

Exámenes Completos

1. ¿Cuál es la diferencia entre una clase abstracta y una interfaz en Java 8?

Una clase abstracta y una interfaz tienen propósitos similares, proporcionar un molde de un objeto sin la posibilidad de instanciar, sin embargo, la diferencia principal radica en que una clase abstracta puede contener métodos abstractos y/o métodos concretos, los cuales pueden definir algún comportamiento específico. Desventaja de clase abstracta: La clase ya no puede heredar de ninguna otra clase, pues en Java se maneja una sola herencia Ventaja de clases abstractas: Se pueden definir constructores en las clases abstractas Ventaja de las interfaces: Permite la implementación de n interfaces Desventaja de interfaces: No se pueden definir constructores en las interfaces.

2. ¿Qué muestra el siguiente código fuente por pantalla?

```
int x = 1;
switch (x){
    case 1:
        System.out.println("Uno");
    case 2:
        System.out.println("Dos");
    case 3:
        System.out.println("Tres");
    default:
        System.out.println("Otro número");
}
```

- a) Dos
- b) Uno Dos Tres Otro número -> Esta es la respuesta correcta, puesto que carecer de sentencia break.
- c) Uno
- d) Otro número

3. Selecciona la respuesta correcta con respecto al resultado del bloque de código.

```
public class Test1 extends Concreate{
    1 usage
    Test1(){
        System.out.println("t ");
    }

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        new Test1();
    }
}
1 usage 1 inheritor
class Concreate extends Send{
    1 usage
    Concreate(){
        System.out.println("c ");
    }
    private Concreate(String s){

    }
}
1 usage 2 inheritors
abstract class Send{
    2 usages
    Send(){
        System.out.println("s ");
    }
}
```

- a) c,s,t
- b) t,s,c
- c) Error en tiempo de ejecución
- d) No compila

4. ¿Qué es REST y cuál es su relación con las API web?

- a) REST es un protocolo de comunicación. Su relación con las API web es que las utiliza para definir los endpoints de una API
- b) REST es in lenguaje de programación. Su relación con las API web es que se utiliza para crear aplicaciones web.
- c) REST es un servicio en la nube. Su relación con las API web que se utiliza para alojar las aplicaciones web.
- d) REST es una arquitectura para aplicaciones web. Su relación con las API web es que se utiliza para definir la estructura y funcionalidades de una API.

5. De los siguientes paquetes, ¿cuáles contienen clases para construir una interfaz gráfica? (Elije todas las que correspondan)

- a) java.net
- b) java.io
- c) javax.swing
- d) java.util
- e) java.awt

6. ¿Cuál de las siguientes líneas deben ir en el espacio en blanco para que el código compile?

```
public class News < _____> { }
```

- a) Solo N
- b) Solo ?
- c) Ninguna de las anteriores
- d) News,y Object
- e) ? y N
- f) N, News y Object

7. ¿Qué es el patrón de diseño DAO y cómo se implementa en Java?

- a) El patrón de diseño DAO es un patrón que se utiliza para abstraer la capa de acceso a datos en una aplicación. Se puede implementar en Java utilizando interfaces y clases concretas.
- b) El patrón de diseño DAO es un patrón que se utiliza para abstraer la capa de negocios de una aplicación. Se puede implementar en Java utilizando clases abstractas y métodos estáticos.
- c) El patrón de diseño DAO es un patrón que se utiliza para abstraer la capa de presentación en una aplicación. Se puede implementar en Java utilizando interfaces y clases concretas.
- d) El patrón de diseño DAO es un patrón que se utiliza para abstraer la capa de infraestructura en una aplicación. Se puede implementar en Java utilizando excepciones y bloques try-catch.

8. ¿Cuál de las siguientes líneas deben ir en el espacio en blanco para que el código compile?

```
public class News < _____> { }
```

- g) Solo N
- h) Solo ?
- i) Ninguna de las anteriores
- j) News,y Object
- k) ? y N
- l) N, News y Object -----

