PREGUNTAS JAVA.

Contenido

1. Which three are bad practices?

a) Checking for an IOException and ensuring that the program can recover if one occurs.
b) Checking for ArrayIndexOutOfBoundsException and ensuring that the program can recover if one occurs.
c) Checking for FileNotFoundException to inform a user that a filename entered is not valid.
d) Checking for Error and, if necessary, restarting the program to ensure that users are unaware problems.
e) Checking for ArrayIndexOutOfBoundsException when iterating through an array to determine when all elements.-have been visited.

2. Indica a JUnit que la propiedad que usa esta anotación es una simulación y, por lo tanto, se inicializa como tal y es susceptible de ser inyectada por @InjectMocks.

Mockito /Mock

Inject /InjectMock

```
Given

public static void main(String[] args){
      int[][] array2D = {{0,1,2}, {3,4,5,6}};
      System.out.print(array2D[0].length + "");
      System.out.print(array2D[1].getClass().isArray() + "");
      System.out.print(array2D[0][1]);
}
What is the result?
```

a) 3 false3
b) 3 false1
c) 2 false1
d) 3 true1
e) 2 true3

3. Which two statments are true?

a) An interface CANNOT be extended by another interface.
b) An abstract class CANNOT be extended by an abstract class.
c) An interface can be extended by an abstract class.
d) An abstract class can implement an interface.
e) An abstract class can be extended by an interface.
f) An abstract class can be extended by a concrete class.

4. What is the result?

```
Given:

class Alpha{ String getType(){ return "alpha";}}
class Beta extends Alpha{String getType(){ return "beta";}}
public class Gamma  extends Beta { String getType(){ return "gamma";}
      public static void main(String[] args) {
        Gamma g1 = (Gamma) new Alpha();
        Gamma g2 = (Gamma) new Beta();
        System.out.print(g1.getType()+ " " +g2.getType());
    }
}
What is the result?
```

a) Gamma gamma
b) Beta beta
c) Alpha beta
d) Compilation fails

5. Which five methods, inserted independiently at line 5, will compile? (Choose five)

```
1 public class Blip{
2       protected int blipvert(int x){ return 0
3 }
4 class Vert extends Blip{
5       //insert code here
6 }
```

a) Private int blipvert(long x) { return 0; }

b) Protected int blipvert(long x) { return 0; }
c) Protected long blipvert(int x, int y) { return 0; }
d) Public int blipvert(int x) { return 0; }
e) Private int blipvert(int x) { return 0; }
f) Protected long blipvert(int x) { return 0; }
g) Protected long blipvert(long x) { return 0; }

6. Which two independently, will allow Sub to compile? (Choose two)

```
Given:

1. class Super{
2.    private int a;
3.    protected Super(int a){ this.a = a; }
4. }
...
11. class Sub extends Super{
12.   public Sub(int a){ super(a);}
13.   public Sub(){ this.a = 5;}
14. }
Which two independently, will allow Sub to compile? (Choose two)
```

a) Change line 2 to: public int a;
b) Change line 13 to: public Sub(){ super(5);}
c) Change line 2 to: protected int a;
d) Change line 13 to: public Sub(){ this(5);}
e) Change line 13 to: public Sub(){ super(a);}

Respuesta correcta

Change line 13 to: public Sub(){ super(5);}

Change line 13 to: public Sub(){ this(5);}

7. What is true about the class Wow?

```
public abstract class Wow {
        private int wow;
        public Wow(int wow) { this.wow = wow; }
        public void wow() { }
        private void wowza() { }
}
```

a) It compiles without error.
b) It does not compile because an abstract class cannot have private methods
c) It does not compile because an abstract class cannot have instance variables.
d) It does not compile because an abstract class must have at least one abstract method. It does not compile because an abstract class must have a constructor with no arguments.

