CCEE mock test

Module name : Object Oriented Programming with Java - test 2

0:59:56

Q.1. What is the output of the below code snippet:

```
class Base {

    public void fun() {
        System.out.println("Base fun()");
    }

    public void fun1() {
        System.out.println("Base fun1()");
        fun();
    }

}

class derived extends Base {

    public void fun() {
        System.out.println("Derived fun()");
    }
}

public class MainClass {

    public static void main(String[] args){
        derived d1 = new Base();
        d1.fun1();
    }
}
```

- Error: Type mismatch: cannot convert from Base to derived
- Base fun1()
  Derived fun()
- Base fun1()
  Base fun()
- Error : cannot call the method of the base class

Q.2. What will happen when you attempt to compile and run this code

```
class Base {

    public final void method1() {
        System.out.println("method");
    }
}

public class Fin extends Base {

    public static void main(String args[]){
        Base b = new Base();
        b.method1();
    }
}
```

- It will print method
- Compile time error
- Runtime error
- It will compile but doesn't print anything

Q.3. Which assignments are illegal?

- long test = 012;
- float f = -412;
- int other = (int)true;
- double d = 0x12345678;

Q.4. What will happen when you attempt to compile and run this code

```
public class Over {

    public static void main(String[] args){
        Under u = new Under();
        u.test();
    }

    int test(){
        System.out.println("over");
        return 1;
    }
}

class Under extends Over {

    int test(){
        super.test();
        System.out.println("Under");
        return 1;
    }
}
```

- This code compiles, runs and displays over followed by Under
- This code compiles, runs and displays Under followed by over
- This code does not compile
- Code will compile but gives runtime error

Q.5. What will be the output of this code:

```
public class TestClass {

    public static void main ( String args[] ){
        Strings s = "hello";
        StringBuffer sb = new StringBuffer("hello");

        if(s == sb.toString()){
            System.out.println("Equal");
        }
        else
            System.out.println("Not Equal");
    }
}
```

- Equal
- Not Equal
- Compile time error
- Nothing will print

Q.6. What will be the output of this code:

```
public class Exceptiondemo {

    public static void main ( String args[] ){
        int i = 1, j = 1;
        try {
            i++;
            j--;
            if( i / j > 1)
```

```
                i++;
        }
        catch (Exception e){
            System.out.println("Exception");
        }
        catch (ArithmeticException e){
            System.out.println("Arithmetic Exception");
        }
        catch (ArrayIndexOutOfBoundsException e){
            System.out.println("Array Index Exception");
        }
        finally{
            System.out.println("finally");
        }
    }
}
```

- Exception
- Arithmetic Exception
- Array Index Exception
- Compile time error: exception ArithmeticException has already been caught

Q.7. What will happen when you attempt to compile and run this code

```
public class Test {

    public static void main( String args[] ){
        int a = 5;
        System.out.println( cube(a) );
    }

    int cube ( int theNum ){
        return theNum * theNum * theNum;
    }
}
```

- 125
- 5
- cube(a)
- Compile time error

Q.8. What will be the output of this code:

```
public class Tester {
    int var;

    Tester (double var){
        this.var = (int)var;
    }

    Tester(int var){
        this("hello");
    }

    Tester(String s){
        this();
        System.out.println(s);
    }

    Tester(){
        System.out.println("good-bye");
    }

    public static void main( String[] args ){
        Tester t = new Tester(5);
    }
}
```

- good-bye
  hello
- hello

- 5
- hello
  good-bye

Q.9. Which of the following statement is false?

- If a class has any abstract methods it must be declared abstract itself.
- All methods in an abstract class must be declared as abstract
- When applied to a class, the final modifier means it cannot be sub-classed
- transient and volatile are Java modifiers

Q.10. What will be the output of this code:

```
public class RTException {

    public static void throwit () {
        System.out.print("throw it ");
        throw new RuntimeException();
    }

    public static void main( String[] args){
        try{
            System.out.print("hello ");
            throwit();
        }
        catch (Exception re){
            System.out.print("caught ");
        }
        finally {
            System.out.print("finally ");
        }
        System.out.println("after ");
    }
}
```

- hello throw it caught finally after
- hello caught
- hello finally after
- Compile time error

Q.11. Which collection class allows you to access its elements by associating a key with an element's value, and provides synchronization?

- java.util.SortedMap
- java.util.TreeMap
- java.util.TreeSet
- java.util.HashTable

Q.12. Which statement is wrong?

- Interface can extend an interface.
- We can define a variable inside an interface.
- Interface can implement an interface.
- Interface is pure abstract class.

Q.13. What access control keyword should you use to allow other classes to access a method freely within its package, but to restrict classes outside of the package from accessing that method?

- public
- protected
- private
- do not supply an access control keyword

Q.14. Objects are passed by value or reference?

