

Temario ASO

1. Java

- Conceptos básicos de POO.
- Clases y objetos.
- Atributos y métodos.
- Herencia.
- Polimorfismo.
- Interfaces y Clases abstractas.
- Encapsulamiento.
- Constructores
- Tipos de datos primitivos y variables.
- Operadores.
- Estructuras de datos (arreglos, matrices, listas).
- Estructuras de control (condicionales y ciclos).
- Construcción y empaquetado de aplicaciones Java.
- Modificadores de acceso.
- Interfaces gráficas (Swing y Awt). Q53
- ENUM. Q88
- Entrada/Salida (I/O).
- Expresiones regulares.
- Threads.
- Var args Q98.

2. Colecciones

- ArrayList, LinkedList
- HashSet, TreeSet.
- HashMap, TreeMap.

3. Manejo de Excepciones

- Bloques try-catch.
- Tipos de excepciones.
- Multicatch & TryWithResources.

4. Lambdas y Streams en java 8

- Expresiones lambda.
- Interfaces funcionales.
- Streams y operaciones sobre colecciones.
- Uso de generics.
- Method Reference

5. Conceptos de Programación

- Principios SOLID (Liskov Substitution, Open/Closed, Interface Segregation, Dependency Inversion Principle, Single Responsibility).
- Code Smeller.

- CRUD

6. Git y Github

- Comandos básicos (commit, push, pull, branch, merge, log, clone, revert).
- Repositorios remotos y locales.
- Flujo de trabajo con ramas.

7. Maven

- Objetivos de Maven.
- Perfiles Maven.
- Comandos básicos Maven (mvn clean, mvn compile, mvn package, mvn install)

8. APIs REST

- Conceptos básicos REST.
- Verbos HTTP (GET, POST, PUT, PATCH, DELETE).
- Endpoints y recursos.
- Formato de datos.

9. Pruebas de software

- Pruebas unitarias.
- Pruebas de integración.
- Cobertura de código.

10. Patrones de diseño

- Patrones de diseño estructurales (Adapter, Proxy, Bridge, Composite).
- Patrón Singleton.
- Patrón DAO.

11. Microservicios

- Conceptos de microservicios
- Patrones de diseño de microservicios.

12. Spring Framework

- Concepto de anotaciones en Spring.
- @RestController, @Component, @Service, @Repository.
- Beans y contenedor de Spring.
- Starters en Spring Boot

Temario Accenture

1. Java (ASO/APX)

1. Programación Orientada a Objetos(POO)
2. Control de flujo y bucles en Java
3. Manejo de Excepciones
4. Clases Abstractas e Interfaces
5. Herencia y polimorfismo (sobrecarga y sobrescritura)
6. Estructuras de datos (colecciones)
7. Patrones de Diseño
8. Pruebas Unitarias
9. Mockito y JUnit

2. Spring

1. Anotaciones
2. Spring boot (ASO/APX)
 1. Transacciones y microservicios
3. Spring batch (APX)
 1. Jobs
 2. Steps

3. Desarrollo web (ASO/APX)

1. Protocolo HTTP/HTTPS
2. Verbos (POST, PUT, GET, DELETE, PATCH)
3. Códigos de respuesta
4. Servicios web
5. REST/SOAP diferencias y ventajas
6. API

- 4. **Patrones de Diseño (ASO)**
 - 1. ¿Qué son los patrones de diseño?
 - 2. Patrón Singleton
 - 3. Patrón Facade y Abstract Facade
 - 4. Patrón MVC
 - 5. Inyección de dependencias
- 5. **Maven (ASO/APX)**
- 6. **Git (ASO/APX)**
- 7. **Bases de datos (APX)**
 - 1. **BBDD relacionales**
 - 2. **BBDD NO relacionales**
- 8. **Docker (crear, levantar, detener y borrar instancias) (ASO/APX)**

AVANCE: 70%

Preguntas

1. ¿Cuál es el comando utilizado para deshacer el último commit en git?
 - a) git reset
 - b) git revert ← Es correcto
 - c) git amend
 - d) git checkout

2. ¿Cuál es la diferencia entre una clase abstracta y una interfaz en java 8?
 - a) Una clase abstracta puede contener implementaciones de métodos, mientras que una interfaz no puede. ← Esta es la que, aunque es incorrecta sale en varios lugares como correcta.
 - b) Una clase abstracta puede contener variables de instancia, mientras que una interfaz no puede ← Si no sale la de abajo, esta sería la que se acerca más.
 - c) Una interfaz puede contener implementaciones de métodos, mientras que una clase abstracta no puede.
 - d) Una interfaz solo puede heredar de una clase, mientras que una clase abstracta puede heredar... (no se ve)
 - e) Una interfaz puede ser implementada por múltiples clases mientras que una clase abstracta solo puede ser subclassificada. ← Si esta opción sale, es correcta

3. 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? (Elije todas las correctas)

- a) Conditional (Duda Preguntar)
- b) Assertion
- c) Assignment ← Correcta
- d) Iteration ← Correcta

4. ¿Qué es un archivo JAR en java?

