The structure of "Java programming" in regular textbooks typically follows a progression from fundamental concepts to advanced topics, building upon each other. Here is a breakdown of the major parts, presented in the common order you would find in a comprehensive textbook.

**Part 1: Foundations and Basic Syntax**

This part is all about getting the student to write their first program and understand the absolute basics of the language.

* **Chapter 1: Introduction to Java**
  + **What is Java?**: History, key characteristics (Simple, Object-Oriented, Portable, Secure, Robust, etc.).
  + **The Java Platform (JRE & JDK)**: Explanation of the Java Virtual Machine (JVM), Java Runtime Environment (JRE), and Java Development Kit (JDK).
  + **Setting up the Environment**: Installing the JDK, setting the JAVA\_HOME and PATH variables.
  + **First Java Program ("Hello, World!")**: Writing, compiling (javac), and running (java) a simple program. Anatomy of a basic class and the main method.
* **Chapter 2: Basic Syntax and Variables**
  + **Identifiers & Keywords**: Rules for naming classes, variables, and methods.
  + **Data Types**: Primitive data types (int, double, char, boolean, etc.) and their sizes.
  + **Variables**: Declaring and initializing variables.
  + **Literals**: Writing literal values (e.g., 10, 3.14f, 'A', true).
  + **Basic Console I/O**: Using System.out.println() for output and Scanner class for input.
* **Chapter 3: Operators and Expressions**
  + **Arithmetic Operators** (+, -, \*, /, %)
  + **Relational and Conditional Operators** (>, <, ==, !=, &&, ||)
  + **Assignment Operators** (=, +=, -=, etc.)
  + **Type Casting**: Implicit and explicit casting.

**Part 2: Control Flow and Fundamental Structures**

This part teaches how to control the flow of program execution and work with collections of data.

* **Chapter 4: Control Flow Statements**
  + **Conditional Statements**: if, if-else, if-else-if ladder, and switch statements.
  + **Looping Statements**: for loop, while loop, do-while loop.
  + **Branching Statements**: break, continue, and return.
* **Chapter 5: Arrays**
  + **Declaring and Instantiating Arrays**.
  + **Accessing and Modifying Array Elements**.
  + **Multi-dimensional Arrays**.
  + **Common Operations**: Looping through arrays (often introducing the *for-each* loop here).

**Part 3: Object-Oriented Programming (OOP) - The Core of Java**

This is the most critical part of any Java textbook, where the paradigm of the language is explained in depth.

* **Chapter 6: Introduction to Classes and Objects**
  + **Classes vs. Objects**: The blueprint vs. the instance.
  + **Defining a Class**: Fields (attributes) and methods (behaviors).
  + **Constructors**: Default and parameterized constructors.
  + **The**this**Keyword**.
* **Chapter 7: Core OOP Concepts**
  + **Encapsulation**: Bundling data and methods; using private access modifier and public *getter/setter* methods.
  + **Inheritance**: The extends keyword, super keyword, method overriding, the Object class.
  + **Polymorphism**: Method overloading (compile-time) and method overriding (runtime).
  + **Abstraction**: Using abstract classes and methods.
* **Chapter 8: Advanced Class Features**
  + static**keyword**: Static variables, static methods, and static blocks.
  + final**keyword**: Final variables, methods, and classes.
  + **Packages**: Organizing classes, import statements.
  + **Access Modifiers**: public, protected, default (package-private), private.
* **Chapter 9: Interfaces and Abstract Classes**
  + **Defining and Implementing Interfaces** (implements keyword).
  + **Default and Static Methods in Interfaces** (Java 8+ features).
  + **Comparing Interfaces and Abstract Classes**: When to use which.

**Part 4: Exception Handling and Built-in APIs**

This part deals with making programs robust and introduces essential libraries.

* **Chapter 10: Exception Handling**
  + **What are Exceptions?**: Checked vs. Unchecked exceptions.
  + try-catch-finally blocks.
  + throw and throws keywords.
  + **Creating Custom Exceptions**.
* **Chapter 11: The Java Collections Framework**
  + **Core Interfaces**: Collection, List, Set, Queue, Map.
  + **Common Implementations**:
    - List: ArrayList, LinkedList
    - Set: HashSet, LinkedHashSet, TreeSet
    - Map: HashMap, LinkedHashMap, TreeMap
  + **Iterating through Collections**: Iterators, for-each loop.
  + **Comparable and Comparator** interfaces for sorting.
* **Chapter 12: Common Java APIs**
  + **The**String**Class**: Immutability, common methods.
  + **The**StringBuilder**and**StringBuffer classes.
  + **Wrapper Classes** (Integer, Double, etc.) and Autoboxing/Unboxing.
  + **Utility Classes**: Math, Arrays.

**Part 5: Advanced Topics**

These topics are covered in more advanced or later chapters of textbooks, preparing students for real-world development.

* **Chapter 13: Generics**
  + **Why Generics?**: Type-safety and eliminating casts.
  + **Generic Classes and Methods**.
  + **Bounded Type Parameters**.
* **Chapter 14: Input/Output (I/O) Streams**
  + **Byte Streams**: InputStream, OutputStream (e.g., FileInputStream).
  + **Character Streams**: Reader, Writer (e.g., FileReader).
  + **Buffered Streams** for efficiency.
  + **Serialization and Deserialization**.
* **Chapter 15: Concurrency (Multithreading)**
  + **Creating Threads**: Extending Thread class vs. Implementing Runnable interface.
  + **Thread Lifecycle**.
  + **Synchronization** (synchronized keyword) to handle thread safety.
  + **Concurrent Collections** (introduction).
* **Chapter 16: Introduction to Functional Programming (Java 8+)**
  + **Lambda Expressions**.
  + **Functional Interfaces**.
  + **Stream API**: Intermediate and terminal operations (filter, map, collect, etc.).
* **Appendix/Additional Topics**: Often includes an introduction to **Modules** (Java 9+), **Networking**, **Java Database Connectivity (JDBC)**, or **Unit Testing with JUnit**.