Eunduring Understanding, Learning Objective, Lesson Plan Map

Legend

Enduring Understanding

Learning Objective

Lesson Plan

Mapping

CON-1 The way variables and operators are sequenced and combined in an expression determines the computed result.

CON-1.A Evaluate arithmetic expressions in a program code.

2.01 Basic Data Concepts

2.02 Declaring & Assigning Variables

CON-1.B Evaluate what is stored in a variable as a result of an expression with an assignment statement.

2.03 String Concatenation & Increment Decrement Operators

CON-1.C Evaluate arithmetic expressions that use casting.

2.04 Mixing Types & Casting

CON-1.D Evaluate expressions that use the Math class methods.

3.02 Limitations of Parameters & Multiple Parameters

3.05 Using Objects & String Processing

CON-1.E Evaluate Boolean expressions that use relational operators in program code.

3.09 Relational Operators & if/else

3.10 Nested if/else Statements

CON-1.F Evaluate compound Boolean expressions in program code.

3.15 Fencepost & Sentinel Loops

CON-2 Programmers incorporate iteration and selection into code as a way of providing instructions for the computer to process each of the many possible input values.

CON-2.C Represent iterative processes using a while loop.

3.12 Cumulative Algorithms

3.13 while Loops

CON-2.D For algorithms in the context of a particular specification that does not require the use of traversals

3.14 Random Numbers

CON-2.E Represent iterative processes using a for loop

3.14 Random Numbers

CON-2.K Apply sequential/linear search algorithms to search for specific information in array or ArrayList objects.

7.01 Searching Algorithms

CON-2.L Apply selection sort and insertion sort algorithms to sort the elements of array or ArrayList objects.

7.02 Sorting Algorithms

CON-2.O Determine the result of executing recursive methods.

8.01 Thinking Recursively

8.02 Writing Recursive Solutions

8.03 Mechanics of Recursion

CON-2.P Apply recursive search algorithms to information in String, 1D array, or ArrayList objects.

8.04 MergeSort

MOD-1 Some objects or concepts are so frequently represented that programmers can draw upon existing code that has already been tested, enabling them to write solutions more quickly and with a greater degree of confidence.

MOD-1.A Call System class methods to generate output to the console.

1.03 String & Console Output

MOD-1.B Explain the relationship between a class and an object.

5.01 Object Oriented Programming

MOD-1.C Identify, using its signature, the correct constructor being called

3.01 Parameters

5.03 Object Initialization: Constructors

MOD-1.E Call non-static void methods without parameters.

5.02 Object State & Behavior

5.04 Encapsulation

MOD-2 Programmers use code to represent a physical object or nonphysical concept, real or imagined, by defining a class based on the attributes and/or behaviors of the object or concept.

MOD-2.A Designate access and visibility constraints to classes, data, constructors, and methods.

5.03 Object Initialization: Constructors

5.04 Encapsulation

5.05 Finding & Fixing Errors

MOD-2.C Describe the functionality and use of program code through comments.

1.04 Common Errors & Comments

MOD-2.D Define behaviors of an object through non-void methods without parameters written in a class.

5.01 Object Oriented Programming

6.02 Overriding Methods & Accessing Inherited Code

MOD-2.G Define behaviors of a class through static methods.

1.05 Static Methods & Method Calls

1.06 Static Methods & Method Calls

5.02 Object State & Behavior

MOD-3 When multiple classes contain common attributes and behaviors, programmers create a new class containing the shared attributes and behaviors forming a hierarchy. Modifications made at the highest level of the hierarchy apply to the subclasses

MOD-3.B Create an inheritance relationship from a subclass to the superclass.

6.03 Interacting with the Object Superclass

6.01 Inheritance Basics

MOD-3.C Define reference variables of a superclass to be assigned to an object of a subclass in the same hierarchy.

6.05 Has-a Relationships

MOD-3.D Call methods in an inheritance relationship.

6.04 Polymorphism

VAR-1 To find specific solutions to generalizable problems, programmers include variables in their code so that the same algorithm runs using different input values.

VAR-1.E For String class: a. Create String objects. b. Call String methods.

3.03 Return Values

VAR-1.G Explain where variables can be used in the program code.

1.02 Algorithms & Computational Thinking

VAR-2 To manage large amounts of data or complex relationships in data, programmers write code that groups the data together into a single data structure without creating individual variables for each value.

VAR-2.A Represent collections of related primitive or object reference data using two-dimensional (2D) array objects.

4.01 Array Basics

VAR-2.C Traverse the elements in a 1D array object using an enhanced for loop.

4.02 For-Each Loop & Arrays Class

VAR-2.D Represent collections of related object reference data using ArrayList objects.

4.07 ArrayList

VAR-2.F Traverse the elements in an ArrayList object using an enhanced for loop.

4.06 Nested Loop Algorithms & Rectangular Arrays