SUGGESTED SKILL

5.A

Describe the behavior of a given segment of program code.

AVAILABLE RESOURCE

- Runestone Academy: AP CSA—Java Review: 2.2—What is a Class and an Object?

TOPIC 2.1
Objects: Instances of Classes

Required Course Content

ENDURING UNDERSTANDING

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.

LEARNING OBJECTIVE

MOD-1.B

Explain the relationship between a class and an object.

ESSENTIAL KNOWLEDGE

MOD-1.B.1

An object is a specific instance of a class with defined attributes.

MOD-1.B.2

A class is the formal implementation, or blueprint, of the attributes and behaviors of an object.

TOPIC 2.2
Creating and Storing Objects (Instantiation)

SUGGESTED SKILLS

1.C
Determine code that would be used to interact with completed program code.

3.A
Write program code to create objects of a class and call methods.

Required Course Content

ENDURING UNDERSTANDING

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.

LEARNING OBJECTIVE

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

ESSENTIAL KNOWLEDGE

MOD-1.C.1
A signature consists of the constructor name and the parameter list.

MOD-1.C.2
The parameter list, in the header of a constructor, lists the types of the values that are passed and their variable names. These are often referred to as formal parameters.

MOD-1.C.3
A parameter is a value that is passed into a constructor. These are often referred to as actual parameters.

MOD-1.C.4
Constructors are said to be overloaded when there are multiple constructors with the same name but a different signature.

MOD-1.C.5
The actual parameters passed to a constructor must be compatible with the types identified in the formal parameter list.

MOD-1.C.6
Parameters are passed using call by value. Call by value initializes the formal parameters with copies of the actual parameters.

continued on next page

Using Objects

LEARNING OBJECTIVE

MOD-1.D

For creating objects:

a. Create objects by calling constructors without parameters.

b. Create objects by calling constructors with parameters.

ESSENTIAL KNOWLEDGE

MOD-1.D.1

Every object is created using the keyword `new` followed by a call to one of the class's constructors.

MOD-1.D.2

A class contains constructors that are invoked to create objects. They have the same name as the class.

MOD-1.D.3

Existing classes and class libraries can be utilized as appropriate to create objects.

MOD-1.D.4

Parameters allow values to be passed to the constructor to establish the initial state of the object.

ENDURING UNDERSTANDING

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.

LEARNING OBJECTIVE

VAR-1.D

Define variables of the correct types to represent reference data.

ESSENTIAL KNOWLEDGE

VAR-1.D.1

The keyword `null` is a special value used to indicate that a reference is not associated with any object.

VAR-1.D.2

The memory associated with a variable of a reference type holds an object reference value or, if there is no object, `null`. This value is the memory address of the referenced object.

TOPIC 2.3
Calling a Void Method

Required Course Content

ENDURING UNDERSTANDING

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.

LEARNING OBJECTIVE

MOD-1.E

Call non-static void methods without parameters.

ESSENTIAL KNOWLEDGE

MOD-1.E.1

An object's behavior refers to what the object can do (or what can be done to it) and is defined by methods.

MOD-1.E.2

Procedural abstraction allows a programmer to use a method by knowing what the method does even if they do not know how the method was written.

MOD-1.E.3

A method signature for a method without parameters consists of the method name and an empty parameter list.

MOD-1.E.4

A method or constructor call interrupts the sequential execution of statements, causing the program to first execute the statements in the method or constructor before continuing. Once the last statement in the method or constructor has executed or a return statement is executed, flow of control is returned to the point immediately following where the method or constructor was called.

continued on next page

SUGGESTED SKILLS

1.C

Determine code that would be used to interact with completed program code.

3.A

Write program code to create objects of a class and call methods.

AVAILABLE RESOURCE

- Classroom Resources > GridWorld Case Study: Part I

Using Objects

LEARNING OBJECTIVE

MOD-1.E

Call non-static void methods without parameters.

ESSENTIAL KNOWLEDGE

MOD-1.E.5

Non-static methods are called through objects of the class.

MOD-1.E.6

The dot operator is used along with the object name to call non-static methods.

MOD-1.E.7

Void methods do not have return values and are therefore not called as part of an expression.

MOD-1.E.8

Using a `null` reference to call a method or access an instance variable causes a `NullPointerException` to be thrown.

TOPIC 2.4
Calling a Void Method with Parameters

Required Course Content

ENDURING UNDERSTANDING

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.

LEARNING OBJECTIVE

MOD-1.F

Call non-static void methods with parameters.

