Course Overview

CompuScholar’s **Java Programming** curriculum is a one-year (two-semester) course that teaches students to code in the Java language. It is aligned to numerous state and national standards for courses such as “Computer Programming I” or similar titles. For details, please see our State Alignments and course description pages:

https://www.compuscholar.com/schools/standards/states/

https://www.compuscholar.com/schools/courses/java/

Other introductory programming courses are not required. Students merely need to have typical computer usage skills prior to starting this course.

**Java Programming** can also be used as an Advanced Placement (AP) “Computer Science A” course. For details, please see our Syllabus and Planner guide for AP CS A teachers.

Course Material

The course consists of the following student-facing elements:

- **Instructional Videos** – optional (not required), but enjoyed by many students as an audio-visual introduction and reinforcement of the lesson topics.
- **Lesson Text** – required reading, contains full topic details and live coding exercises
- **Quizzes and Exams** – multiple-choice and automatically graded by our system
- **Chapter Activities** – hands-on projects, submitted for a grade

Teachers additionally have access to:

- **Teacher’s Guides** – for each lesson, with suggested classroom discussion questions
- **Quiz and Exam Answer Keys** – PDFs for quick reference
- **Activity Solution Guides** – fully coded activity solutions for each chapter activity

866-604-2357 (office)
866-332-8005 (fax)  
www.CompuScholar.com
Programming Environment and Device Requirements

CompuScholar provides an in-browser Java coding environment. This online feature may be used by students to complete all exercises and activities in all “core” chapters. When using our online coding environment:

- No local software installation is needed.
- All activities can be completed from any web browser on any device (including Chromebooks and tablets).

Later, optional chapters contain a mixture of activities. Teachers may select any of these topics for students as desired to meet specific state requirements. Some optional activities can be done in CompuScholar’s online environment, while others are completed using an external IDE.

When needed, we recommend a locally installed JDK and Eclipse platform for an external IDE (see chapters 28 and 29 for instructions). Teachers may also select any other locally installed or online IDE. **Device requirements for your optional, external IDE depend on the IDE selected.**

Project Grading

Each chapter normally contains one or more hands-on, graded activities. The activities in all “core” chapters are fully auto-graded by our system. Teachers have complete control over the auto-graded results.

Some activities in later, optional chapters are free-form (creative) or completed in an external IDE. The teacher is responsible for grading those creative or external projects.

Course Navigation

**Chapters 1 - 24** are considered “core” curriculum that cover the required topics in most state computer science courses. Students are generally advised to complete these chapters, in sequence. If a specific topic is not required by your state or appropriate for your classroom (e.g., recursion or sorting), then teachers may choose to skip those chapters.
Chapters 25 – 33 contain optional topics that may not be required in every state. Teachers are encouraged to review the optional material and direct students to any topic(s) that are a good fit for their local classrooms. Some optional chapters require use of an external IDE and/or will be graded by the teacher.

Supplemental Chapters 1 – 4 contain a variety of topics that may be required by individual states to satisfy requirements for career exploration, computing in modern society, computer networking and other enrichment topics. Teachers may optionally select any of these topics for students, time permitting.

Please refer to the specific computer science requirements for your state when selecting optional or supplemental topics. Our State Alignments page contains guidance for many states, or you can contact CompuScholar for additional help.

Course Planner

The following pages contain a suggested timeline for completing course content over two semesters. A typical school year consists of 36 calendar weeks or 180 days of school. After completing the “core” content, most classes will have approximately 6-8 weeks left in the school year to explore optional and supplemental topics. Teachers may direct students to any appropriate topics, time permitting.

Each “day” listed below represents one typical day or class period of 45 – 60 minutes. In most cases, we anticipate students will complete one lesson per day (including the quiz), 1 day per lab, and 1 day per chapter test. Some classes may move faster or slower than the suggested pace.
Semester 1 Timeline

