05/03/2017

Java 111 Final Exam Review (from version 1)

Total Questions: 28

Most Correct Answers: **#8**

Least Correct Answers: **#16**

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

1/14 (A) True

0/14 (B) True only in an interface

0/14 (C) True only in an abstract class

12/14 [✓ D] False

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

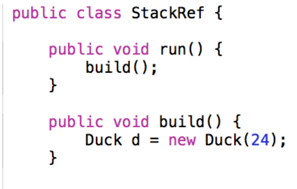
0/14 (A) Duck()

0/14 (B) 24

11/14 [✓ C] d

0/14 (D) new Duck()

2/14 (E) Duck



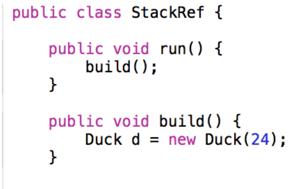
```
public class StackRef {

    public void run() {
        build();
    }

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

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

11/14 [✓ A] Heap

2/14 (B) Stack

0/14 (C) Frame



```
public class StackRef {

    public void run() {
        build();
    }

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

4. The currently executing method is the one on the bottom of the stack.

2/14 (A) True

11/14 [✓ B] False

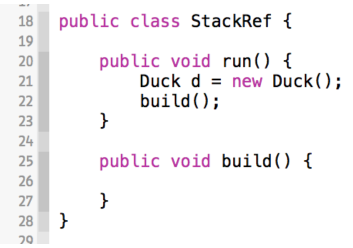
5. In the code example below, how long will "d" live on the stack?

11/14 [✓ A] Until run() pops off the stack

0/14 (B) Until build() is added to the top of the stack

1/14 (C) Until build() pops off the stack

1/14 (D) Until d is garbage collected



```
public class StackRef {

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

    public void build() {

    }
}
```

6. Which of the following are true?

12/14 ● A A constructor is the code that runs when somebody says "new" on a class type, like this: Duck d = new Duck();

13/14 ● B A constructor must have the same name as the class and no return type

1/14 ○ C If you add a 2-argument constructor to a class, the compiler will create a default constructor

12/14 ● D The default constructor created by the compiler has no arguments.

7. Constructors cannot be overloaded.

1/14 ○ A True

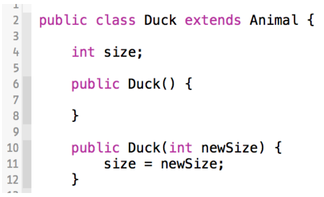
12/14 ● B False

8. A constructor on an object's parent must run when you make a new object.

13/14 ● A True

0/14 ○ B False

9. 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 anon0673329aafc546f3

✗ this();

Anon anon09fdeb0816284bdf

✗ Duck(14);

Anon anon1c2377c8e444487c

✓ this(14);

Anon anon2103533d9ffd4adf

✗ Duck d = new Duck(14);

Anon anon2f9711c9066346ad

✗ size = 14;

Anon anon5e271d568c1146c1

✗ size = 0;

Anon anon607e18d253d844c4

✗ Duck duck = new Duck(14);

Anon anonadcf6b9f19a94005

✗ this

Anon anonbbbaee7192fe4443

✗ size= 14;

Anon anoncd2340c674d04489

✗ this.size(14);

Anon anond2832aaaf6484a39

✓ this(14);

Anon anond6279b4fb7b14186

✓ this(14);

Anon anonea90b944efd94d0c

✗ newSize(14)

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

10/14 [✓ A] True

3/14 B False

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

13/14 [✓ A] True

0/14 B False

12. The keyword "static" lets a method run without any instance of the class.

12/14 [✓ A] True

1/14 B False

13. A static method cannot be dependent on any instance variable.

9/14 [✓ A] True

4/14 B False

14. Static final variables are also known as:

1/14 A Constructor args

1/14 B Primitives

0/14 C Booleans

11/14 [✓ D] Constants

0/14 E None of the above

15. Which of the following are true?

13/14 [✓ A] The naming convention for constants is to make the name all uppercase.

12/14 [✓ B] A final class cannot be extended (subclassed).

12/14 [✓ C] A static method can access a static variable.

2/14 D A final method can be overridden.

0/14 E None of the above

16. An abstract class can only have abstract methods.

0/14 A True

0/14 [✓ B] False

17. Marking a class with the "abstract" keyword prevents a developer from instantiating that class.

11/14 [✓ A] True

2/14 B False

18. Which of the following are true?

4/14 A An interface must be created using the keyword "abstract".

