socrative

Answer Key
February 5, 2021

OCA Practice Questions Ch. 2 ('Copy')

1. Which of the following declarations does not compile?

✓ **A** double num1, int num2 =0;

B int num1, num2;

C int num1, num2 =0;

D int num1 = 0, num2 =0;

i Option A does not compile because Java does not allow declaring different types as part of the same declaration. The other three options show various legal combinations of combining multiple variables in the same declarations with optional default values.

2.
```
public static void main(String... args) {
   String chair, table = "metal";
   chair = chair + table;
   System.out.println(chair);
}
```

What is the output of the following?

A metal

B metalmetal

C nullmetal

✓ **D** The code does not compile.

i The table variable is initialized to "metal". However, chair is not initialized. In Java, initialization is per variable and not for all the variables in a single declaration. Therefore, the second line tries to reference an uninitialized local variable and does not compile, which makes Option D correct.

3. Which is correct about an instance variable of type String?

A It defaults to an empty string.

✓ **B** It defaults to null.

C It does not have a default value.

D It will not compile without initializing on the declaration line.

i Instance variables have a default value based on the type. For any non-primitive, including String, that type is a reference to null. Therefore Option B is correct. If the variable was a local variable, Option C would be correct.

4. Which of the following is not a valid variable name?

A _blue

✓ B 2blue

C blue$

D Blue

i An identifier name must begin with a letter, $, or _. Numbers are only permitted for subsequent characters. Therefore, Option B is not a valid variable name.

5. Which of these class names best follows standard Java naming conventions?

A fooBar

✓ B FooBar

C FOO_BAR

D F_o_o_B_a_r

i In Java, class names begin with an uppercase letter by convention. Then they use lowercase with the exception of new words. Option B follows this convention and is correct. Option A follows the convention for variable names. Option C follows the convention for constants. Option D doesn't follow any Java conventions.

6.
```
public String convert(int value) {
   return value.toString();
}
public String convert(Integer value) {
   return value.toString();
}
public String convert(Object value) {
   return value.toString();
}
```

How many of the following methods compile?

A None

B One

✓ C Two

D Three

i Objects have instance methods while primitives do not. Since int is a primitive, you cannot call instance methods on it. Integer and String are both objects and have instance methods. Therefore, Option C is correct.

7. Which of the following does not compile?

A int num = 999;

B int num = 9_9_9;

✓ C int num = _9_99;

D None of the above; they all compile.

i Underscores are allowed between any two digits in a numeric literal. Underscores are not allowed at the beginning or end of the literal, making Option C the correct answer.

8. Which of the following is a wrapper class?

A int

B Int

✓ C Integer

D Object

i Option A is incorrect because int is a primitive. Option B is incorrect because it is not the name of a class in Java. While Option D is a class in Java, it is not a wrapper class because it does not map to a primitive. Therefore, Option C is correct.

9.

```
public class Values {
   integer a = Integer.valueOf("1");
   public static void main(String[] nums) {
      integer a = Integer.valueOf("2");
      integer b = Integer.valueOf("3");
      System.out.println(a + b);
   }
}
```

What is the result of running this code?

A 4

B 5

✓ C The code does not compile.

D The code compiles but throws an exception at runtime.

i There is no class named integer. There is a primitive int and a class Integer. Therefore, the code does not compile, and Option C is correct. If the type was changed to Integer, Option B would be correct.

10. Which best describes what the new keyword does?

A Creates a copy of an existing object and treats it as a new one

B Creates a new primitive

✓ C Instantiates a new object

D Switches an object reference to a new one

i The new keyword is used to call the constructor for a class and instantiate an instance of the class. A

primitive cannot be created using the new keyword. Dealing with references happens after the object created by new is returned.

11. Which is the first line to trigger a compiler error?

```
double d1 = 5f;    // p1
double d2 = 5.0;   // p2
float f1 = 5f;     // p3
float f2 = 5.0;    // p4
```

- A p1
- B p2
- C P3
- ✓ D P4

i Java uses the suffix f to indicate a number is a float. Java automatically widens a type, allowing a float to be assigned to either a float or a double. This makes both lines p1 and p3 compile. Line p2 does compile without a suffix. Line p4 does not compile without a suffix and therefore is the answer.

12. Which of the following lists of primitive types are presented in order from smallest to largest data type?

- ✓ A byte, char, float, double
- B byte, char, double, float
- C char, byte, float, double
- D char, double, float, bigint

i A byte is smaller than a char, making Option C incorrect. bigint is not a primitive, making Option D incorrect. A double uses twice as much memory as a float variable, therefore Option A is correct.