<table>
<thead>
<tr>
<th>Days</th>
<th>CompuScholar Chapter</th>
<th>Notes</th>
</tr>
</thead>
</table>
| 6    | **Chapter 1: Computing Concepts**  
* Evolution of Computers  
* Computer Hardware  
* Computer Software  
* Computer Ethics  
* Computer Security | Common curricular requirements |
| 6    | **Chapter 2: Getting Started with Java**  
* Common Programming Languages  
* The Java Platform  
* Writing Your First Program  
* Help and Reference Documentation  
ACTIVITY: Shopping List | Online, auto-graded |
| 5    | **Chapter 3: Data Types and Variables**  
* Primitive Data Types  
* Variables  
* Printing Data  
ACTIVITY: Treasure Map | Online, auto-graded |
| 5    | **Chapter 4: Working with Numbers**  
* Simple Math Operations  
* Compound Assignments and Shortcuts  
* Type Casting and Truncation  
ACTIVITY: Magic Math | Online, auto-graded |
| 8    | **Chapter 5: Introducing Objects**  
* Java Classes  
* Reference Variables and Strings  
* Calling Methods  
* Properties and Constructors  
* Static and Overloaded Methods  
* User Input with Scanner  
ACTIVITY: Sketch Robot | Online, auto-graded |
| 5 | **Chapter 6: Working with Strings**  
* Comparing Strings  
* Common String Operations  
* Formatting and Building Strings  
ACTIVITY: String Theory | Online, auto-graded |
|---|---|
| 5 | **Chapter 7: Numbering Systems and Java Math**  
* Java Wrapper Classes & Numeric Conversion  
* Numbers in Binary, Octal and Hex  
* Java Math Class  
ACTIVITY: Math Factory | Online, auto-graded |
| 6 | **Chapter 8: Logic and Decision-Making**  
* Logical Expressions and Relational Operators  
* Making Decisions with if()  
* Using "else-if" and "else"  
* The "switch" Statement  
ACTIVITY: Banking System | Online, auto-graded |
| 5 | **Chapter 9: More Complex Logic**  
* Comparing Objects and References  
* Compound Expressions  
* Boolean Algebra and Truth Tables  
ACTIVITY: Wild Card | Online, auto-graded |
| 5 | **Chapter 10: Handling Exceptions**  
* Understanding Exceptions  
* Catching Exceptions  
* Validating User Input  
ACTIVITY: Calculator Madness | Online, auto-graded |
| 4 | **Chapter 11: Debugging**  
* Finding Runtime Errors  
* Debugger Concepts  
ACTIVITY: Bug Hunt | Online, auto-graded |
| 6 | **Chapter 12: Iteration**  
* For Loops  
* While Loops  
* Continue, Break and Return  
* Nested Loops  
ACTIVITY: Fun Factorials | Online, auto-graded |
|---|---|---|
| 6 | **Chapter 13: Algorithms**  
* Designing with Flowcharts  
* Writing Pseudocode  
* Common Mathematical Algorithms  
* Common String Algorithms  
ACTIVITY: Meal Time | Online, auto-graded |
| 7 | **Chapter 14: Creating Java Classes**  
* Object-Oriented Concepts  
* Defining Classes and Packages  
* Class Properties  
* Constructors  
* Class Methods  
ACTIVITY: Dog House | Online, auto-graded |
| 7 | **Chapter 15: Working with Methods**  
* Documentation and Design  
* Variable Scope and Access  
* Data Encapsulation  
* Method Overloading  
* Object Interfaces  
ACTIVITY: Let's Go Racing! | Online, auto-graded |
| 5 | **Chapter 16: Static Concepts**  
* Static Properties  
* Static Methods  
* Static, Object and "this" References  
ACTIVITY: Art School | Online, auto-graded |
| 4 | **Chapter 17: Mid-Term Project**  
* Introducing the "Remote Control" Project  
ACTIVITY: Creating the Schedule  
ACTIVITY: Building a Television  
ACTIVITY: Defining the Remote | Online, auto-graded |
| 95 | **Approximate Days, Semester 1, “core” chapters** | |
# Semester 2 Timeline

<table>
<thead>
<tr>
<th>Days</th>
<th>CompuScholar Chapter</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td><strong>Chapter 18: 1D Arrays</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>* Array Concepts</td>
<td>Online, auto-graded</td>
</tr>
<tr>
<td></td>
<td>* Array Traversal</td>
<td></td>
</tr>
<tr>
<td></td>
<td>* Iterators and Enhanced for() loops</td>
<td></td>
</tr>
<tr>
<td></td>
<td>* Array Algorithms</td>
<td></td>
</tr>
<tr>
<td></td>
<td>* More Array Algorithms</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACTIVITY: Whack-A-Mole</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td><strong>Chapter 19: Lists and ArrayLists</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>* Java Lists</td>
<td>Online, auto-graded</td>
</tr>
<tr>
<td></td>
<td>* ArrayLists</td>
<td></td>
</tr>
<tr>
<td></td>
<td>* Iterators and Enhanced for() Loops</td>
<td></td>
</tr>
<tr>
<td></td>
<td>* Algorithms with ArrayLists</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACTIVITY: Train Yard Jumble</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td><strong>Chapter 20: Searching and Sorting</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>* Bubble Sort</td>
<td>Online, auto-graded</td>
</tr>
<tr>
<td></td>
<td>* Selection Sort</td>
<td></td>
</tr>
<tr>
<td></td>
<td>* Insertion Sort</td>
<td></td>
</tr>
<tr>
<td></td>
<td>* Sequential and Binary Searches</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACTIVITY: Ducks in a Row</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td><strong>Chapter 21: 2D Arrays</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>* 2D Arrays</td>
<td>Online, auto-graded</td>
</tr>
<tr>
<td></td>
<td>* Traversal and Ordering</td>
<td></td>
</tr>
<tr>
<td></td>
<td>* Array of Arrays</td>
<td></td>
</tr>
<tr>
<td></td>
<td>* 2D Array Algorithms</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACTIVITY: Gold Rush</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td><strong>Chapter 22: Inheritance</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>* Superclass and Subclass Concepts</td>
<td>Online, auto-graded</td>
</tr>
<tr>
<td></td>
<td>* Subclass Constructors</td>
<td></td>
</tr>
<tr>
<td></td>
<td>* Using Superclass and Subclass References</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACTIVITY: Lab Rats</td>
<td></td>
</tr>
</tbody>
</table>