9. ¿Qué es un stream en Java 8 y para qué se utiliza?
- a) Un objeto que representa una conexión de entrada o salida de datos.
 - b) Un objeto que representa una secuencia de elementos y se utiliza para procesar colecciones de forma declarativa.
 - c) Un objeto que se utiliza para leer y escribir archivos de texto.
 - d) Un objeto que se utiliza para crear y manipular bases de datos.
10. Son patrones de diseño de microservicios
- a) Circuit Breaker, Adaptative Lifo, MQ Strategy
 - b) System, Process y Client
 - c) Retry, Circuit Breaker, Adaptative Lifo y Bulkhead
 - d) Ninguna de las anteriores
11. ¿Cuándo se debe usar un bloque finally en una declaración try regular (no una prueba con recursos)?
- a) Nunca.
 - b) Cuando el código del programa no termina por sí solo.
 - c) Cuando hay dos o más bloques catch en una sentencia try.
 - d) Cuando hay exactamente un bloque catch en una sentencia try
 - e) Cuando no hay bloques catch en una declaración try
12. De los siguientes ¿qué tipos de declaraciones se deben usar para contar la cantidad de monedas de 5 centavos en una matriz de cadenas de varias monedas?
- a) Assertion
 - b) Iteration
 - c) Assignment
 - d) Conditional
13. ¿Qué afirmaciones son verdaderas tanto para las clases abstractas como para las interfaces? (Elije todas las correctas)
- a) Ambos pueden contener métodos estáticos.
 - b) Ambos se pueden ampliar con la clave extend.
 - c) Ambos pueden contener métodos predeterminados.
 - d) Ambos heredan de java.lang.Object.
 - e) Ninguno de los dos puede ser instanciado directamente.
 - f) Ambos pueden contener variables finales estáticas públicas.
 - g) Supone que todos los métodos dentro de ellos son abstractos.

Las clases abstractas y los métodos pueden contener métodos no definidos, sin embargo, una de las diferencias es que las clases abstractas pueden contener la definición de un método siempre y cuando este no sea abstracto.

14. ¿Cuál no es un objetivo de Maven?

- a) Clean
- b) Package
- c) Debug
- d) Install

15. ¿Si deseas obtener una copia de un repositorio Git existente en un servidor qué comando se utiliza?

El comando utilizado será git clone, de esta manera se procede a clonar dicho repositorio.

- a) git commit
- b) git log
- c) git clone
- d) git add

16. ¿Cuál es el comando utilizado para fusionar una rama en Git?

- a) git push
- b) git branch
- c) git pull
- d) git merge

17. ¿Qué es un repositorio remoto en Git?

Un repositorio remoto es un repositorio que permanece en la nube, de esta manera, cualquier persona involucrada en el proyecto puede acceder al código, modificarlo, subir los cambios y compartir sus cambios.

- a) Una herramienta que se utiliza para compartir y fusionar cambios entre diferentes ramas de un repositorio.
- b) Una copia local de un repositorio que se utiliza para hacer cambios en el código fuente.
- c) Un servidor Git que almacena una copia central del repositorio.
- d) Un archivo que contiene una instantánea del código fuente en un momento determinado.

18. ¿Cuál es el comando utilizado para actualizar la rama local con los cambios de la rama remota en Git?

- a) git pull
- b) git push
- c) git clone
- d) git checkout

19. Dada la siguiente clase

```
public class Helper {  
    public static < U extends Exception > void  
        printException(U u){  
        System.out.println(u.getMessage());  
    }  
  
    public static void main(String[] args) {  
        //línea 9  
    }  
}
```

¿Cuál de las siguientes instrucciones puede colocarse en la línea 9 para que la clase Helper compile?

- a) `Helper.printException(new Exception("B"));`
- b) `Helper.printException(new FileNotFoundException("A"));`
- c) `Helper.<Throwable>printException(new Exception("C"));`
- d) `Helper.<NullPointerException>printException(new NullPointerException("D"));`
- e) `Helper.printException(new Throwable("E"));`

20. ¿Cuál es la salida al ejecutar el siguiente código?

```
public class Fish {  
    public static void main(String[] args) {  
        int numFish = 4;  
        String fishType = "tuna";  
        String anotherFish = numFish + 1;  
        System.out.println(anotherFish + " " + fishType);  
        System.out.println(numFish + " " + 1);  
    }  
}
```

- a) 51tuna
- b) 5tuna
- c) 5
- d) 41
- e) 5 tuna
- f) 4 1
- g) El código no compila

21. ¿Qué es Git?

- a) Una herramienta de automatización de compilación que se utiliza para compilar un proyecto.
- b) Una herramienta de generación de informes que se utiliza para generar informes sobre el rendimiento de una aplicación.
- c) Una herramienta de gestión de dependencias que se utiliza para descargar bibliotecas y paquetes en un proyecto de Java.
- d) Una herramienta de control de versiones que se utiliza para almacenar y administrar el código fuente de un proyecto.