11/14 [✓ B] An interface defines only abstract methods.

12/14 [✓ C] A class can implement multiple interfaces.

10/14 [✓ D] All interface methods are implicitly public.

0/14 E None of the above.

19. Write an abstract method called calculatePremium that two parameters: a double for rate and an int for term. The method has returns a double.

Anon anon0673329aafc546f3

✗
```
public abstract class calculatePremium(double rate, int term) {
return double
}
```

Anon anon09fdeb0816284bdf

✗
```
public abstract double calculatePremium(double rate, int term) {
return rate * term
}
```

Anon anon1c2377c8e444487c

✓
```
public abstract double calculatePremium(double rate, int term);
```

Anon anon2103533d9ffd4adf

✗
```
public abstract calculatePremium(Double rate, int term) {
this.rate = rate;
this.term = term;

return rate;
}
```

Anon anon2f9711c9066346ad

✗ `public abstract calculatePremium(double rate, int term);`

Anon anon5e271d568c1146c1

✗ `abstract public double calculatePremium(double rate, int term);`

Anon anon607e18d253d844c4

✓ `public abstract double calculatePremium(double rate, int term);`

Anon anonadcf6b9f19a94005

✓ `public abstract double calculatePremium(double rate, int term);`

Anon anonbbbaee7192fe4443

✗
```
public abstract calculatePremium(double rate, int term
) {

return rate;
}
```

Anon anoncd2340c674d04489

✗ `public abstract calculatePremium(double rate, int term) {returns double;}`

Anon anond2832aaaf6484a39

✗ `public abstract double calculatePremium(double rate, int term) {}`

Anon anond6279b4fb7b14186

✓ `public abstract double calculatePremium(double rate, int term);`

Anon anonea90b944efd94d0c

✗
```
public abstract double calculatePremiun(double rate, int term) {
return rate;
}
```

20. If you override a superclass method in a subclass, you cannot invoke (call) the superclass method.

2/14 (A) True

11/14 [✓ B] False

21. Given the following: SuperStarCoder rockstar = new JavaRockStar(); what is the object reference type?

2/14 A JavaRockStar

1/14 B rockstar

10/14 [✓ C] SuperStarCoder

0/14 D new

0/14 E None of the above

22. Given the following: SuperStarCoder rockstar = new JavaRockStar(); what is the actual object type?

13/14 [✓ A] JavaRockStar

0/14 B rockstar

0/14 C SuperStarCoder

0/14 D new

0/14 E None of the above

23. What is the proper way to create an interface called Payable?

0/14 A public abstract interface class Payable {}

0/14 B public abstract Payable {}

0/14 C public abstract interface class Payable extends Payable {}

12/14 [✓ D] public interface Payable {}

0/14 E None of the above

24. Which of the following are true?

13/14 [✓ A] You can write a new instance method in the subclass that has the same signature as the one in the superclass, thus overriding it.

13/14 [✓ B] You can declare new methods in the subclass that are not in the superclass.

13/14 [✓ C] You can declare new fields in the subclass that are not in the superclass.

7/14 [✓ D] A subclass cannot extend multiple superclasses.

5/14 E A subclass inherits the private instance variables and private methods of its superclass.

25. A superclass called Fruit contains a method called display() that outputs a message to the terminal. Which code segment IN THE SUBCLASS Apple will successfully call that method?

13/14 A super.display();

0/14 B Fruit.display();

0/14 C display(Fruit);

0/14 D Apple.display;

0/14 E this.display();

26. Which of the following are true:

11/14 A The arguments and return types of an overriding method must look to the outside world exactly like the overridden method in the superclass.

10/14 B Overloaded methods have the same name, but different argument lists.

6/14 C Overloaded methods can have different return types as long as the arguments lists are different.

5/14 D When overriding a public method, you can declare the method as private.

2/14 E All of the above are true.

27. An import statement saves you from having to type out the full name of classes.

9/14 A True

4/14 B False

28. What is the proper way to add a Kumquat object to an ArrayList of Kumquats called myKumquatList? Assume I have created a Kumquat object like this: Kumquat myKumquat = new Kumquat();

0/14 A myKumquatList.kumquat = myKumquat;

0/14 B myKumquatList[0] = myKumquat;

13/14 C myKumquatList.add(myKumquat);

0/14 D myKumquatList[1].add(myKumquat)