- a) Un archivo que contiene un archivo de configuración Maven
- b) Un archivo que contiene un archivo de configuración Git.
- c) Un archivo que contiene una clase Java compilada ← Correcta
- d) Un archivo que contiene un archivo de configuración de Spring

5. ¿Qué es la sobrecarga de métodos en Java?

- a) Cuando un método tiene múltiples definiciones con el mismo nombre y tipo de parámetros.
- b) Cuando un método tiene múltiples definiciones con diferentes nombres y cantidades de parámetros.
- c) Cuando un método tiene múltiples definiciones con diferentes tipos de cantidades de parámetros. ← Correcta
- d) Cuando un método tiene múltiples definiciones con diferentes nombres y tipos de parámetros.

6. ¿Cuál es la diferencia entre un ArrayList y un LinkedList en Java?

- a) ArrayList es más rápido que un LinkedList para agregar y eliminar elementos.
- b) ArrayList es más eficiente en el uso de memoria que LinkedList.
- c) LinkedList es una clase abstracta mientras que ArrayList es una clase concreta.
- d) LinkedList es más rápido que ArrayList para acceder a elementos aleatorios.

7. ¿Cuándo se debe usar un bloque finally en una declaración try regular (no una prueba con recursos)?

- a) Cuando no hay bloques catch en una declaración try.
- b) Nunca.
- 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 el código del programa no termina por sí solo.

8. ¿Cuál es el propósito principal de los test unitarios?

- a) Comprobar la eficiencia del hardware.
- b) Medir la velocidad de la aplicación.
- c) Ahorrar tiempo en el desarrollo.
- d) Asegurar la calidad del software.

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

```
public class Test3 {  
    public static void main(String[] args) {  
        String cad1 = "hola";  
        String cad2 = new String( original: "hola");  
        String cad3 = "hola";  
  
        if (cad1 == cad2)  
            System.out.println("ca1 es igual a cad2");  
        else System.out.println("cad1 diferente a cad2");  
  
        if (cad1 == cad3)  
            System.out.println("cad1 es igual a cad3");  
        else  
            System.out.println("cad1 diferente a cad3");  
    }  
}
```

- a) cad1 diferente a cad2
cad1 es igual a cad3
- b) ca1 es igual a cad2
ca1 es igual a cad3

c) No compila

d) cad1 diferente a cad2

cad1 diferente a cad3

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

```
class Mammal{
    public Mammal(int age){
        System.out.println("Mammal");
    }
}
public class Platypus extends Mammal{
    public Platypus(){
        System.out.println("Platypus");
    }

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

a) Mammal.

b) MammalPlatypus.

c) El código no se compila en la línea 11.

d) El código no compila en la línea 8

11. ¿Cómo se manejan las excepciones en java?

a) Con la instrucción try-catch.

b) Con la instrucción if-else.

c) Con la instrucción for.

d) Con la instrucción while.

12. ¿La anotación @Ignore es usada para omitir un test por lo que no se ejecuta?

a) Verdadero

b) Falso

13. ¿Cuál es el resultado de compilar y ejecutar el siguiente código?


```

public class Tester {
    static {
        int x = 3;
    }
    2 usages
    static int x;
    public static void main(String[] args) {
        x--; // line 7
        System.out.println(x);
    }
}

```

- a) Error de compilación en la línea 7, x no se inicializa.
- b) -1
- c) -2
- d) 0

14. ¿Qué es un operador de short circuit?

- a) Sirve para realizar más eficientes las operaciones condicionales evitando ejecutar operaciones si estas ya no son necesarias.
- b) Operador que nos sirve para crear una nueva clase anónima.
- c) Sirve para lanzar una excepción personalizada.
- d) Es un patrón de arquitectura de microservicios que nos permite evitar el consumo de servicios que están en mantenimiento.

15. ¿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.

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

- a) Un endpoint es la URL que se utiliza para acceder a 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 un objeto que se utiliza para almacenar datos en una API REST.

17. ¿Qué hace el siguiente programa?