Git es una herramienta de versionamiento de código, esta ayuda a generar un historial de cambios a lo largo del ciclo de vida de un software.

22. ¿Cuál de las siguientes opciones son correctas? (Elija todas las correctas)

```
public class StringBuilders {  
    1 usage  
    public static StringBuilder work(StringBuilder a, StringBuilder b){  
        a = new StringBuilder("a");  
        b.append("b");  
        return a;  
    }  
  
    public static void main(String[] args) {  
        StringBuilder s1 = new StringBuilder("s1");  
        StringBuilder s2 = new StringBuilder("s2");  
        StringBuilder s3 = work(s1,s2);  
        System.out.println("s1 = " + s1);  
        System.out.println("s2 = " + s2);  
        System.out.println("s3 = " + s3);  
    }  
}
```

- a) s2 = s2
- b) s3 = null
- c) s1 = s1-----
- d) s3 = a
- e) El código no compila
- f) s2 = s2b
- g) s1 = a

23. ¿A qué hace referencia el principio de Liskov?

- a) Nos indica que una clase no debe tener solo una funcionalidad sino varias para reducir el uso de objetos.
- b) Este principio nos indica que dentro del programa una clase puede ser sustituida por cualquier clase que se extienda de ella sin alterar el comportamiento del programa.
- c) Nos indica que cualquier clase se puede extender para agregar funcionalidad, pero no se puede modificar.
- d) Este principio nos indica que dentro del programa una clase puede ser sustituida por su clase padre sin alterar el comportamiento del programa.

24. ¿Qué es un “code smell”?

- a) Un componente de la biblioteca estándar de Java
- b) Un error en tiempo de compilación que se produce en Java
- c) Un indicador de que puede haber un problema en el código que puede ser difícil de detectar o que podría ser una fuente potencial de errores o problemas de mantenimiento en el futuro.
- d) Una práctica de programación recomendada en Java.

25. ¿Cuál de las siguientes afirmaciones son verdaderas? (Elije todas las correctas)

- a) Las excepciones de tiempo de ejecución son lo mismo que las excepciones comprobadas.
- b) Las excepciones en tiempo de ejecuciones son lo mismo que las excepciones no comprobadas.
- c) Solo pueden manejar subclases de exception.
- d) Solo puede declarar excepciones comprobadas (checked).
- e) Puede declarar solo excepciones no comprobadas.

26. ¿Cuáles son las salidas del siguiente código? (Elije todas las correctas)

```
public class StrinBuilders {  
    1 usage  
    public static StringBuilder work(StringBuilder a, StringBuilder b){  
        a = new StringBuilder("a");  
        b.append("b");  
        return a;  
    }  
  
    public static void main(String[] args) {  
        StringBuilder s1 = new StringBuilder("s1");  
        StringBuilder s2 = new StringBuilder("s2");  
        StringBuilder s3 = work(s1,s2);  
        System.out.println("s1 = " + s1);  
        System.out.println("s2 = " + s2);  
        System.out.println("s3 = " + s3);  
    }  
}
```

a) El código no compila.

b) s3 = a

c) s1 = a

d) s2 = s2

e) s1 = s1

f) s3 = null

g) s2 = s2b

27. Selecciona la respuesta correcta con respecto al resultado del bloque de código.

```
public class Test1 extends Concrete{  
    1 usage  
    Test1(){  
        System.out.println("t ");  
    }  
  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        new Test1();  
    }  
}  
  
1 usage 1 inheritor  
class Concrete extends Send{  
    1 usage  
    Concrete(){  
        System.out.println("c ");  
    }  
    private Concrete(String s){  
    }  
}  
  
1 usage 2 inheritors  
abstract class Send{  
    2 usages  
    Send(){  
        System.out.println("s ");  
    }  
}
```

a) Error en tiempo de ejecución.

b) No compila.

c) t,s,c

d) c,s,t

e) s,c,t

28. ¿Cuál es la función del operador de doble dos puntos (::) en Java 8?

- a) El operador doble dos puntos se utiliza para crear una nueva instancia de una clase en Java 8.
- b) EL operador de doble dos puntos no se utiliza en java 8.
- c) El operador de doble dos puntos se utiliza para acceder a métodos estáticos en Java 8.
- d) El operador de doble dos puntos se utiliza para acceder a métodos no estáticos en Java 8.