8. What is the result?

```
class Atom {
        Atom() { System.out.print("atom "); }
}
class Rock extends Atom {
        Rock(String type) { System.out.print(type); }
}
public class Mountain extends Rock {
        Mountain() {
                super("granite ");
                new Rock("granite ");
        }
        public static void main(String[] a)  { new Mountain(); }
}
```

a) Compilation fails.
b) Atom granite.
c) Granite granite.
d) Atom granite granite.
e) An exception is thrown at runtime.
f) Atom granite atom granite

9. What is printed out when the program is excuted?

```
public class MainMethod {
        void main() {
                System.out.println("one");
        }
        static void main(String args) {
                System.out.println("two");
        }
        public static final void main(String[] args) {
                System.out.println("three");
        }
        void mina(Object[] args) {
                System.out.println("four");
        }
}
```

a) one
b) two
c) three
d) four
e) There is no output.

10. What is the result?

```
class Feline {
        public String type = "f ";
        public Feline() {
                System.out.print("feline ");
        }
}
public class Cougar extends Feline {
        public Cougar() {
                System.out.print("cougar ");
        }
        void go() {
                type = "c ";
                System.out.print(this.type + super.type);
        }
        public static void main(String[] args) {
                new Cougar().go();
        }
}
```

a) Cougar c f.
b) Feline cougar c c.
c) Feline cougar c f.
d) Compilation fails.
e) Respuesta correcta
f) Feline cougar c c.

11. What is the result?

```
class Alpha { String getType() { return "alpha"; } }
class Beta extends Alpha { String getType() { return "beta"; } }
public class Gamma extends Beta { String getType() { return "gamma"; }
        public static void main(String[] args) {
                Gamma g1 = new Alpha();
                Gamma g2 = new Beta();
                System.out.println(g1.getType() + " " + g2.getType());
        }
}
```

a) Alpha beta
b) Beta beta.
c) Gamma gamma.
d) Compilation fails.

12. What is the result?

```
import java.util.*;
public class MyScan {
        public static void main(String[] args) {
                String in = "1 a 10 . 100 1000";
                Scanner s = new Scanner(in);
                int accum = 0;
                for (int x = 0; x < 4; x++) {
                        accum += s.nextInt();
                }
                System.out.println(accum);
        }
}
```

a) 11

b) 111
c) 1111
d) An exception is thrown at runtime.

13. What is the result?

```
public class Bees {
        public static void main(String[] args) {
                try {
                        new Bees().go();
                } catch (Exception e) {
                        System.out.println("thrown to main");
                }
        }
        synchronized void go() throws InterruptedException {
                Thread t1 = new Thread();
                t1.start();
                System.out.print("1 ");
                t1.wait(5000);
                System.out.print("2 ");
        }
}
```

a) The program prints 1 then 2 after 5 seconds.
b) The program prints: 1 thrown to main.
c) The program prints: 1 2 thrown to main.
d) The program prints:1 then t1 waits for its notification.

14. Which statement is true?

```
class ClassA {
        public int numberOfInstances;
        protected ClassA(int numberOfInstances) {
                this.numberOfInstances = numberOfInstances;
        }
}
public class ExtendedA extends ClassA {
        private ExtendedA(int numberOfInstances) {
                super(numberOfInstances);
        }
        public static void main(String[] args) {
                ExtendedA ext = new ExtendedA(420);
                System.out.print(ext.numberOfInstances);
        }
}
```

a) 420 is the output.
b) An exception is thrown at runtime.
c) All constructors must be declared public.
d) Constructors CANNOT use the private modifier.
e) Constructors CANNOT use the protected modifier.

15. The SINGLETON pattern allows: *

a) Have a single instance of a class and this instance cannot be used by other classes
b) Having a single instance of a class, while allowing all classes have access to that instance.
c) Having a single instance of a class that can only be accessed by the first method that calls it.

16. What is the result?

```
import java.text.*;
public class Align {
        public static void main(String[] args) throws ParseException {
                String[] sa = {"111.234", "222.5678"};
                NumberFormat nf = NumberFormat.getInstance();
                nf.setMaximumFractionDigits(3);
                for (String s : sa) { System.out.println(nf.parse(s)); }
        }
}
```

a) 111.234 222.567
b) 111.234 222.568
c) 111.234 222.5678
d) An exception is thrown at runtime.