```
public class Palabra {
    public static void main(String[] args) {
        String sPalabra = "palabra";
        int inc = 0;
        int des = sPalabra.length() - 1;
        boolean bError = false;
        while ((inc < des) && (!bError)){
            if (sPalabra.charAt(inc) == sPalabra.charAt(des)){
                inc++;
                des--;
            } else {
                bError = true;
            }
        }
    }
}
```

- a) El programa no compila.
- b) Cuenta las letras que hay, en una palabra.
- c) Verifica si una palabra es un palíndromo.

18. ¿Cuál de las siguientes opciones son verdaderas? (elija todas las correctas)

- a) Java es un lenguaje orientado a objetos.
- b) El código Java compilado en Windows puede ejecutarse en Linux.
- c) Java permite la sobrecarga de operadores
- d) Java es un lenguaje de programación funcional. ← No sabemos
- e) Java es un lenguaje procedimental. ← No sabemos
- f) Java tiene punteros a ubicaciones específicas en la memoria.

19. ¿Qué es Maven y para qué se utiliza en el desarrollo de aplicaciones?

- a) Maven es un lenguaje de programación. Se utiliza en el desarrollo de aplicaciones Java para escribir código.
- b) Maven es un servidor de base de datos. Se utiliza en el desarrollo de aplicaciones java para alojar datos.
- c) Maven es un sistema de control de versiones. Se utiliza en el desarrollo de aplicaciones java.
- d) Maven es una herramienta de gestión de dependencias. Se utiliza en el desarrollo de aplicaciones en el proyecto.

20. ¿Cuál de lo siguiente es cierto? (elija todas las correctas)

- a) javac compila un archivo .java en un archivo .bytecode. ← Si no tuviera el punto sería correcta
- b) Java toma el nombre del archivo .bytecode como parámetro.
- c) javac compila un archivo .java en un archivo .class
- d) Java toma el nombre de la clase como parámetro.
- e) Java toma el nombre del archivo .class como parámetro.
- f) javac compila un archivo .class como archivo java.

21. ¿Qué es Git y cuáles son algunos de sus comandos básicos?

- a) Git es un lenguaje de programación. Algunos comandos básicos de Git incluyen "print" e "if-else".

- b) Git es una herramienta para el análisis de código. Algunos comandos básicos de Git incluyen "analyze"... (no se ve).
- c) Git es un sistema de control de versiones. Algunos comandos básicos de Git incluyen "commit" y "push".
- d) Git es una herramienta para realizar pruebas de software. Algunos comandos básicos de Git incluyen... no se ve, pero no es correcta.

22. Dados los siguientes segmentos de código, ¿Qué respuesta no es una implementación de java válida?

- a) `int variableA = 10;`
`float variableB = 10.5f;`
`int variableC = variableA + variableB;`
- b) `byte variableA = 10;`
`double variableB = 10.5f;`
`double variableC = variableA + variableB;`
- c) `byte variableA = 10;`
`float variableB = 10.5f;`
`float variableC = variableA + variableB;`

23. ¿Qué escenario es el mejor uso de una excepción?

- a) La computadora se incendió.
- b) No sabe cómo codificar un método.
- c) No se encuentra un elemento al buscar en una lista.
- d) Se pasa un parámetro inesperado a un método.
- e) Quiere recorrer una lista.

24. ¿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.

25. 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

NINGUNA ES CORRECTA

26. ¿Cuáles de las siguientes afirmaciones sobre el polimorfismo son verdaderas?
(Elija todas las correctas)

- a) Si un método toma una superclase de 3 objetos, cualquiera de esas clases puede pasarse como parámetro del método

- b) Un método que toma un parámetro con tipo `java.lang.Object` tomará cualquier referencia
- c) Una referencia a un objeto se puede convertir a una subclase de objetos en una conversión explícita \leftarrow Cast
- d) Todas las excepciones de conversión se pueden detectar en tiempo de compilación
- e) Al definir un método de instancia pública en la súper clase, garantiza que el método específico se llamará al método en la clase principal en tiempo de ejecución

27. ¿Son patrones de diseño de software estructural?

- a) Adapter, Proxy, Prototype y Bridge.
- b) Adapter, Bridge, Proxy y Composite.
- c) Agile, Builder, Singleton y Prototype.
- d) Builder, Singleton y Prototype y Abstract Factory.

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

```
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);
        double result = numbers.stream()
            .mapToInt(n -> n)
            .average()
            .orElse(0);
        System.out.println(result);
    }
}
```

- a) 3.0
- b) 1
- c) 5
- d) 2.5

29. ¿Qué son las pruebas de integración?

- a) Pruebas que comprueben el rendimiento de la aplicación.
- b) Pruebas que comprueben el funcionamiento de la interfaz de usuario.
- c) Pruebas que comprueban el funcionamiento de varias unidades de código juntas.
- d) Pruebas que comprueban el funcionamiento de una sola unidad de código.

30. ¿Qué comando se utiliza para enviar los cambios confirmados en un repositorio local al repositorio remoto?

- a) git push
- b) it pull
- c) git commit
- d) git add

31. Seleccione la respuesta correcta, dado el siguiente bloque de código.

