

Java 111 Chapter 9, 17 Quiz

Total Questions: 23

Least Correct Answers: #8

Most Correct Answers: #8

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

- **0/2** (A) True
- 0/2 (B) True only in an abstract class
- 1/2 C False

2. Which of the following is are true?

- 0/2 (A) If an object reference is declared as a local variable it goes on the heap.
- Local variables live on the stack in the frame corresponding to the method where the variables are declared.
- All objects live in the heap regardless of whether the reference is a local or instance variable.
- 1/2 Instance variables are variables declared inside a class but outside any method.

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

- **0/2** (A) Duck()
- **0/2** (B) 24
- 1/2 C d
- **0/2** D new Duck()
- 0/2 (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?

- 1/2 A Heap
- **0/2** (B) Stack
- **0/2** (c) Frame

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

5. The currently executing method is located where in memory?

- 0/2 (A) Bottom of the heap
- **0/2** (B) Top of the heap
- **0/2** (c) Bottom of the stack
- 1/2 D Top of the stack

6. When is an object eligible for garbage collection?

- 1/2 A When the reference is assigned to another object
- 1/2 B When the reference is set to null

- 0/2 (C) As soon as it is instantiated
- 1/2 When the reference variable 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?
- 0/2 (A) Until run() pops off the stack
- 1/2 B Until build() pops off the stack
- 0/2 (c) Until the jvm is restarted
- 0/2 D d will live in the heap, not the stack

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

- 8. In the code example below, how long will "d" live on the stack?
- 1/2 A Until run() pops off the stack
- 0/2 (B) Until build() is added to the top of the stack
- 0/2 (c) Until build() pops off the stack
- 0/2 D Until d is garbage collected

```
public class StackRef {

public void run() {

Duck d = new Duck();

build();

public void build() {

public void b
```

- 9. What, if anything, is wrong with this code snippet?
- 0/2 (A) The constructor for Gremlin is missing
- gizmo is "scoped" only to the run method, so it can't be used anywhere else

12 public class MovieCharacters {

- **0/2** (c) Nothing is wrong with this code
- 0/2 D The run() method should not have a return type of void
- 0/2 (E) When the build() method runs, gizmo from line 15 will have been garbage collected
 - 10. Which of the following are true?
- A constructor is the code that runs when somebody says "new" on a class type, like this: Duck d = new Duck();
- 0/2 (B) A constructor must have the same name as the class and a return type of New
- 1/2 If you do not put a constructor in your class, the compiler creates a default constructor
- 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.
- 1/2 A True
- **0/2** (B) False
- 12. All constructors in an object's inheritance tree must run when you make a new object.
- 1/2 A True
- **0/2** (B) False

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

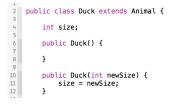
Anon anoneb8e3621de614e75

```
super();
```

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

Anon anoneb8e3621de614e75

```
this(14);
```



- 15. A constructor can have a call to super() OR this(), but NEVER both.
- 1/2 A True
- 0/2 (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.
- **0/2** (A) True
- 1/2 B False
 - 17. One key feature of using packages is to prevent class name conflicts.
- 1/2 A True
- **0/2** B False
- 18. You have a class called Project1 in this directory structure: src/edu/madisoncollege/javaprojects. Write the package statement that should appear at the top of the Project1 class.

Anon anoneb8e3621de614e75

- ✓ package edu.madisoncollege.javaprojects;
- 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)?
- 0/2 (A) cd to the labs directory, then type: javac Book.java
- cd to the projects directory, then type: javac -classpath classes -d classes java111/project5/labs/Book.java
- 0/2 C cd to the projects directory, then type: javac java111/project5/labs5/Book.java