13. Which of the following is not a valid order for elements in a class?

- A Constructor, instance variables, method names
- B Instance variables, constructor, method names
- C Method names, instance variables, constructor
- ✓ D None of the above: all orders are valid.

i The instance variables, constructor, and method names can appear in any order within a class declaration.

14.

```
String title = "Weather";                    // line x1
int hot, double cold;                        // line x2
System.out.println(hot + " " + title);       // line x3
```

Which of the following lines contains a compiler error?

A x1

✓ B x2

C x3

D None of the above

i Java does not allow multiple Java data types to be declared in the same declaration, making Option B the correct answer. If double was removed, both hot and cold would be the same type. Then the compiler error would be on x3 because of a reference to an uninitialized variable.

15.

```
1:  public class Bowling {
2:     { System.out.println(); }
3:     public Bowling () {
4:        System.out.println();
5:     }
6:     static { System.out.println(); }
7:     { System.out.println(); }
8:  }
```

How many instance initializers are in this code?

A None

B One

✓ C Two

D Three

i Lines 2 and 7 illustrate instance initializers. Line 6 is a static initializer. Lines 3–5 are a constructor.

16.

```
public static void main(String[] args) {
    _______ defaultValue;
    System.out.println(defaultValue);
}
```

Of the types double, int, and short, how many could fill in the blank to have this code output 0?

✓ A None

B One

C Two

D Three

i Since defaultValue is a local variable, it is not automatically initialized. That means the code will not compile with any type. Therefore, Option A is correct. If this was an instance variable, Option C would be correct as int and short would be initialized to 0 while double would be initialized to 0.0.

17. What is true of the finalize() method?

✓ **A** It may be called zero or one times.

B It may be called zero or more times.

C It will be called exactly once.

D It may be called one or more times.

i The finalize() method may not be called, such as if your program crashes. However, it is guaranteed to be called no more than once.

18. Which of the following is not a wrapper class?

A Double

B Integer

C Long

✓ **D** String

i String is a class, but it is not a wrapper class. In order to be a wrapper class, the class must have a one-to-one mapping with a primitive.

19. Suppose you have the following code. Which of the images best represents the state of the references right before the end of the main method, assuming garbage collection hasn't run?

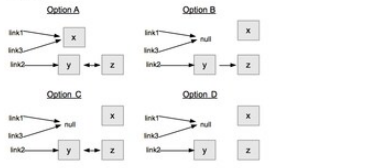
A Option A

B Option B

✓ **C** Option C

D Option D

```
1:  public class Link {
2:     private String name;
3:     private Link next;
4:     public Link(String name, Link next) {
5:       this.name = name;
6:       this.next = next;
7:     }
8:     public void setNext(Link next) {
9:       this.next = next;
10:    }
11:    public Link getNext() {
12:       return next;
13:    }
14:    public static void main(String... args) {
15:       Link link1 = new Link("x", null);
16:       Link link2 = new Link("y", link1);
17:       Link link3 = new Link("z", link2);
18:       link2.setNext(link3);
19:       link3.setNext(link2);
20:       link1 = null;
21:       link3 = null;
22:    }
23: }
```



i Lines 15–17 create the three objects. Lines 18–19 change the references so link2 and link3 point to each other. The lines 20–21 wipe out two of the original references. This means the object with name as x is inaccessible.

20.

```
_____ pi = 3.14;
```

Which type can fill in the blank?

- A byte
- B float
- ✓ C double
- D short

i Options A and D are incorrect because byte and short do not store values with decimal points. Option B is tempting. However, 3.14 is automatically a double. It requires casting to float or writing 3.14f in order to be assigned to a float. Therefore, Option C is correct.

21.

```
public static void main(String[] args) {
   int Integer = 0;          // k1
   Integer int = 0;          // k2
   Integer ++;               // k3
   int++;                    // k4
}
```

What is the first line in the following code to not compile?

- A k1
- ✓ B k2
- C k3
- D k4

i Integer is the name of a class in Java. While it is bad practice to use the name of a class as your local variable name, this is legal. Therefore, k1 does compile. It is not legal to use a reserved word as a variable name. All of the primitives including int are reserved words. Therefore, k2 does not compile, and Option B is the answer. Line k4 doesn't compile either, but the question asks about the first line to not compile.

