AP Computer Science A
Language Features and Other Testable Topics

Tested on the AP CS A Exam	Learning Objective(s)	Not tested on the AP CS A Exam, but potentially relevant/useful
Comments		
`/* */`, `//`, and `/** */`	MOD-2.C.1 thru 2.C.2	Javadoc tool
Primitive Types		
`int`	VAR-1.C.1	`char`, `byte`, `short`, `long`, `float`
`Integer.MIN_VALUE/MAX_VALUE`	CON-1.C.5, VAR-1.F.2	
Overflow	CON-1-C.6	
`double`	VAR-1.C.1	
`boolean` (values: `true`, `false`)	VAR-1.C.1, VAR-1.C.3, CON-1.E.3, CON-1.F.3	
Operators		
Arithmetic: `+`, `-`, `*`, `/`, `%`	CON-1.A.2 thru 1.A.7	
Increment/Decrement: `++`, `--`	CON-1.B.5	
Assignment: `=`, `+=`, `-=`, `*=`, `/=`, `%=`	CON-1.B.1, CON-1.B.4	
String concatenation: `+`, `+=`	VAR-1.E.3 thru 1.E.4	`StringBuilder`
Numeric casts: `(int)`, `(double)`	CON-1.C.1 thru 1.C.3	`(char)`, `(float)`
Relational: `==`, `!=`, `<`, `<=`, `>`, `>=`	CON-1.E.1 thru 1.E.3	
Logical: `!`, `&&`, `||`	CON-1.F.1 thru 1.F.2	`&`, `|`, `^`
		Shift: `<<`, `>>`, `>>>`
Short circuited evaluation	CON-1.F.3	Bitwise: `~`, `&`, `|`, `^`
DeMorgan's Laws	CON-1.G.1	Conditional: `?:`
Escape Sequences		
`\"`, `\\`, `\n` inside strings	VAR-1.E.5	`\'`, `\t`, `\unnnn`
Input/Output		
`System.out.print`, `System.out.println`	MOD-1.A.1 thru 1.A.2	`Scanner`, `System.in`, `System.out`, `System.err`, Stream input/output, GUI input/output
		Parsing input: `Integer.parseInt`, `Double.parseDouble`
		Formatting output: `System.out.printf`, `String.format`
Control Statements		
`if`	CON-2.A.2 thru 2.A.3	
`if-else`	CON-2.A.4	
`if-else-if`	CON-2.A.5	`switch`
Nested `if`	CON-2.B.1	
`while`	CON-2.C.1 thru 2.C.5, CON-2.E.4 thru 2.E.5	`break`, `continue`, `do-while`
`for`, enhanced `for` (`for`-each)	CON-2.C.1 thru 2.C.5, CON-2.E.1 thru 2.E.5, VAR-2.C.1 thru 2.C.4, VAR-2.G.3	
Nested iteration	CON-2.G.1 thru 2.G.2, VAR-2.G.1 thru 2.G.3	
`return`	MOD-1.E.4, CON-2.C.5, MOD-2.D.2 thru 2.D.5	
Variables		
Parameter variables	MOD-1.C.1 thru 1.C.2	final parameter variables
Local variables	VAR-1.G.1 thru 1.G.3	final local variables
Instance variables	MOD-2.A.4, MOD-3.A.3, VAR-1.G.2	final instance variables
visibility (`private`)	MOD-2.H.1 thru 2.H.3	
Static (class) variables		
visibility (`public`, `private`)		
`final`	VAR-1.C.4	