ESSENTIAL KNOWLEDGE

MOD-1.F.1

A method signature for a method with parameters consists of the method name and the ordered list of parameter types.

MOD-1.F.2

Values provided in the parameter list need to correspond to the order and type in the method signature.

MOD-1.F.3

Methods are said to be overloaded when there are multiple methods with the same name but a different signature.

SUGGESTED SKILLS

2.C

Determine the result or output based on the statement execution order in a code segment containing method calls.

3.A

Write program code to create objects of a class and call methods.

AVAILABLE RESOURCE

- Practice-It!: BJP4 Chapter 3: Parameters and Objects—Self-Check 3.2–3.9

SUGGESTED SKILLS

1.C
Determine code that would be used to interact with completed program code.

3.A
Write program code to create objects of a class and call methods.

AVAILABLE RESOURCES

- The Exam > **2018 AP Computer Science A Exam Free-Response Question #1 (Frog Simulation)**

TOPIC 2.5
Calling a Non-void Method

Required Course Content

ENDURING UNDERSTANDING

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.

LEARNING OBJECTIVE

MOD-1.G

Call non-static non-void methods with or without parameters.

ESSENTIAL KNOWLEDGE

MOD-1.G.1

Non-void methods return a value that is the same type as the return type in the signature. To use the return value when calling a non-void method, it must be stored in a variable or used as part of an expression.

TOPIC 2.6
`String` Objects: Concatenation, Literals, and More

Required Course Content

ENDURING UNDERSTANDING

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.

LEARNING OBJECTIVE

VAR-1.E

For `String` class:

a. Create `String` objects.

b. Call `String` methods.

ESSENTIAL KNOWLEDGE

VAR-1.E.1

`String` objects can be created by using string literals or by calling the `String` class constructor.

VAR-1.E.2

`String` objects are immutable, meaning that `String` methods do not change the `String` object.

VAR-1.E.3

`String` objects can be concatenated using the + or += operator, resulting in a new `String` object.

VAR-1.E.4

Primitive values can be concatenated with a `String` object. This causes implicit conversion of the values to `String` objects.

VAR-1.E.5

Escape sequences start with a `\` and have a special meaning in Java. Escape sequences used in this course include `\"`, `\\`, and `\n`.

SUGGESTED SKILL

2.A

Apply the meaning of specific operators.

AVAILABLE RESOURCES

- Runestone Academy: AP CSA—Java Review: 4—Strings
- Practice-It!: BJP4 Chapter 3: Parameters and Objects—Self-Check 3.18

SUGGESTED SKILLS

2.C
Determine the result or output based on the statement execution order in a code segment containing method calls.

3.A
Write program code to create objects of a class and call methods.

AVAILABLE RESOURCES

- Java Quick Reference (see Appendix)
- **Runestone Academy: AP CSA—Java Review: 4.3—String Methods on the Exam**
- **CodingBat Java: String-1**
- **Practice-It!: BJP4 Chapter 3: Parameters and Objects—Self-Check 3.19 and 3.20**

TOPIC 2.7
`String` Methods

Required Course Content

ENDURING UNDERSTANDING

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.

LEARNING OBJECTIVE

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

ESSENTIAL KNOWLEDGE

VAR-1.E.6
Application program interfaces (APIs) and libraries simplify complex programming tasks.

VAR-1.E.7
Documentation for APIs and libraries are essential to understanding the attributes and behaviors of an object of a class.

VAR-1.E.8
Classes in the APIs and libraries are grouped into packages.

VAR-1.E.9
The `String` class is part of the `java.lang` package. Classes in the `java.lang` package are available by default.

VAR-1.E.10
A `String` object has index values from 0 to `length()-1`. Attempting to access indices outside this range will result in an `StringIndexOutOfBoundsException`.

VAR-1.E.11
A `String` object can be concatenated with an object reference, which implicitly calls the referenced object's `toString` method.

continued on next page

LEARNING OBJECTIVE

VAR-1.E

For `String` class:

a. Create `String` objects.

b. Call `String` methods.

ESSENTIAL KNOWLEDGE

VAR-1.E.12

The following `String` methods and constructors—including what they do and when they are used—are part of the Java Quick Reference:

- `String(String str)` — Constructs a new `String` object that represents the same sequence of characters as `str`
- `int length()` — Returns the number of characters in a `String` object
- `String substring(int from, int to)` — Returns the substring beginning at index `from` and ending at index `to - 1`
- `String substring(int from)` — Returns `substring(from, length())`
- `int indexOf(String str)` — Returns the index of the first occurrence of `str`; returns –1 if not found
- `boolean equals(String other)` — Returns `true` if `this` is equal to `other`; returns `false` otherwise
- `int compareTo(String other)` — Returns a value < 0 if `this` is less than `other`; returns zero if `this` is equal to `other`; returns a value > 0 if `this` is greater than `other`

VAR-1.E.13

A string identical to the single element substring at position `index` can be created by calling `substring(index, index + 1)`.