29. ¿Qué significa el acrónimo CRUD en una API REST?

- a) Code, Register, Update, Debug
- b) Create, Read, Update, Delete
- c) Call, Receive, Use, Debug
- d) Customize, Request, Use, Debug

CRUD significa Create, Read, Update, Delete, por lo general son las acciones básicas dentro de una BD.
--

30. ¿Qué es un bean en Spring?

- a) Un archivo de configuración XML que se utiliza para definir la estructura de una tabla de base de datos.
- b) Una instancia de una clase que se administra por el contenedor de Spring.
- c) Una herramienta de inyección de dependencias que se utiliza para inyectar dependencias en una clase.
- d) Una clase que se utiliza para configurar la conexión a una base de datos.

31. ¿Para qué nos sirve utilizar un profile dentro del archivo pom.xml?

- a) Etiqueta por la cual podemos definir la versiones de nuestras dependencias.
- b) Es la etiqueta por la cual podemos definir las características que tendrá nuestro proyecto al ser compiladas.
- c) Etiqueta por la cual definimos los parámetros de conexión a un repositorio.
- d) No existe esta etiqueta en Maven.

32. ¿Cuál es el comando utilizado para actualizar la rama local con los cambios de la rama remota en Git?

- a) git checkout
- b) git clone
- c) git push
- d) git pull

33. Dadas las siguientes clases Vehicle y Car

```
package my.vehicles;

public class Vehicle {
    public String make;
    protected String model;
    private int year;
    int mileage;
}
```

```
package my.vehicles.cars;

import my.vehicles.*;

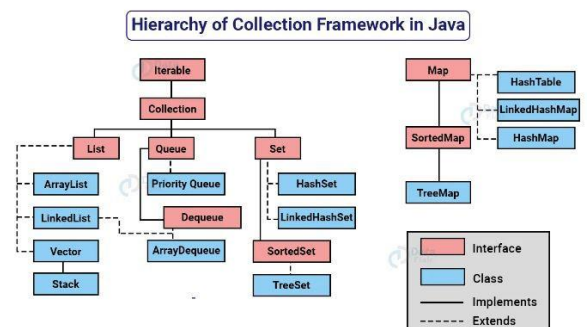
public class Car extends Vehicle {
    public Car() {
        //línea 7
    }
}
```

¿Cuál de las siguientes instrucciones pueden colocarse en la línea 7 para que la clase Car compile correctamente? (Seleccione las que apliquen)

- a) `mileage = 15285;`
- b) Ninguna de las anteriores.
- c) `make = "Honda";`
- d) `year = 2009;`
- e) `model = "Pilot";`

34. Enumere cuatro interfaces de la API colecciones

- a) `List`, `Map`, `Set`, `Queue`.
- b) `ArrayList`, `Map`, `Set`, `Queue`.
- c) `List`, `HashMap`, `HashSet`, `PriorityQueue`.
- d) `List`, `Map`, `HashSet`, `PriorityQueue`.



35. Selecciona la respuesta correcta con respecto al resultado del bloque de código.

```
public class Test5 {  
    public static void main(String args[]) {  
        Side primerIntento = new Head();  
        Tail segundoIntento = new Tail();  
        Coin.overload(primerIntento);  
        Coin.overload((Object)segundoIntento);  
        Coin.overload(segundoIntento);  
        Coin.overload((Side)primerIntento);  
    }  
}  
  
interface Side { String getSide();}  
  
class Head implements Side {  
    public String getSide() { return "Head ";}  
}  
  
class Tail implements Side {  
    public String getSide() { return "Tail ";}  
}  
  
class Coin {  
    public static void overload(Head side) {System.out.println(side.getSide());}  
    public static void overload(Tail side) {System.out.println(side.getSide());}  
    public static void overload(Side side) {System.out.println("Side ");}  
    public static void overload(Object side) {System.out.println("Object ");}  
}
```

- a) Head
Object
Tail
Side
- b) No compila
- c) Side
Object
Tail
Side
- d) Head
Head
Tail
Tail
- e) Side
Head
Tail
Side

36. ¿Cuál es la salida al ejecutar el siguiente código?

```
public class Lion {  
    public void roar(String roar1, StringBuilder roar2) {  
        roar1.concat("!!!");  
        roar2.append("!!!");  
    }  
    public static void main(String[] args) {  
        String roar1 = "roar";  
        StringBuilder roar2 = new StringBuilder("roar");  
        new Lion().roar(roar1, roar2);  
        System.out.println(roar1 + " " + roar2);  
    }  
}
```

- a) roar roar!!!
- b) roar!!! Roar
- c) Se lanza una excepción
- d) roar!!! roar!!!
- e) roar roar
- f) El código no compila

37. ¿Cuál de los siguientes es cierto acerca de una subclase completa?