AP Computer Science A
Language Features and Other Testable Topics

Tested on the AP CS A Exam	Learning Objective(s)	Not tested on the AP CS A Exam, but potentially relevant/useful
Methods		
Visibility (`public`, `private`)	MOD-2.A.1 thru 2.A.2, MOD-2.A.6	visibility (`protected`)
Static vs. non-static	MOD-1.E.5 thru 1.E.6, MOD-2.G.1 thru 2.G.5, MOD-1.H.1	`public static void main(String[]` args), command line arguments
Method signatures	MOD-1.C.1, MOD-1.E.3, MOD-1.F.1 thru 1.F.3	variable number of parameters, final parameters
Overloading, overriding	MOD-1.F.3, MOD-3.B.10	
Formal vs. actual parameters passed using call-by-value	MOD-1.C.2 thru 1.C.3, MOD-1.C.5 thru 1.C.6, MOD-1.F.1 thru 1.F.6, VAR-1.G.3	
Dot (.) operator	MOD-1.E.6, MOD-1.H.1	
Void method	MOD-1.E.7, MOD-2.E.1	
Non-void method	MOD-1.G.1, MOD-2.D.2 thru 2.D.4, MOD-2.F.2	
Precondition	MOD-2.C.3, MOD-2.C.5	
Postcondition	MOD-2.C.4, MOD-2.C.5	
`toString`	MOD-2.D.6 thru 2.D.7	
Constructors		
`new`	MOD-1.D.1 thru 1.D.2, MOD-1.D.4	default initialization of instance variables, initialization blocks
`super()`, `super(`*args*`)`	MOD-3.B.6 thru 3.B.9, MOD-3.B.14 thru 3.B.15	`this(`*args*`)`
Visibility (`public`)	MOD-2.A.5	visibility (`private`, `protected`)
Initialize state of the object	MOD-2.B.1 thru 2.B.3	
No-argument	MOD-2.B.5	
Overloading	MOD-1.C.4	
Classes		
Visibility (`public`)	MOD-2.A.3	visibility (`private`, `protected`)
Accessor methods	MOD-3.A.4, MOD-2.D.1	final class,
Modifier (mutator) methods	MOD-2.E.2	nested classes,
Design/create/modify class	MOD-1.B.2, MOD-3.A.2, VAR-1.G.4	inner classes, enumerations
Encapsulation	MOD-3.A.1 thru 3.A.4	
Inheritance		
`extends`	MOD-3.B.4	
`super.`*method* (*args*)	MOD-3.B.14 thru 3.B.15	
Understand inheritance hierarchies, (subclass/superclass relationship)	MOD-3.B.1 thru 3.B.5	
Design/create/modify subclasses	MOD-3.B.11 thru 3.B.15	
Declare/create objects of subclasses	MOD-3.C.1 thru 3.C.4, MOD-3.D.2 thru 3.D.3	
Polymorphism	MOD-3.C.1 thru 3.C.4	
Object Comparison		
object identity (==, !=) vs. object equality (`equals`)	CON-1.H.2 thru 1.H.4	
implementation of `equals`	CON-1.H.4, MOD-3.E.4	
`String compareTo`	VAR-1.E.12	`Comparable` interface
Miscellaneous OOP		
"is-a" and "has-a" relationships	MOD-2.B.1, MOD-3.B.3, MOD-3.C.1 thru 3.C.4	`instanceof`, (*ClassName*) cast
`null`	VAR-1.D.1 thru 1.D.2, MOD-1.E.8	
`this`	VAR-1.H.1, VAR-1.H.2	`this.`*var*, `this.`*method* (*args*)

AP Computer Science A
Language Features and Other Testable Topics

Tested on the AP CS A Exam	Learning Objective(s)	Not tested on the AP CS A Exam, but potentially relevant/useful
Packages		
`import` *packageName.className*	VAR-2.D.6	`import` *packageName* .*, static import, package packageName, class path
Standard Java Library		
`Object`	MOD-3.D.1, MOD-3.E.1 thru 3.E.4	clone
`Integer`, `Double`	VAR-1.F.1 thru 1.F.3	
`String`	VAR-1.E.1 thru 1.E.4, VAR-1.E.9 thru 1.E.13, VAR-1.A.1, CON-2.F.1	
`Math`	CON-1.D.1 thru 1.D.4	
`ArrayList<E>`	VAR-2.D.1 thru 2.D.7, VAR-2.E.1 thru 2.E.4	`Collection<E>`, `Arrays`, `Collections`
Autoboxing Unboxing	VAR-1.F.4 thru 1.F.5 VAR-1.F.6 thru 1.F.7	
Arrays		
1-dimensional arrays Capacity: *arr*`.length` Default values: `0`, `0.0`, `false`, `null`	VAR-2.A.1 thru 2.A.4, VAR-2.A.6 thru 2.A.7, VAR-2.B.1 thru 2.B.3, VAR-2.C.2 thru 2.C.3	`new` *type* `[] {…}`
2-dimensional rectangular arrays	VAR-2.F.1 thru 2.F.4, VAR-2.G.1 thru 2.G.3	ragged arrays (non-rectangular), arrays with 3 or more dimensions
Initializer list: `{ … }`	VAR-2.A.5, VAR-2.F.3	
Row-major and column-major order traversals of 2D array elements	VAR-2.F.5	
Exceptions		

`ArithmeticException`
`NullPointerException`
`IndexOutOfBoundsException`
`ArrayIndexOutOfBoundsException`
`ConcurrentModificationException`
	CON-1.A.8
	MOD-1.E.8
	VAR-1.E.10, VAR-2.E.3
	VAR-2.A.7, VAR-2.B.3
	VAR-2.E.4
		`try/catch/finally`
		`throw`, `throws`
		`assert`

Common Algorithms		
Identify if integer is(not) divisible by another; Identify individual digits in an integer	CON-2.D.1	
Determine min/max value; Compute sum/average/mode	CON-2.D.2	
Array (1D, 2D) traversal (minimum/maximum value, sum, average, mode, element found, access consecutive pairs, determine if duplicates, count elements meeting criteria, shift or rotate elements left/right, reverse order of elements)	CON-2.I.1 thru 2.I.2, CON-2.J.1 thru 2.J.2, CON-2.N.1 thru 2.N.2	
`ArrayList` traversal (insert elements, delete elements, and all listed for arrays above)	CON-2.J.1 thru 2.J.2, CON-2.I.1 thru 2.I.2	
Sequential/linear search	CON-2.K.1 thru 2.K.2, CON-2.N.1	
Binary search	CON-2.P.1 thru 2.P.4	
Selection sort, Insertion sort	CON-2.L.1	
Merge sort	CON-2.Q.1	
Informal run-time comparisons	CON-2.M.1	
Recursion		
Base case, Recursive call	CON-2.O.1 thru 2.O.6	writing recursive methods