17. What is the result?

Given

```
public class SuperTest {
        public static void main(String[] args) {
                //statement1
                //statement2
                //statement3
        }
}
class Shape {
        public Shape() {
                System.out.println("Shape: constructor");
        }
        public void foo() {
                System.out.println("Shape: foo");
        }
}
class Square extends Shape {
        public Square() {
                super();
        }
        public Square(String label) {
                System.out.println("Square: constructor");
        }
        public void foo() {
                super.foo();
        }
        public void foo(String label) {
                System.out.println("Square: foo");
        }
}
```

Imagen sin leyenda

What should statement1, statement2, and statement3, be respectively, in order to produce the result?

Shape: constructor
Shape: foo
Square: foo

a) Square square = new Square ("bar"); square.foo ("bar"); square.foo();
b) Square square = new Square ("bar"); square.foo ("bar"); square.foo ("bar");
c) Square square = new Square (); square.foo (); square.foo(bar);
d) Square square = new Square (); square.foo (); square.foo("bar");
e) Square square = new Square (); square.foo (); square.foo ();

18. Which three implementations are valid?

```
interface SampleCloseable {
        public void close() throws java.io.IOException;
}
```

a) class Test implements SampleCloseable { public void close() throws java.io.IOException { // do something } }
b) class Test implements SampleCloseable { public void close() throws Exception { // do something } }
c) class Test implements SampleCloseable { public void close() throws FileNotFoundException { // do something } }
d) class Test extends SampleCloseable { public void close() throws java.io.IOException { // do something } }
e) class Test implements SampleCloseable { public void close() { // do something }}

19. What is the result?

```
class MyKeys {
        Integer key;
        MyKeys(Integer k) { key = k; }
        public boolean equals(Object o) {
                return ((MyKeys) o).key == this.key;
        }
}
```

And this code snippet:

```
Map m = new HashMap();
MyKeys m1 = new MyKeys(1);
MyKeys m2 = new MyKeys(2);
MyKeys m3 = new MyKeys(1);
MyKeys m4 = new MyKeys(new Integer(2));
m.put(m1, "car");
m.put(m2, "boat");
m.put(m3, "plane");
m.put(m4, "bus");
System.out.print(m.size());
```

a) 2
b) 3
c) 4

20. What value of x, y, z will produce the following result? 1234,1234,1234 -----, 1234, ------ *

```
public static void main(String[] args) {
        // insert code here
        int j = 0, k = 0;
        for (int i = 0; i < x; i ++) {
                do {
                        k = 0;
                        while (k < z) {
                                k++;
                                System.out.print(k + " ");
                        }
                        System.out.println(" ");
                        j++;
                } while (j < y);
                System.out.println("---");
        }
}
```

a) int x = 4, y = 3, z = 2;
b) int x = 3, y = 2, z = 3;
c) int x = 2, y = 3, z = 3;
d) int x = 2, y = 3, z = 4;
e) int x = 4, y = 2, z= 3;

21. Which three lines will compile and output "Right on!"?

```
13.  public class Speak {
14.      public static void main(String[] args) {
15.          Speak speakIT = new Tell();
16.          Tell tellIt = new Tell();
17.          speakIT.tellItLikeItIs();
18.          (Truth) speakIT.tellItLikeItIs();
19.          ((Truth) speakIT).tellItLikeItIs();
20.          tellIt.tellItLikeItIs();
21.          (Truth) tellIt.tellItLikeItIs();
22.          ((Truth) tellIt).tellItLikeItIs();
23.      }
24.  }

class Tell extends Speak implements Truth {
    @Override
    public void tellItLikeItIs() {
        System.out.println("Right on!");
    }
}

interface Truth {
    public void tellItLikeItIs();
}
```

a) Line 17
b) Line 18
c) Line 19
d) Line 20
e) Line 21
f) Line 22

22. What is the result?

```
class Feline {
    public String type = "f";
    public Feline() {
        System.out.print(s: "feline ");
    }
}
public class Cougar extends Feline{
    public Cougar() {
        System.out.print(s: "cougar ");
    }
    void go(){
        String type = "c";
        System.out.print(this.type + super.type);
    }

    Run | Debug
    public static void main(String[] args) {
        new Cougar().go();
    }
}
```

a) Feline cougar c f
b) Feline cougar c c
c) Feline cougar f f
d) No compila

23. ¿Qué acción realiza el archivo de dependencia pom.xml?*

a) Revisa que versiones de dependencias se tienen con otros proyectos
b) Elimina las dependencias con otros proyectos

c) Recupera todas las dependencias con otros proyectos
d) Modifica las dependencias que se tienen con otros proyectos