22. Suppose foo is a reference to an instance of a class. Which of the following is not true about foo.bar?

- A bar is an instance variable.
- ✓ B bar is a local variable.
- C It can be used to read from bar.
- D It can be used to write to bar.

i Dot notation is used for both reading and writing instance variables, assuming they are in scope. It cannot be used for referencing local variables, making Option B the correct answer.

23. Which of the following is not a valid class declaration?

- A class building {}
- B class Cost$ {}
- ✓ C class 5MainSt {}
- D class _Outside {}

i Class names follow the same requirements as other identifiers. Underscores and dollar signs are allowed. Numbers are allowed, but not as the first character of an identifier. Therefore, Option C is correct. Note that class names begin with an uppercase letter by convention, but this is not a requirement.

24.

```
__________ d = new _________(1_000_000_.00);
```

Which of the following can fill in the blanks to make this code compile?

- A double, double
- B double, Double
- C Double, double
- ✓ D None of the above

i This question is tricky as it appears to be about primitive vs. wrapper classes. Looking closely, there is an underscore right before the decimal point. This is illegal as the underscore in a numeric literal can only appear between two digits.

25. Which is correct about a local variable of type String?

- A It defaults to an empty string.
- B It defaults to null.
- ✓ C It does not have a default value.
- D It will not compile without initializing on the declaration line.

i Local variables do not have a default initialization value. If they are referenced before being set to a value, the code does not compile. Therefore, Option C is correct. If the variable was an instance variable, Option B would be correct. Option D is tricky. A local variable will compile without an initialization if it isn't referenced anywhere or it is assigned a value before it is referenced.

26.

```
static _______ defaultValue;

   public static void main(String[] args) {
      System.out.println(defaultValue);
   }
```

Of the types double, int, long, and short, how many could fill in the blank to have this code output 0?

A One

B Two

✓ C Three

D Four

i Since defaultValue is an instance variable, it is automatically initialized to the corresponding value for that type. For double, that value is 0.0. By contrast, it is 0 for int, long, and short. Therefore Option C is correct.

27. Which of the following is true about primitives?

A You can call methods on a primitive.

✓ B You can convert a primitive to a wrapper class object simply by assigning it.

C You can convert a wrapper class object to a primitive by calling valueOf().

D You can store a primitive directly into an ArrayList.

i Option B is an example of autoboxing. Java will automatically convert from primitive to wrapper class types and vice versa. Option A is incorrect because you can only call methods on an object. Option C is incorrect because this method is used for converting to a wrapper class from a String. Option D is incorrect because autoboxing will convert the primitive to an object before adding it to the ArrayList.

28.

```
Integer integer = new Integer(4);
System.out.print(integer.byteValue());

System.out.print("-");

int i = new Integer(4);
System.out.print(i.byteValue());
```

What is the output of the following?

A 4-0

B 4-4

✓ C The code does not compile.

D The code compiles but throws an exception at runtime.

i Java does not allow calling a method on a primitive. While autoboxing does allow the assignment of an Integer to an int, it does not allow calling an instance method on a primitive. Therefore, the last line does not compile.

29.

```
public class TennisBall {
   public TennisBall() {
      System.out.println("bounce");
   }
   public static void main(String[] slam) {
       ________________________
   }
}
```

Given the following code, fill in the blank to have the code print bounce.

A TennisBall;

B TennisBall();

C new TennisBall;

✓ D new TennisBall();

i In order to call a constructor, you must use the new keyword. It cannot be called as if it was a normal method. This rules out Options A and B. Further, Option C is incorrect because the parentheses are required.

30. Which of the following correctly assigns animal to both variables?
I. String cat = "animal", dog = "animal";
II. String cat = "animal"; dog = "animal";
III. String cat, dog = "animal";
IV. String cat, String dog = "animal";

✓ A I

B I, II

C I, III

D I, II, III, IV

i Option A (I) correctly assigns the value to both variables. II does not compile as dog does not have a type. Notice the semicolon in that line, which starts a new statement. III compiles but only assigns the value to dog since a declaration only assigns to one variable rather than everything in the declaration. IV does not compile because the type should only be specified once per declaration.

31. Which two primitives have wrapper classes that are not merely the name of the primitive with an uppercase letter?

A byte and char

B byte and int

✓ C char and int

D None of the above

i The wrapper class for int is Integer and the wrapper class for char is Character. All other primitives have the same name. For example, the wrapper class for boolean is Boolean.

32. Which of the following is true about String instance variables?

✓ **A** They can be set to null.

B They can never be set from outside the class they are defined in.

C They can only be set in the constructor.

