CodeHS

Introduction to Java Semester B Course Syllabus

The CodeHS Intro to Java Semester A and Semester B courses are aligned to all College Board seven curriculum requirements extensively as shown in the table below. However, some more advanced topics, like recursion and the various sorting algorithms have been omitted. Therefore, these courses do NOT prepare students for the AP CS A exam. The curriculum requirements laid out by the College Board are:

- CR1: Teaches students to design and implement computer-based solutions to problems.
- CR2a: Teaches students to use and implement commonly used algorithms.
- CR2b: Teaches students to use commonly used data structures.
- CR3: Teaches students to select appropriate algorithms and data structures to solve problems.
- CR4: Teaches students to code fluently in an object-oriented paradigm using the programming language Java.
- CR5: Teaches students to use elements of the standard Java library.
- CR6: Includes a structured-lab component composed of a minimum of 20 hours of hands-on lab experiences.
- CR7: Teaches students to recognize the ethical and social implications of computer use.

Course Overview and Goals

The CodeHS Intro to Java Semester B course is a semester-long course designed to help students master the basics of Java. It is the second course in a two course sequence and should be completed after the Intro to Java Semester A course. All learning materials and resources teachers and students need for a successful semester-long Java course can be found on the CodeHS website.

Learning Environment: The course utilizes a blended classroom approach. The content is fully web-based, with students writing and running code in the browser. Teachers utilize tools and resources provided by CodeHS to leverage time in the classroom and give focused 1-on-1 attention to students. Each unit of the course is broken down into lessons. Lessons consist of video tutorials, short quizzes, example programs to explore, and written programming exercises, adding up to over 50 hours of hands-on programming practice in total. [CR6] Several units have free response questions that have students consider the applications of programming and incorporate examples from their own lives.

Programming Environment: Students write and run Java programs in the browser using the CodeHS editor. [CR1] [CR6]

Quizzes: At the end of each unit, students take a summative multiple choice unit quiz that assesses their knowledge of the Java concepts covered in the unit. Included in each lesson is a formative short multiple choice quiz.

More information: Browse the content of this course at https://codehs.com/course/8255

Prerequisites

Students should complete the Intro to Java Semester A course prior to starting this course. It is also expected that students know basic English and algebra. Students should be comfortable with functions and function notation, such as *f(x) = x + 2* and *f(x) = g(h(x))*.

Course Breakdown

Unit 1: Classes and Object Oriented Programming (8 weeks)

Browse the full content of this unit at https://codehs.com/library/course/8255/module/12033

Objectives / Topics Covered [CR1] [CR4] [CR5]	● Using classes as a client ● Classes vs Objects ● Class methods ● Instance variables ● Constructors ● Visibility ● Information hiding ● `this` ● `static` ● `super` ● The Java `Math` class and methods `(abs, pow, sqrt, sin, cos)` ● Creating random values with the CodeHS `Randomizer` class ● Designing classes ● Creating classes ● Getter and setter methods ● Inheritance

	● Method overloading ● Local variables and scope ● Comparing objects vs primitive types ● Abstract classes ● packages ● Polymorphism ● Interfaces ● Modifying classes to implement interfaces ● `Object` is the superclass of all classes

Assignments / Labs
[CR1] [CR4] [CR5] [CR6]
	● Several programming exercises to master each of the topics above. 35 exercises in total.
	● Examples
	○ Using the Student Class
	In this program we have a `Student` class in Student.java and a tester program at StudentTester.java.
	If you open up StudentTester.java you will see we have a bit of code there already. We’ve created two new students, Alan and Ada.
	We create a `Student` instance by calling the constructor and passing in the first name, last name, and grade level as an integer.
	Your task is to create a `Student` with your information! Once you have created the `Student`, print it out to the console.
	○ Design and implement a `Fraction` class from scratch, including a constructor, getter and setter methods, a `toString` method, and methods to add, subtract, and multiply by other `Fraction` objects.
	○ Implement a `RockPaperScissors` class with a `getWinner(String user, String computer)` method that allows a user to play the game Rock, Paper, Scissors against a computer that picks moves randomly.
	○ Add an abstract method to an existing `Shape` class called `public abstract double getPerimeter()` and implement this method on each of the Shape subclasses, `Square`, `Rectangle`, `Pentagon`, and `Circle`
	○ Fun with Solids
	Given the `Solid` abstract class, extend it with:
	`Pyramid`
	`Cylinder`
	`RectangularPrism`
	`Sphere`
	Make sure to create the constructor, `volume` and