- a) Una subclase concreta no se puede marcar como final.
- b) Una subclase concreta debe implementar todos los métodos definidos en una interfaz heredada.
- c) Una subclase concreta debe implementar todos los métodos abstractos heredados.
- d) Una subclase concreta puede declararse como abstracta.
- e) Los métodos abstractos no pueden ser anulados por una subclase concreta.

38. ¿Cuál es la salida del siguiente código?

```
1  public abstract class Catchable {
2      protected abstract void catchAnObject(Object x);
3
4      public static void main(String [] args) {
5          java.util.Date now = new java.util.Date();
6          Catchable target = new MyStringCatcher();
7          target.catchAnObject(now);
8      }
9  }
10
11  class MyStringCatcher extends Catchable {
12      public void catchAnObject(Object x) {
13          System.out.println("Caught object");
14      }
15
16      public void catchAnObject(String s) {
17          System.out.println("Caught string");
18      }
19  }
```

- a) Error de compilación línea 12
- b) Error compilación línea 16
- c) Caught string
- d) Error compilación línea 2
- e) Caught Object

39. Seleccione la respuesta que considere correcta, dado el siguiente bloque de código.

```
1  import java.util.Arrays;
2  import java.util.List;
3
4  public class Example {
5
6      public static void main(String[] args) {
7          List<Integer> numbers = Arrays.asList(1, 2, 3, 4, 5);
8
9          int result = numbers.stream()
10              .filter(n -> n % 2 == 0)
11              .reduce(0, (a, b) -> a + b);
12
13          System.out.println(result);
14
15      }
16  }
17
```

```

import java.util.Arrays;
import java.util.List;
public class Example {
    public static void main(String[] args) {
        List<Integer> numbers Arrays.asList(1, 2, 3, 4, 5);
        int result = numbers.stream()
            .filter (n ->20)
            .reduce(0, (a, b) -> a + b);
        System.out.println(result);
    }
}
)

```

- a) 3
- b) 9
- c) 14
- d) 6

40. Qué declaración representa una declaración válida que permitirá la inclusión de clases del paquete java.util?

- a) #include java.util.*;
- b) #include java.util;
- c) import java.util.*;
- d) import java.util;

41. ¿Qué es la cobertura de código?

- a) La cantidad de veces que se ejecuta una línea de código.
- b) La cantidad de errores detectados por una prueba.
- c) La cantidad de código que se ejecuta durante una prueba.
- d) La cantidad de tiempo que tarda una prueba en ejecutarse.

42. ¿Qué es Git?

- e) Una herramienta de automatización de compilación que se utiliza para compilar un proyecto.
- f) Una herramienta de generación de informes que se utiliza para generar informes sobre el rendimiento de una aplicación.
- g) Una herramienta de gestión de dependencias que se utiliza para descargar bibliotecas y paquetes en un proyecto de Java.
- h) Una herramienta de control de versiones que se utiliza para almacenar y administrar el código fuente de un proyecto.

43. ¿Cuál es el formato correcto para hacer un commit en Git?

- a) Descripción breve del cambio y nombre del autor.
- b) Tipo de cambio, descripción breve, cuerpo opcional y notas de pie de página.
- c) Solo se necesita una breve descripción del cambio.
- d) Nombre de la rama, descripción detallada del cambio y fecha.

44. ¿Qué es el patrón de diseño Singleton y cómo se implementa en Java 8?

- a) El patrón de diseño Singleton es un patrón que se utiliza para garantizar que una clase tenga una única instancia en todo el sistema. Se implementa utilizando una variable estática y un constructor privado.
- b) El patrón de diseño Singleton es un patrón que se utiliza para abstraer la capa de infraestructura en una aplicación. Se implementa utilizando excepciones y bloques try-catch.
- c) El patrón de diseño Singleton es un patrón que se utiliza para abstraer la capa de presentación en una aplicación. Se implementa con interfaces y clases concretas.
- d) El patrón de diseño Singleton es un patrón que se utiliza para abstraer la capa de negocios en una aplicación. Se implementa utilizando clases abstractas y métodos estáticos.