D They can only be set once per run of the program.

i Assuming the variables are not primitives, they allow a null assignment. The other statements are false.

33. Which statement is true about primitives?

✓ **A** Primitive types begin with a lowercase letter.

B Primitive types can be set to null.

C String is a primitive.

D You can create your own primitive types.

i An example of a primitive type is int. All the primitive types are lowercase, making Option A correct. Unlike object reference variables, primitives cannot reference null. String is not a primitive as evidenced by the uppercase letter in the name and the fact that we can call methods on it. You can create your own classes, but not primitives.

34. How do you force garbage collection to occur at a certain point?

A Call System.forceGc()

B Call System.gc()

C Call System.requireGc()

✓ **D** None of the above

i While you can suggest to the JVM that it might want to run a garbage collection cycle, the JVM is free to ignore your suggestion. Option B is how to make this suggestion. Since garbage collection is not guaranteed to run, Option D is correct.

35.

```
public static void main(String[] fruits) {
   String fruit1 = new String("apple");
   String fruit2 = new String("orange");
   String fruit3 = new String("pear");

   fruit3 = fruit1;
   fruit2 = fruit3;
   fruit1 = fruit2;
}
```

How many of the String objects are eligible for garbage collection right before the end of the main method?

A None

B One

✓ C Two

D Three

i All three references point to the String apple. This makes the other two String objects eligible for garbage collection and Option C correct.

36.

```
__________ d = new ________(1_000_000.00);
```

Which of the following can fill in the blanks to make this code compile?

A double, double

✓ B double, Double

C Double, double

D None of the above

i A constructor can only be called with a class name rather than a primitive, making Options A and C incorrect. The newly constructed Double object can be assigned to either a double or Double thanks to autoboxing. Therefore, Option B is correct.

37.

```
1:  public class InitOrder {
2:     public String first = "instance";
3:     public InitOrder() {
4:        first = "constructor";
5:     }
6:     { first = "block";  }
7:     public void print() {
8:        System.out.println(first);
9:     }
10:    public static void main(String... args) {
11:      new InitOrder().print();
12:    }
13: }
```

What does the following output?

A block

✓ B constructor

C instance

D The code does not compile.

First line 2 runs and sets the variable using the declaration. Then the instance initializer on line 6 runs. Finally, the constructor runs. Since the constructor is the last to run of the three, that is the value that is set when we print the result, so Option B is correct.

38.

```
int i = null;
Integer in = null;
String s = null;
```

How many of the following lines compile?

A None

B One

✓ C Two

D Three

Objects are allowed to have a null reference while primitives cannot. int is a primitive, so assigning null to it does not compile. Integer and String are both objects and can therefore be assigned a null reference. Therefore, Option C is correct.

39. Which pairs of statements can accurately fill in the blanks in this table?

Variable Type	Can be called within the class from what type of method
Instance	Blank 1: _____
Static	Blank 2: _____

A Blank 1: an instance method only, Blank 2: a static method only

B Blank 1: an instance or static method, Blank 2: a static method only

✓ C Blank 1: an instance method only, Blank 2: an instance or static method

D Blank 1: an instance or static method, Blank 2: an instance or static method

An instance variable can only be referenced from instance methods in the class. A static variable can be referenced from any method. Therefore, Option C is correct.

40. Which of the following does not compile?

A double num = 2.718;

✓ B double num = 2._718;

C double num = 2.7_1_8;

D None of the above; they all compile.

Underscores are allowed between any two digits in a numeric literal. Underscores are not allowed adjacent to a decimal point, making Option B the correct answer.

41. Which of the following lists of primitive numeric types is presented in order from smallest to largest data type?

✓ **A** byte, short, int, long

B int, short, byte, long

C short, byte, int, long

D short, int, byte, long

i These four types represent nondecimal values. While you don't need to know the exact sizes, you do need to be able to order them from largest to smallest. A byte is smallest. A short comes next, followed by int and then long. Therefore, Option A is correct.

42.

```
package animal;
public class Cat {
  public String name;
  public static void main(String[] meow) {
    Cat cat = new Cat();
    ____________________ = "Sadie";
  }
}
```

Fill in the blank to make the code compile:

✓ **A** cat.name

B cat-name

C cat.setName

D cat[name]

i Java uses dot notation to reference instance variables in a class, making Option A correct.