	`surfaceArea` methods for each class (the `Math` class will come in handy). Also extend `RectangularPrism` with `Cube`. ○ Modify the `Fraction` class to implement the `Comparable` interface

Unit 2: Data Structures (7 weeks)

Browse the full content of this unit at https://codehs.com/library/course/8255/module/12034

Objectives / Topics Covered [CR1] [CR2b] [CR3] [CR4] [CR5]	● Declaring and initializing arrays ● Constructing `ArrayLists` ● Indexing into arrays/`ArrayLists` ● Iterating over arrays/`ArrayLists` ● Getting the length of an array/`ArrayLists` ● `ArrayIndexOutOfBoundsException` ● `IndexOutOfBoundsException` ● Understand array variables are references to objects ● Arrays/`ArrayLists` as parameters and return values ● Inserting and deleting array/`ArrayList` elements ● Wrapper classes - `Double`, `Integer` ● Storing objects/primitives in arrays vs. `ArrayLists` ● Numerical representations of integers ○ Representations of non-negative integers in different bases ○ Implications of finite integer bounds ● The `List` interface ● Declaring and initializing 2-D rectangular arrays ● Using nested loops to iterate through 2-D arrays ● row-major order ● Students reminded about indices starting at 0 ● Constructing, adding to, and iterating through `HashMaps` ● Deciding which data structures to use when designing a class

Assignments / Labs
[CR1] [CR2b] [CR3] [CR4] [CR5] [CR6]
	● Several programming exercises to master each of the topics above. 23 exercises in total.
	● Examples
	○ Write a method that returns the index of the minimum value in an array
	○ Write a method that returns the first value in an `ArrayList`
	○ See how an `ArrayList` works under the hood. Write an `ExpandingArray` class that stores an array as an instance variable and supports the methods
	`public void add(int index, int element)`
	`public void add(int element)`

	`public int remove(int index)`
	`public int size()`
	`public String toString()`
	○ Write the method `public int sumRow(int[][] matrix, int row)` Which sums row row in the 2D array called matrix.
	○ Explore and add to the code for a `BlackJack` game with a `Card` class, `Deck` class, `Hand` class, and `BlackJack` class
	○ Implement the game Battleship with several incremental checkpoints
	■ Implement the `Ship` class
	■ Implement the `Location` class
	■ Implement the `Grid` class
	■ Implement adding a `Ship` to a `Grid`
	■ Design and implement the `Player` class
	■ Design and implement the `Battleship` class
	■ Add extra features to the game

Unit 3: Final Project (4 weeks)

Browse the full content of this unit at https://codehs.com/library/course/8255/module/12035

Objectives / Topics Covered [CR1] [CR4]	● Allow students to think creatively about the applications of the concepts covered in the course ● Scoping a project ● Designing an application from scratch ● Incremental development ● Testing ● Debugging
Assignments / Labs [CR1] [CR4] [CR6]	● Brainstorm ideas for a final project ● Plan out milestones for incremental development ● Design the different classes you will create for this project ● Create your final product

Unit 4: Optional Supplemental Materials

Objectives / Topics Covered [CR1] [CR4] [CR7]	● Extra practice with Java concepts ○ String processing ○ Recursion ○ Designing Classes ○ Arrays and `ArrayLists` ○ Searching and sorting algorithms ● File reading / writing ● The Java `Scanner` class ● The Java `BufferedReader` and `BufferedWriter` classes

	● Running Java programs outside of the browser ● Running Java programs from the command line ● The Java `main` method ● Computing in Context ○ Understand computer ethics such as acceptable use policies, copyright, intellectual property, privacy, and the implications of developing software used by real people in real life situations
Assignments / Labs [CR1] [CR4] [CR6] [CR7]	● Several additional exercises and advanced projects covering the topics listed above