45. ¿Qué es un Microservicio?

- a) Ninguna de las anteriores
- b) Es un componentes que se pueden desplegar de forma independiente en función múltiple, es decir, englobando endpoint que no necesariamente están relacionados.
- c) Es el conjunto de endpoints contenidos en múltiples desarrollos que se despliegan en conjunto y que están estrechamente relacionados.
- d) Es un componentes que se pueden desplegar de forma independiente y que suelen ser de función única, es decir, englobando endpoint y están estrechamente relacionandos.

46. ¿Cuál no es un objetivo de Maven?

- a) Debug
- b) Clean
- c) Package
- d) Install

47. En los verbos REST ¿Cuál es la diferencia en el uso de PATCH y PUT?

- a) Son exactamente iguales, no hay diferencia de uso.
- b) PATCH requiere se le envíe la entidad completa mientras que PUT solo los atributos a modificar.
- c) PUT requiere se le envíe la entidad completa mientras que PATCH solo los atributos a modificar.
- d) PATCH es un verbo deprecado sustituido por PUT.

48. ¿Qué es una expresión lambda en Java 8?

- a) Una expresión lambda es una forma de escribir una clase anónima en Java 8.
- b) Una expresión lambda es una forma de escribir una función anónima en Java 8.
- c) Una expresión lambda es un método que se llama automáticamente cuando se crea un objeto.
- d) Una expresión lambda es un método que se llama de forma explícita desde el código.

49. ¿Cuál es la diferencia entre una clase abstracta y una interfaz en Java 8?

- a. Una interfaz solo puede heredar de una clase, mientras una clase abstracta puede heredar de múltiples interfaces
- b. Una clase abstracta puede contener implementaciones de métodos, mientras que una interfaz no puede.
- c. Una clase abstracta puede contener variables de instancia, mientras que una interfaz no puede.
- d. Una interfaz puede contener implementaciones de métodos, mientras que una clase abstracta no puede.

50. ¿Cuál es la diferencia entre las anotaciones: @RestController, @Component, @Service y @Repository?

- a) @Controller es una anotación que nos ayuda a construir una API REST mientras que @Service, @Component y @Repository solo marcan las clases que se deben de inicializar.
- b) @Controles, @Component son anotaciones que crean bean y exponen la serialización de las clases mientras que @Service y @Repository requieren de una inicialización manual.
- c) No existe diferencia funcional entre ellas sino semántica, las 4 son anotaciones de Spring que crean un bean y lo agregan al contexto de Spring.
- d) @Service y @Repository son anotaciones que crear bean y exponen la serialización de las clases mientras que @Controller. @Component requiere de una inicialización manual.

51. ¿Cuál es una buena práctica al escribir pruebas unitarias?

- a) Ejecutar pruebas con poca frecuencia.
- b) Asegurarse de que las pruebas sean claras y concisas.
- c) Probar solo una pequeña parte de una función.
- d) Hacer que las pruebas dependan de otras pruebas.

52. ¿Cuál es la ventaja de usar APIs REST sobre otros tipos de servicios web?

- a) Mayor seguridad.
- b) Mayor facilidad de implementación.1
- c) Mayor velocidad de transferencia de datos.
- d) Mayor compatibilidad con diferentes plataformas.

53. Selecciona la respuesta correcta con respecto al resultado del bloque de código.

```
public class Test4 {  
  
    public static void main(String[] args) {  
        List list = Arrays.asList(25,7,25,67);  
        System.out.println(list);  
        System.out.println(new HashSet(list));  
        System.out.println(new TreeSet(list));  
        System.out.println(new HashSet(list));  
        System.out.println(new ConcurrentSkipListSet(list));  
    }  
}
```

```
public class Test4 (  
  
    public static void main(String[] args) {  
        List list = Arrays.asList(25,7,25,67);  
        System.out.println(list);
```

```

        System.out.println(new HashSet(list));
        System.out.println(new TreeSet(list));
        System.out.println(new HashSet(list));
        System.out.println(new ConcurrentSkipListSet(list));
    }
}

```

- a) No compila
- b) [25, 7, 25, 67]
 [67, 7, 25]
 [7, 25, 67]
 [67, 7, 25]
 [7, 25, 67]
- c) [25, 7, 67]
 [67, 7, 25]
 [7, 25, 67]
 [67, 7, 25]
 [7, 25, 67]
- d) [67, 7, 25]
 [67, 7, 25]
 [67, 7, 25]
 [67, 7, 25]
 [67, 7, 25]
- e) [25, 7, 25, 67]
 [7, 25, 67]
 [67, 7, 25]
 [7, 25, 67]
 [67, 7, 25]