43. Which of the following is the output of this code, assuming it runs to completion?

A play-

✓ **B** play-play-

C play-clean-play-

D play-play-clean-clean-

```
package store;
public class Toy {
   public void play() {
      System.out.print("play-");
   }
   public void finalizer() {
      System.out.print("clean-");
   }
   public static void main(String[] fun) {
      Toy car = new Toy();
      car.play();
      System.gc();
      Toy doll = new Toy();
      doll.play();
   }
}
```

i If there was a finalize() method, this would be a different story. However, the method here is finalizer. Tricky! That's just a normal method that doesn't get called automatically. Therefore clean is never output.

44.

```
public class Penguin {

   private double beakLength;

   public static void setBeakLength(Penguin p, int b) {
      ____________________________
   }
}
```

Which is the most common way to fill in the blank to implement this method?

- ✓ **A** p.beakLength = b;
- **B** p['beakLength'] = b;
- **C** p[beakLength] = b;
- **D** None of the above

i Options B and C do not compile. In Java, braces are for arrays rather than instance variables. Option A is the correct answer. It uses dot notation to access the instance variable. It also shows that a private variable is accessible in the same class and that a narrower type is allowed to be assigned to a wider type.

45.

```
________ first = Integer.parseInt("5");
________ second = Integer.valueOf("5");
```

Fill in the blanks to indicate whether a primitive or wrapper class can be assigned without the compiler using the autoboxing feature.

- **A** int, int
- ✓ **B** int, Integer
- **C** Integer, int
- **D** Integer, Integer

i The parseInt() methods return a primitive. The valueOf() methods return a wrapper class object. In real code, autoboxing would let you assign the return value to either a primitive or wrapper class. In terms of what gets returned directly, Option B is correct.

46.

```
1:   public class Person {
2:      public Person youngestChild;
3:
4:      public static void main(String... args) {
5:         Person elena = new Person();
6:         Person diana = new Person();
7:         elena.youngestChild = diana;
8:         diana = null;
9:         Person zoe = new Person();
10:        elena.youngestChild = zoe;
11:        zoe = null;
12:     }
13:  }
```

How many objects are eligible for garbage collection right before the end of the main method?

A None

✓ B One

C Two

D Three

i On line 9, all three objects have references. The elena and zoe objects have a direct reference. The diana object is referenced through the elena object. On line 10, the reference to the diana object is replaced by a reference to the zoe object. Therefore, the diana object is eligible to be garbage collected, and Option B is correct.

47.

```
public class TennisBall {
}
```

Which is a valid constructor for this class?

A public TennisBall static create() { return new TennisBall(); }

B public TennisBall static newInstance() { return new TennisBall():}

✓ C public TennisBall() {}

D public void TennisBall() {}

i Options A and B are static methods rather than constructors. Option D is a method that happens to have the same name as the class. It is not a constructor because constructors don't have return types.

48. Which of the following is not a possible output of this code, assuming it runs to completion?

```
package store;
public class Toy {
   public void play() {
      System.out.print("play-");
   }
   public void finalize() {
      System.out.print("clean-");
   }
   public static void main(String[] args) {
      Toy car = new Toy();
      car.play();
      System.gc();
      Toy doll = new Toy();
      doll.play();
   }
}
```

✓ A play-

B play-play-

C play-play-clean-

D play-play-clean-clean-

i Remember that garbage collection is not guaranteed to run on demand. If it doesn't run at all, Option B would be output. If it runs at the requested point, Option C would be output. If it runs right at the end of the main() method, Option D would be output. Option A is the correct answer because play is definitely

called twice. Note that you are unlikely to see all these scenarios if you run this code because we have not used enough memory for garbage collection to be worth running. However, you still need to be able to answer what could happen regardless of it being unlikely.

49. Which converts a primitive to a wrapper class object without using autoboxing?

A Call the asObject() method

✓ B Call the constructor of the wrapper class

C Call the convertToObject() method

D Call the toObject() method

i Each wrapper class has a constructor that takes the primitive equivalent. The methods mentioned in Options A, C, and D do not exist.

50. What is the output of the following?

```
package beach;
public class Sand {
   public Sand() {
      System.out.print("a");
   }
   public void Sand() {
      System.out.print("b");
   }
   public void run() {
      new Sand();
      Sand();
   }
   public static void main(String... args) {
      new Sand().run();
   }
}
```

A a

B ab

✓ C aab

D None of the above

i The main() method calls the constructor which outputs a. Then the main method calls the run() method. The run() method calls the constructor again, which outputs a again. Then the run() method calls the Sand() method, which happens to have the same name as the constructor. This outputs b. Therefore, Option C is correct.