24. ¿Cuál es el resultado?

```
import java.util.*;
public class App {
        public static void main(String[] args) {
                List p = new ArrayList();
                p.add(7);
                p.add(1);
                p.add(5);
                p.add(1);
                p.remove(1);
                System.out.println(p);
        }
}
```

a) [7, 5]
b) [7, 1]
c) [7, 5, 1]
d) [7, 1, 5, 1]

25. Which five methods, inserted independiently at line 5, will compile? (Choose five)

```
1 public class Blip{
2       protected int blipvert(int x){ return 0
3 }
4 class Vert extends Blip{
5       //insert code here
6 }
```

a) Public int blipvert(int x) { return 0; }
b) Protected long blipvert(int x) { return 0; }
c) Protected int blipvert(long x) { return 0; }
d) Private int blipvert(long x) { return 0; }
e) Protected long blipvert(int x, int y) { return 0; }
f) Private int blipvert(int x) { return 0; }
g) Protected long blipvert(long x) { return 0; }

26. What is the result?

```
Given:

1. class Super{
2.    private int a;
3.    protected Super(int a){ this.a = a; }
4. }
...
11. class Sub extends Super{
12.   public Sub(int a){ super(a);}
13.   public Sub(){ this.a = 5;}
14. }
Which two independently, will allow Sub to compile? (Choose two)
```

a) Change line 2 to: public int a;
b) Change line 13 to: public Sub(){ super(5);}
c) Change line 2 to: protected int a;
d) Change line 13 to: public Sub(){ this(5);}
e) Change line 13 to: public Sub(){ super(a);}

27. What is the result?

```
Given

public static void main(String[] args){
      int[][] array2D = {{0,1,2}, {3,4,5,6}};
      System.out.print(array2D[0].length + "");
      System.out.print(array2D[1].getClass().isArray() + "");
      System.out.print(array2D[0][1]);
}
What is the result?
```

a) 3false3
b) 3false1
c) 2false1
d) 3true1
e) 2true3

28. Which two statments are true?

a) An interface CANNOT be extended by another interface.
b) An abstract class can be extended by a concrete class.
c) An abstract class CANNOT be extended by an abstract class.
d) An interface can be extended by an abstract class.
e) An abstract class can implement an interface.
f) An abstract class can be extended by an interface.

29. What is the result?

```
Given:

class Alpha{ String getType(){ return "alpha";}}
class Beta extends Alpha{String getType(){ return "beta";}}
public class Gamma  extends Beta { String getType(){ return "gamma";}
      public static void main(String[] args) {
        Gamma g1 = (Gamma) new Alpha();
        Gamma g2 = (Gamma) new Beta();
        System.out.print(g1.getType()+ " " +g2.getType());
    }
}
What is the result?
```

a) Gamma gamma
b) Beta beta
c) Alpha beta
d) Compilation fails

30. Which three are bad practices?

a) Checking for an IOException and ensuring that the program can recover if one occurs.
b) Checking for ArrayIndexOutOfBoundsException and ensuring that the program can recover if one occurs.
c) Checking for FileNotFoundException to inform a user that a filename entered is not valid.
d) Checking for Error and, if necessary, restarting the program to ensure that users are unaware problems.
e) Checking for ArrayIndexOutOfBoundsException when iterating through an array to determine when all elements.-have been visited.

31. Las 3 principales partes de un task *

a) Chunk, processing, output
b) Input, processing, output
c) Input, load, processing, Output
d) Input, Load, Output

32. En batch, cada step tiene *

a) itemInput, itemProcessor y itemWriter
b) itemReader, itemProcessor y itemWriter
c) itemReader, itemProcessor y item Output

What is the result?

What is the result?

What is the result?

What is the result?