Beyond CS Camp

Summer Class

Java Programming /AP Computer Science Preview

Description

Java is one of the top programming languages that runs across multiple platforms and is used for developing a wide range of applications. With its strong logic and clear structure, Java becomes the top Object-oriented programming language. High school AP Computer Science A is about using Java language to solve problems.

In this course, students will be introduced to critical thinking for developing efficient computer programs such as functional abstraction and OOP paradigm. Major topics include basic data types, conditionals, loops, methods, objects and classes, abstraction and encapsulation, polymorphism and inheritance, array and ArrayList, input and output, file operations, exception and error handling, graphic user interfaces, basic algorithms, and, etc. Upon completion, students will be able to design, code, test, and debug multi-class JAVA programs.

Course Objectives

Upon completion of this course, students will:

1. Develop common programming concepts and techniques universal to all programming languages.
2. Be proficient in using the basic components of the Java programming language.
3. Be able to set up Java environment and use the Eclipse IDE to develop Java applications
4. Define both primitive and object data types and use variable expressions in programs to compute numeric and string operations.
5. Define, analyze, and implement the procedures that involve decision making and repetitions by using Java conditional and loop control structures.
6. Apply the techniques of procedural decomposition to organize a program into computational and functional modules.
7. Design and implement functions and methods that can be used as program units and demonstrate the way parameters are passed in such functions and methods.
8. Understand the major object-oriented programming concepts, such as encapsulation, abstraction, inheritance, and polymorphism. And be able to apply these concepts in designing Java classes.
9. Apply the concept of Java interface to categorize objects by behaviors and know the difference of static and dynamic binding.

10. Be able to design and implement user interactive programs by using console input and output mechanism.
11. Be able to perform file I/O operations. Know how to process data that stores in files and to write information to files.
12. Design and implement programs that interact with the user through Graphic User Interface. Know how to use different layouts and GUI components, such as frame, panel, button, text field, textbox, and so on.
13. Implement some basic algorithms that use array or ArrayList as the data structure for solving problems.
14. Be able to design, develop, test, and debug multi-class Java programs.
15. Be able to apply the exception and error handling techniques to design more robust programs.
16. Be able to use Java API documentation to check the definition and usage of the standard Java packages, interfaces, and classes.

Unit of Study

Class #	Lesson	Topic
1	1	Introduction to Java
	2	Java Fundamentals
2	3	Java API and Some Standard Classes
3	4	Making Decisions
4	5	While Loops
5	6	Methods
6	7	For Loops
	8	Arrays
7	9	String Class
8	10	Basic Algorithms, Recursions
9	11	Classes and Objects
10	12	Inheritance
11	13	Polymorphism and Interface
12	14	ArrayList, 2D Array
13	15	Exceptions and Error Handling
	16	File Operations
14	17	Graphic User Interface
15	18	Final Project

Course Content

1. Introduction and Set up
    a. Understanding programming
    b. History of Java
    c. Main features
    d. How to execute Java programs?
    e. Java programming phases
    f. How to run java programs?
    g. Install Java Development Kit-JDK
    h. Install and set up Java IDE-eclipse
    i. Using eclipse to for Java programming
    j. The first Java program

2. Java Fundamentals
    a. Comments
    b. Variables
    c. Basic data types
    d. Using strings
    e. Basic operators
    f. Data type conversions
    g. Basic debugging
    h. Formatted print statement

3. Java API and Some Standard Classes
    a. A basic understanding of Java objects and classes
    b. Define a Java class
    c. Create class instances
    d. Static and instance method calls
    e. Java library and Java packages
    f. Explore Java API
    g. Use the Math class
    h. Generate random numbers
    i. Using the Scanner class to take console input

4. Making Decisions
    a. Relational operators
    b. Logical operators
    c. Compare strings
    d. The if-else statement
    e. The if--else if--else statement

f. Nested if-else statements
g. The switch control

5. While Loop
    a. Introduce the while loop
    b. The count-controlled while loop
    c. The condition-controlled while loop
    d. The result-controlled while loop
    e. The do-while loop

6. The for Loop
    a. The for loop structure
    b. How it is executed
    c. Nested for loops
    d. The for-each loop

7. Array (1D and 2D)
    a. The purpose of array
    b. Constructing an array
    c. Array elements and index
    d. Initializing an array
    e. Traverse an array

8. Methods
    a. What is a method
    b. Why to use methods
    c. Defining a method
    d. Parameters and arguments
    e. Return type vs void
    f. Invoking a method

9. Classes and Objects
    a. Understand Objects and Classes in Java
    b. Define a Class
    c. Use Constructors
    d. Create Class Instances
    e. Encapsulation and Abstraction
    f. Use Mutators and Accessors
    g. Static variables and static methods
    h. Variable Scope and Keyword “this”
    i. Pass by Value

j. Pass by Reference
k. The Null Object
l. Aggregation: has- a relationship

10. Inheritance
    a. Understand inheritance
    b. Superclass and subclass
    c. Inheritance hierarchy
    d. Is-a relationship
    e. Define a subclass
    f. Subclass' constructors
    g. Use "super" keyword
    h. Access inherited fields
    i. Method overriding
    j. The object class
    k. Abstract class and abstract methods

11. Polymorphism and Interfaces
    a. Method overloading
    b. Overloading vs. overriding
    c. Polymorphism
    d. Static vs. dynamic binding
    e. The concept of interface
    f. Declare an interface
    g. Implement an interface
    h. Benefit of interface
    i. Interface vs. abstract class
    j. Interface in Java API

12. Exceptions and Error Handling
    a. What are Java exceptions
    b. What do you do with exceptions?
    c. The try-catch-finally blocks
    d. Throws clause
    e. Exception handling rules
    f. Java run time errors
    g. Debugging a program

13. File Operations
    a. What is I/O?
    b. File basics

c. File directory and path
d. File text I/O
e. The file object
f. Read from a file
g. Write to a file
h. File exception handling
i. Append text to a file
j. Token based vs. line based processing
k. Java I/O classes

14. ArrayList, Wrapper Classes, and Comparable Interface
    a. ArrayList vs. regular array
    b. Creating an ArrayList
    c. Adding objects to ArrayList
    d. ArrayList operations
    e. Generic class type
    f. Wrapper classes
    g. Object class
    h. toString() method
    i. The Comparable interface
    j. The String Class

15. Graphic User Interface I
    a. Using message dialog
    b. Using input dialog
    c. Using frame
    d. LayoutManager
    e. FlowLayout
    f. GridLayout
    g. Using labels
    h. Using text field
    i. Using button
    j. Using textbox

16. Graphic User Interface II
    a. Event handler
    b. Event listener
    c. BorderLayout
    d. Using Panel
    e. UI Layers and composite layouts
    f. Using CheckBox

g. Design User Interactive Programs with UI

17. Recursion and Algorithms
    a. What is algorithm?
    b. Recursion vs. looping
    c. The base case
    d. The factorial recursive method
    e. The Fibonacci recursive method
    f. The Find-Highest Algorithm

18. Sorting Algorithms and Binary Search
    a. Background on sorting data
    b. Insertion sort
    c. Selection sort
    d. Merge sort
    e. Binary search