866-604-2357 (office)  
866-332-8005 (fax)  
www.CompuScholar.com
### Chapter 23: Polymorphism
- Overriding Superclass Methods
- Abstract Classes and Methods
- Using Superclass Features with "super"
- The "Object" Superclass

**ACTIVITY: Social Ladder**

| Online, auto-graded |

### Chapter 24: Recursion
- Recursion
- Recursive Binary Search
- Merge Sort

**ACTIVITY: File Explorer**

| Online, auto-graded |

| 42 | **Approximate Days in Semester 2** (all “core” chapters complete at this point) |

Classes who complete the first 24 chapters at this point have spent approximately 136 days and completed all “core” requirements. Remaining class time should be spent in any desired, teacher-selected topics from Chapters 25 – 33 or the Supplemental Chapters.

Please see below for information on the **optional chapters and Supplemental topics**.
The following table suggests the timeline needed for each **optional or supplemental chapter**, along with notes as to the programming environment and grading approach. There are more “optional” chapters available than students can complete in a single year, so teachers can pick topics as time permits!

<table>
<thead>
<tr>
<th>Days</th>
<th>CompuScholar Chapter</th>
<th>Notes</th>
</tr>
</thead>
</table>
| 5    | **Chapter 25: File Access**  
* Data Streams  
* Reading and Writing Text Data  
* Reading and Writing Binary Data  
ACTIVITY: Address CSV | Online, auto-graded |
| 5    | **Chapter 26: Object Composition and Copying**  
* Functional Decomposition  
* Composite Classes  
* Copying Objects  
ACTIVITY: Designing a Composite Class | Document submission, teacher graded |
| 10-15| **Chapter 27: Team Project**  
* Design Processes and Teamwork  
* Requirements and Design Documents  
ACTIVITY: Team Project Requirements  
ACTIVITY: Project Design  
ACTIVITY: Team Project Implementation  
* Testing Your Code  
ACTIVITY: Team Project Testing | CompuScholar online environment or external IDE, teacher-graded project |
| 3    | **Chapter 28: Running Java Locally**  
* Installing the JDK  
* Local Source Code  
* Building and Running from the Command Line | “How-to” chapter to create local development environment |
| 4    | **Chapter 29: The Eclipse IDE**  
* Introducing Eclipse  
* Eclipse Java IDE Walk-Through  
* Creating an Eclipse Project  
* The Eclipse Debugger | “How-to” chapter to install and use a local IDE |
| 6 | **Chapter 30: Graphical Java Programs**  
* Java Swing  
* Creating a Simple Window  
* Event-Driven Programming  
* Layout Managers  
ACTIVITY: Phone Dialer | Requires external IDE (e.g., Eclipse) with Java Swing support. Teacher-graded projects. |
|---|---|
| 5 | **Chapter 31: Swing Input Controls**  
* Text and Numeric Input  
* List Input  
* Option Input  
ACTIVITY: Pizza Place |
| 5 | **Chapter 32: Vector and Bitmap Images**  
* Screen Coordinates  
* Drawing Shapes  
* Drawing Images  
ACTIVITY: Sky Art |
| 4 | **Chapter 33: Program Efficiency**  
* Algorithm Performance (Big-O)  
* Measuring Sorting Efficiency  
ACTIVITY: Comparison of Sorting Algorithms | External IDE, teacher-graded project |
| 12 | **Supplemental Chapter 1: Enrichment Topics** | See individual lessons and activities for programming environment and grading approach. |
| 8 | **Supplemental Chapter 2: Software and Industry** | Offline work, teacher-graded projects |
| 4 | **Supplemental Chapter 3: Computers and Modern Society** | Offline work, teacher-graded projects |
| 6 | **Supplemental Chapter 4: Computer Networking** | Offline work, teacher-graded projects |