socrative by MasteryConnect

Quiz name: Java 111 Chapter 9, 17 Quiz (from version 1)
Question with Most Correct Answers: #8
Question with Fewest Correct Answers: #23

Date: 12/01/2015
Total Questions: 23

1. Instance variables are variables declared inside a method or method parameter.

2/11 (A) True
0/11 (B) True only in an abstract class
8/11 (C) False

---

2. Which of the following are true?

1/11 (A) If an object reference is declared as a local variable it goes on the heap.
8/11 (B) Local variables live on the stack in the frame corresponding to the method where the variables are declared.
7/11 (C) All objects live in the heap regardless of whether the reference is a local or instance variable.
11/11 (D) Instance variables are variables declared inside a class but outside any method.

---

3. In the code example below, which is the object reference variable?

0/11 (A) Duck()
0/11 (B) 24
11/11 (C) d
0/11 (D) new Duck()
0/11 (E) Duck

```
public class StackRef {

    public void run() {
        build();
    }

    public void build() {
        Duck d = new Duck(24);
    }
```

---

4. In the code example below, where will the Duck object live?

10/11 (A) Heap
0/11 (B) Stack
1/11 (C) Frame

```
public class StackRef {

    public void run() {
        build();
    }

    public void build() {
        Duck d = new Duck(24);
    }
```

---

5. The currently executing method is the one on the top of the stack.

11/11 (A) True
0/11 (B) False

---

6. When is an object eligible for garbage collection?

5/11 A When the reference is assigned to another object

6/11 B When the reference is set to null

1/11 C As soon as it is instantiated

9/11 D When the reference goes out of scope (it is no longer pointing to the object)

---

7. In the code example below, how long will "d" live on the stack?

```
public class StackRef {

    public void run() {
        build();
    }

    public void build() {
        Duck d = new Duck(24);
    }
```

0/11 A Until run() pops off the stack

11/11 B Until build() pops off the stack

0/11 C Until the jvm is restarted

0/11 D d will live in the heap, not the stack

---

8. In the code example below, how long will "d" live on the stack?

```
18  public class StackRef {
19
20      public void run() {
21          Duck d = new Duck();
22          build();
23      }
24
25      public void build() {
26
27      }
28  }
```

11/11 A Until run() pops off the stack

0/11 B Until build() is added to the top of the stack

0/11 C Until build() pops off the stack

0/11 D Until d is garbage collected

---

9. What, if anything, is wrong with this code snippet?

```
12  public class MovieCharacters {
13
14      public void run() {
15          Gremlin gizmo = new Gremlin();
16          build();
17      }
18
19      public void build() {
20          gizmo = new Gremlin();
21      }
```

0/11 A The constructor for Gremlin is missing

7/11 B gizmo is "scoped" only to the run method, so it can't be used anywhere else

1/11 C Nothing is wrong with this code

0/11 D The run() method should not have a return type of void

3/11 E When the build() method runs, gizmo from line 15 will have been garbage collected

---

10. Which of the following are true?

10/11 A A constructor is the code that runs when somebody says "new" on a class type, like this: Duck d = new Duck();

1/11 B A constructor must have the same name as the class and a return type of New

11/11 C If you do not put a constructor in your class, the compiler creates a default constructor

11/11 D The default constructor created by the compiler has no arguments.

---

11. You can have more than one constructor in your class as long as the argument lists are different. This means you have overloaded constructors.

11/11 A True

0/11 B False

---

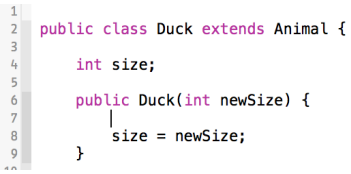
12. All constructors in an object's inheritance tree must run when you make a new object.