54. ¿Cuál es la salida al ejecutar el siguiente código?

```

public class Fish {
    public static void main(String[] args) {
        int numFish = 4;
        String fishType = "tuna";
        String anotherFish = numFish + 1;
        System.out.println(anotherFish + " " + fishType);
        System.out.println(numFish + " " + 1);
    }
}

```

```

public class Fish {
    public static void main(String[] args) {
        int numFish = 4;
        String fishType = "tuna";
        String anotherFish = numFish + 1;
        System.out.println(anotherFish + + + fishType);
        System.out.println(numFish + " " + 1);
    }
}

```

- a) 51tuna
- b) 5tuna
- c) 5
- d) 41
- e) 5 tuna
- f) 4 1
- g) El código no compila

55. ¿Cuáles son los 4 pilares de la programación orientada a objetos?

- a) Polimorfismo, Coerción, Herencia y Encapsulamiento.
- b) Encapsulamiento, Coerción, Polimorfismo y Abstracción.
- c) Polimorfismo, Herencia, Encapsulamiento y Sincronía.
- d) Polimorfismo, Abstracción, Herencia y Encapsulamiento.

56. ¿Cuál es el comando utilizado para ver el historial de cambios en git?

- a) git diff
- b) git status
- c) git log
- d) git commit

57. ¿Qué utilidad de línea de comandos basada en MS Windows le permitirá ejecutar el intérprete de Java sin abrir la ventana de la consola?

- a) jconsole
- b) javaw
- c) interpw
- d) java -wo

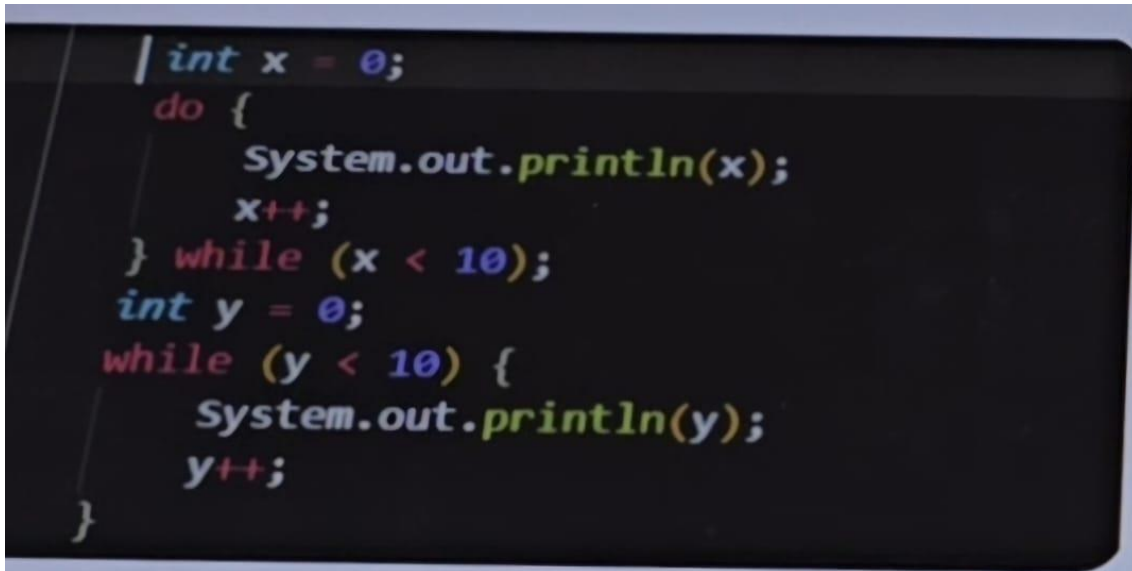
58. ¿Qué es un endpoint en una API REST?

- a) Un endpoint es un objeto que se utiliza para almacenar datos en una API REST.
- b) Un endpoint es un método que se utiliza para procesar datos en una API REST.
- c) Un endpoint es un controlador que se utiliza para administrar una API REST.
- d) Un endpoint es la URL que se utiliza para acceder a una API REST.

59. ¿Cuál de lo siguiente es cierto?

