one line at a time during runtime.
Scripting Language	A subtype of interpreted languages used for automating tasks and general programming (e.g., Python, Perl, JavaScript, VBScript).
Markup Language	A type of interpreted language that uses tags to format and structure documents (e.g., HTML, XML).
HTML (Hypertext Markup Language)	A markup language used to create and format content for web pages.
XML (Extensible Markup Language)	A markup language used to store and transport data in a structured format.
Compiled Language	A programming language that must be translated into machine code by a compiler before it can be executed.
Compiler	A software tool that converts source code into machine code or an executable file before the program runs.
Executable File	A file that contains machine code generated by a compiler, ready to be run by the computer's operating system.
Machine Code	The lowest-level code, consisting of binary instructions that a computer's processor can directly execute.
Query Language	A specialized programming language used to retrieve, modify, and interact with data in a database (e.g., SQL).
SQL (Structured Query Language)	A query language used to perform tasks like querying, updating, and managing data in relational databases.
Assembly Language	A low-level programming language that is hardware-specific and closely related to machine code.
Low-Level Language	A language that operates very close to the hardware level, providing little abstraction from the computer's architecture.
High-Level Language	A language that provides more abstraction from hardware, using human-readable syntax (e.g., Python, Java).

SEMtech!
Student Engagement & Mentoring in Technology

Term	Definition
Source Code	The original code written by a programmer in a human-readable programming language.
Syntax	The set of rules that defines the combinations of symbols considered correctly structured in a programming language.
Tag	A markup language element that indicates formatting or data structure, typically enclosed in angle brackets (e.g., `<b>`, `</b>`).
Hardware Dependency	The extent to which code must be tailored to specific hardware; high in assembly languages, low in high-level languages.
Embedded System	A computer system designed to perform specific tasks, often using assembly language due to hardware constraints (e.g., IoT devices).
Execution	The process by which a computer performs instructions written in a program.
Run-Time	The period during which a program is running (executing).
Automation	The use of scripts or software to perform tasks without manual intervention.
Data Formatting	The process of organizing and structuring data for display or storage, often using markup languages.
Relational Database	A type of database that stores data in tables with relationships between them; typically accessed using SQL.
Pseudo-Code	A simplified, language-agnostic way of writing program logic that resembles real code but is not meant for execution.
General-Purpose Language	A language that can be used across various domains and applications, not limited to a specific task (e.g., Python, Java).