6/11 (A) True

5/11 (B) False

---

13. Type the code that should be entered on line 7 to call the Duck's super constructor.



```
1
2  public class Duck extends Animal {
3
4      int size;
5
6      public Duck(int newSize) {
7          |
8          size = newSize;
9      }
```

Anon 1ed62

Super Animal();

Anon df246

super();

Anon 82f1a

this.Duck()

Anon aff61

Animal duck = new Animal();

Anon 1aa67

super();

Anon 8e35d

I can't remember

Anon 9dd1b

super();

Anon 7d38c

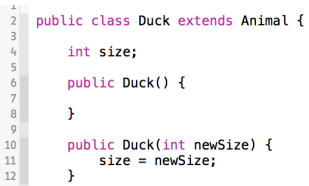
super.Animal()

Anon 30d7e

super();

---

14. Type the code that should appear on line 7 to call the Duck's single arg constructor with a parameter of 14.



```
2  public class Duck extends Animal {
3
4      int size;
5
6      public Duck() {
7
8      }
9
10     public Duck(int newSize) {
11         size = newSize;
12     }
```

Anon 1ed62

this.Duck(14);

Anon df246

this(14);

Anon 6a0bc

this.Duck(14)

Anon 82f1a

this(14);

Anon aff61

new Duck(007);

Anon 1aa67

this(14);

Anon 8e35d

this();

Anon 9dd1b

this(24);

Anon 7d38c

this(14);

Anon 30d7e

this();

---

15. A constructor can have a call to super() OR this(), but NEVER both.

5/11 (A) True

6/11 (B) False

---

16. It is good practice to keep source code and compiled code (class files) separate, but there is no way to do this in Java.

0/11 (A) True

11/11 (B) False

---

17. One key feature of using packages is to prevent class name conflicts.

11/11 (A) True

0/11 (B) False

---

18. A class must be put into a directory structure that matches the package hierarchy.

11/11 (A) True

0/11 (B) False

---

19. My class Book that has a package structure of java111.project5.labs and "lives" in the projects/src/java111/project5/labs directory. What is the proper way to compile Book into its proper package structure in the classes directory (assume projects/classes/java111/project5/labs)?

1/11 (A) cd to the labs directory, then type: javac Book.java

6/11 (B) cd to the projects directory, then type: javac -classpath classes -d classes java111/project5/labs/Book.java

0/11 (C) cd to the projects directory, then type: javac java111/project5/labs5/Book.java

4/11 (D) cd to the projects directory, then type: javac -classpath classes -d classes java111.project5.labs.Book.java

---

20. Given, "javac -classpath classes -d classes java111/project5/labs/Book.java", what does the -d parameter do?

0/11	A	Tells the compiler to debug the Book class
0/11	B	Tells the compiler to build the directories java111, project5, labs in the proper structure if they do not already exist
0/11	C	Tells the compiler to send the compiled classes to classes/java111/project5/labs/
3/11	D	All of the above
8/11	E	Answers 2 and 3 are both correct

---

21. How can I run my Book class which resides in projects/classes/java111/project5/labs/Book.class

0/11	A	cd to the src directory and type: java Book
0/11	B	cd to the classes directory and type: java Book
0/11	C	cd to the projects directory and type: java java111.project5.labs.Book
10/11	D	cd to the projects directory and type: java -classpath classes java111.project5.labs.Book
1/11	E	cd to the projects directory and type: java -classpath classes Book

---

22. Which of the following are valid javadoc comments?

0/11	A	// @author aSchmidt
0/11	B	//* @author aSchmidt *//
11/11	C	/** @author aSchmidt */
0/11	D	/* @author aSchmidt */

---

23. Javadoc can be created for all classes in my java111.project5 package using what command from my projects directory?

2/11	A	javadoc -d docs -sourcepath src java111.project5
1/11	B	javadoc -d docs -sourcepath src/java111.*
5/11	C	javadoc -d docs -sourcepath src java111/project5
3/11	D	javadoc -d docs java111.project5.*
0/11	E	It's not possible, javadoc can only be run for one class at a time