- a) java toma el nombre del archivo .bytecode como parámetro.
- b) javac compila un archivo .java en un archivo .class
- c) Java toma el nombre de la clase como parámetro.
- d) javac compila un archivo .class en un archivo .java
- e) javac compila un archivo .java en un archivo .bytecode
- f) Java toma el nombre del archivo .class como parámetro.

60. ¿Cuál es el valor de x e y al final el programa?



```
| int x = 0;
do {
    System.out.println(x);
    x++;
} while (x < 10);
int y = 0;
while (y < 10) {
    System.out.println(y);
    y++;
}
```

- a) X=9 y=10
- b) X=10 y=9
- c) X=10 y=10
- d) X=9 y=9

61. ¿Dado el siguiente enum y clase cuál es la opción que puede ir en el espacio en blanco para que el código compile?

```
enum Season { SPRING, SUMMER, WINTER }
public class Weather {
    public int getAverageTemperature(Season s) {
        switch (s) {
            default:
                _____ return 30;
        }
    }
}
```

- a) Ninguno de los anteriores
- b) case SUMMER ->
- c) case Season.Winter:
- d) case FALL:
- e) case Winter, Spring:
- f) case SUMMER | WINTER:

62. ¿Cuál es una buena práctica al escribir pruebas unitarias?

- a) Hacer que las pruebas dependan de otras pruebas
- b) probar solo una pequeña parte de una función
- c) ejecutar pruebas con poca frecuencia
- d) asegurarse de que las pruebas sean claras y concisas

63. ¿Cuál es el resultado de compilar y ejecutar el siguiente programa?

```
public static void main(String[] args) {
    boolean stmt1 = "champ" == "champ";
    boolean stmt2 = new String( original: "champ") == "champ";
    boolean stmt3 = new String( original: "champ") == new String( original: "champ");
    System.out.println(stmt1 && stmt2 || stmt3);
}
```

- a) False
- b) no se produce salida
- c) true
- d) error de compilación

64. ¿Cómo se manejan las excepciones en Java?

- a) Las excepciones se manejan con bloques switch case en Java. La excepción trae with Resources es una forma de lanzar una excepción en un método.
- b) Las excepciones se manejan con bloques while en Java. La excepción trae with Resources es una forma de manejar excepciones de compilación
- c) las excepciones se manejan con bloques if else en Java. La excepción trae with resource es una forma de manejar las excepciones en tiempo de ejecución.
- d) Las excepciones se manejan con bloques try catch finally en Java. La excepción trae with Resources es una forma de cerrar automáticamente los recursos abiertos en un bloque try.

65. ¿Qué clase del paquete java.io permite leer y escribir archivos en ubicaciones específicas dentro de un archivo?

- a) File
- b) filename filter
- c) file descriptor
- d) RandomAccessFile

66. Todas las siguientes definiciones de clases my School classroom y my City School ¿qué números de línea en el método main generan un error de compilación? (Elija todas las opciones correctas)

```
1: package my.school;
2: public class Classroom {
3:     private int roomNumber;
4:     protected String teacherName;
5:     static int globalKey = 54321;
6:     public int floor = 3;
7:     Classroom(int r, String t) {
8:         roomNumber = r;
9:         teacherName = t; } }

1: package my.city;
2: import my.school.*;
3: public class School {
4:     public static void main(String[] args) {
5:         System.out.println(Classroom.globalKey);
6:         Classroom room = new Classroom(101, "Mrs. Anderson");
7:         System.out.println(room.roomNumber);
8:         System.out.println(room.floor);
9:         System.out.println(room.teacherName); } }
```

- a) Ninguna, el código compila bien
- b) línea 6

- c) línea 9
- d) línea 7
- e) línea 8
- f) línea 5

67. ¿Qué es una expresión lambda en Java?

- a) Una instancia de una clase que implementa una interfaz funcional
- b) Una instancia de una clase abstracta que se utiliza para implementar métodos anónimos
- c) Una forma concisa de representar una función anónima que se puede pasar como argumento
- d) Un método que no tiene cuerpo

68. ¿Qué hace el siguiente código fuente?

```
int x = 0;
boolean flag = false;
while ((x < 10) || !flag) {
    System.out.println(x);
    x++;
}
```

- a) Muestra los números del 1 al 10
- b) muestra un 10
- c) se queda en un bucle infinito
- d) muestra los números del 0 al 9

1. Comando Docker que se utiliza para construir la imagen a partir del Dockerfile

`docker build`