- By value
- By reference
- it depends upon how you specify
- None of the above

Q.15. If you write System.exit(0) at the end of try block, will the finally block still execute?

- Yes
- No
- It depends upon return statement
- Can't say

Q.16. Which is not a method of Object class?

- toString
- clone
- equals
- compare

Q.17. Which is valid declaration of a String?

- String s2 = 'null';
- String s3 = (String) 'abc';
- String s1 = null;
- Strings4 = (String) 'g'

Q.18. Which is valid declaration within an interface?

- public static short stop = 23;
- protected short stop = 23;
- transient short stop = 23;
- final void madness (short stop);

Q.19. What will be the output of this code:

```
public class Equals{

    public static void main(String[] args){
        int x = 100;
        double y = 100.1;
        boolean b = (x == y);
        System.out.println(b);
    }
}
```

- true
- false
- Compile time error
- Runtime error

Q.20. Which collection class allows you to grow or shrink its size and provides indexed access to its elements, but whose methods are not synchronised?

- java.util.HashSet
- java.util.LinkedHashSet
- java.util.List
- java.util.ArrayList

Q.21. What will be the output of this code:

```
String x = "xyz";
x.toUpperCase();
String y = x.replace('Y', 'y');
y = y + "abc";
System.out.println(y);
```

- abcXyZ
- XyZabc
- xyzabc
- compilation fails

Q.22. How can you serialize an object?

- You have to make the class of the object implement the interface Serializable.
- You must call the method serializeObject() (which is inherited from class Object) on the object.
- You should call the static method serialize(Object obj) from class Serializer, with as argument the object to be serialized.
- You don't have to do anything, because all objects are serializable by default.

Q.23. What will be the output of this code:

```
class X {
    Y b = new Y();
    X() {
        System.out.print("X");
```

```
    }
}
class Y {
    Y() {
        System.out.print("Y");
    }
}
public class Z extends X {
    Y y = new Y();
    Z() {
        System.out.print("Z");
    }

    public static void main(String[] args) {
        new Z();
    }
}
```

- Z
- YZ
- XYZ
- YXYZ

Q.24. Which of the following statements are true?

- static methods do not have access to the implicit variable called this
- A static method may be called without creating an instance of its class
- All of Above
- None of Above

Q.25. Which of the following are methods of the Runnable interface?

- run
- start
- yield
- stop

Q.26. What are a native methods?

- Native methods are usually used to interface with system calls or libraries written in other programming languages.
- Native methods are as same like as abstract method.
- Both of these.
- None of these.

Q.27. What will be the output of this code:

```
class Base {
    void show(){
        System.out.println("Base Class");
    }
}

class Derived {
    void show(){
        System.out.println("Derived Class");
    }
}
```

```
public class MainClass {
    public static void main(String[] args) {
        Base b1 = new Base();
        Derived d1 = new Derived();
        b1.show();
        d1.show();
        Base b2 = d1;
        b2.show();
        d1.show();
    }
}
```

- Error : Cannot convert Base to Derived
- Base Class
  Derived Class
  Base Class
  Derived Class
- Base Class
  Derived Class
  Derived Class
  Base Class
- Base Class
  Derived Class
  Derived Class
  Derived Class

Q.28. String Class is

- is final
- is public
- is serializable
- All of the above

Q.29. Which keyword when applied on a method indicates that only one thread should execute the method at a time. Select the one correct answer.

- transient
- static
- final
- Synchronized

Q.30. which method is used to know the element add the top of the stack?

- pop()
- top()
- peek()
- search()

Q.31. Which interface provides the capability to store objects using a key-value pair?

- Java.util.Map
- Java.util.Set

- Java.util.List
- Java.util.Collection

Q.32. Which of the following statements about arrays is syntactically wrong?

- Person[] p = new Person[5];
- Person p[5];
- Person[] p [];
- Person p [][] = new Person[2][];

Q.33. Process of writing the state of object into byte stream

- Deserialisation
- Externlisation
- serialization
- print

Q.34. To display text on the applet _____ method is used.

- showString()
- drawString()
- println()
- printString()

Q.35. Which of the following are true about interfaces.

- Methods declared in interfaces are implicitly private.
- Variables declared in interfaces are implicitly public, static, and final.
- The keyword implements indicate that an interface inherits from another.
- An interface can not extend any number of interfaces.

Q.36. What is the process of defining two or more methods within same class that have same name but different parameters declaration?

- method overloading
- method overriding
- method hiding
- none of the mentioned

Q.37. Which package contains all the classes and interfaces for the Collection framework

- java.lang
- java.collections

- java.util
- java.io

Q.38. Which of the below is invalid identifier with the main method?

- public
- static
- private
- final

Q.39. Which keyword is used by the method to refer to the object that invoked it?

- import
- catch
- abstract
- this

Q.40. Which function is used to perform some action when the object is to be destroyed?

- finalize()
- delete()
- main()
- none of the mentioned

Submit