SUGGESTED SKILL

2.C

Determine the result or output based on the statement execution order in a code segment containing method calls.

AVAILABLE RESOURCE

- Java Quick Reference (see Appendix)

TOPIC 2.8
Wrapper Classes: `Integer` and `Double`

Required Course Content

ENDURING UNDERSTANDING

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.

LEARNING OBJECTIVE

VAR-1.F

For wrapper classes:

a. Create `Integer` objects.

b. Call `Integer` methods.

c. Create `Double` objects.

d. Call `Double` methods.

ESSENTIAL KNOWLEDGE

VAR-1.F.1

The `Integer` class and `Double` class are part of the `java.lang` package.

VAR-1.F.2

The following `Integer` methods and constructors—including what they do and when they are used—are part of the Java Quick Reference:

- `Integer(int value)` — Constructs a new `Integer` object that represents the specified `int` value
- `Integer.MIN_VALUE` — The minimum value represented by an `int` or `Integer`
- `Integer.MAX_VALUE` — The maximum value represented by an `int` or `Integer`
- `int intValue()` — Returns the value of this `Integer` as an `int`

continued on next page

LEARNING OBJECTIVE

VAR-1.E

For wrapper classes:

a. Create `Integer` objects.

b. Call `Integer` methods.

c. Create `Double` objects.

d. Call `Double` methods.

ESSENTIAL KNOWLEDGE

VAR-1.F.3

The following `Double` methods and constructors—including what they do and when they are used—are part of the Java Quick Reference:

- `Double(double value)` — Constructs a new `Double` object that represents the specified `double` value
- `double doubleValue()` — Returns the value of this `Double` as a `double`

VAR-1.F.4

Autoboxing is the automatic conversion that the Java compiler makes between primitive types and their corresponding object wrapper classes. This includes converting an `int` to an `Integer` and a `double` to a `Double`.

VAR-1.F.5

The Java compiler applies autoboxing when a primitive value is:

- Passed as a parameter to a method that expects an object of the corresponding wrapper class.
- Assigned to a variable of the corresponding wrapper class.

VAR-1.F.6

Unboxing is the automatic conversion that the Java compiler makes from the wrapper class to the primitive type. This includes converting an `Integer` to an `int` and a `Double` to a `double`.

VAR-1.F.7

The Java compiler applies unboxing when a wrapper class object is:

- Passed as a parameter to a method that expects a value of the corresponding primitive type.
- Assigned to a variable of the corresponding primitive type.

SUGGESTED SKILLS

1.B
Determine code that would be used to complete code segments.

3.A
Write program code to create objects of a class and call methods.

AVAILABLE RESOURCES

- Java Quick Reference (see Appendix)
- **Practice-It!: BJP4 Chapter 3: Parameters and Objects—Exercises 3.7 and 3.8**

TOPIC 2.9
Using the `Math` Class

Required Course Content

ENDURING UNDERSTANDING

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.

LEARNING OBJECTIVE

MOD-1.H
Call static methods.

ESSENTIAL KNOWLEDGE

MOD-1.H.1
Static methods are called using the dot operator along with the class name unless they are defined in the enclosing class.

ENDURING UNDERSTANDING

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

LEARNING OBJECTIVE

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

ESSENTIAL KNOWLEDGE

CON-1.D.1
The `Math` class is part of the `java.lang` package.

CON-1.D.2
The `Math` class contains only static methods.

continued on next page

LEARNING OBJECTIVE

CON-1.D

Evaluate expressions that use the `Math` class methods.

ESSENTIAL KNOWLEDGE

CON-1.D.3

The following static `Math` methods—including what they do and when they are used—are part of the Java Quick Reference:

- `int abs(int x)` — Returns the absolute value of an `int` value
- `double abs(double x)` — Returns the absolute value of a `double` value
- `double pow(double base, double exponent)` — Returns the value of the first parameter raised to the power of the second parameter
- `double sqrt(double x)` — Returns the positive square root of a `double` value
- `double random()` — Returns a double value greater than or equal to 0.0 and less than 1.0

CON-1.D.4

The values returned from `Math.random` can be manipulated to produce a random `int` or `double` in a defined range.