```

class ClassX{
    7 usages
    static int y = 2;
    1 usage
    ClassX(int x){
        this();
        y = y * 2;
    }

    1 usage
    ClassX(){
        y++;
    }
}

1 usage
public class Class2 extends ClassX{
    1 usage
    Class2(){
        super(y);
        y = y + 3;
    }

    public static void main(String[] args) {
        new Class2();
        System.out.println(y);
    }
}

```

- a) Error de compilación
- b) 9
- c) 7
- d) No se ve, pero la correcta es 9

32. ¿Cuál es el comando utilizado para crear una nueva rama en Git?

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

33. ¿Cuál es el resultado de compilar la siguiente clase?


```

public class Book {
    3 usages
    private int ISBN;
    private String title, author;
    private int pageCount;

    public int hashCode(){
        return ISBN;
    }

    public boolean equals(Object obj){
        if(!(obj instanceof Book)){
            return false;
        }
        Book other = (Book) obj;
        return this.ISBN == other.ISBN;
    }
}

```

- a) Línea 15 no compila porque other.ISBN es un atributo con modificador de acceso private.
- b) Línea 14 no compila porque no está declarada o manejada ClassCastException
- c) No compila porque no está sobrescribiendo el método equals correctamente.
- d) La clase compila satisfactoriamente.

34. ¿Cuál es la primer línea en fallar al compilar?

```

1  class Tool {
2  private void repair() {} //r1
3  1 override
4  void use(){}
5  }
6  class Hammer extends Tool{
7  private int repair(){return 0; } //r3
8  private void use(){}//r4
9  public void bang(){}//r5
10 }
11 |

```

- a) r5
- b) r4
- c) r3
- d) Ninguna de las anteriores.

35. ¿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.

36. ¿Cuál de las siguientes excepciones lanza la JVM? (Elija todas las correctas)

- a) `ArrayIndexOutOfBoundsException` (`RuntimeException`)
- b) `NumberFormatException`. (`RuntimeException`)
- c) `ExceptionInInitializerError` (`Error`)

- d) `Java.io.IOException` (Exception)
- e) `NullPointerException` (RunTimeException)

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

- a) `git Branch`
- b) `git merge`
- c) `git push`
- d) `git pull`

38. ¿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 un 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 es 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.

39. ¿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`

40. ¿Qué es un microservicio?

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

41. Dado el siguiente código:

```
public class Main {  
    public static void main(String[] args) {  
        int[] numeros = {1,2,3,4,5};  
        int suma = 0;  
        for (int i = 1; i <= numeros.length; i++){  
            suma += numeros[i];  
        }  
  
        System.out.println("La suma de los números es: " + suma);  
    }  
}
```

¿Este código compila sin errores?

- a) Si, compila sin errores.
- b) No, hay un error en el ciclo for.
- c) No, hay un error en la inicialización de la variable "suma".
- d) No, hay un error en la declaración del arreglo.

42. ¿Qué método se utiliza para obtener el mensaje de una excepción en Java?

- a) getClass()
- b) printStackTrace()
- c) toString()
- d) getMessage()

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

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

En esta te pide que selecciones varias, pero no se puede

44. ¿Cuál es el resultado de ejecutar el siguiente código?

```
String s = "hello";  
s.toUpperCase();  
System.out.println(s);
```

- a) NullPointerException

- b) hello
- c) HELLO
- d) Hello

45. ¿Cuál es el paquete de importación necesario para usar la clase ArrayList?

- a) import.java.net.*;
- b) import.java.awt.*;
- c) import.java.io.*;
- d) import.java.util.*;

No debería llevar punto entre el import y java, lo que haría que ninguna es correcta

46. ¿Cuál es el formato de los datos que se envían y reciben en una API REST?

- a) YAML
- b) XML
- c) JSON
- d) Todos los anteriores

47. ¿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. <- CORRECTA 3
- 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. <- Esta es correcta 1
- d) El operador de doble dos puntos se utiliza para acceder a métodos no estáticos en Java 8. <- Esta también es correcta en ciertos escenarios 2

48. ¿Qué palabra clave se utiliza para definir una excepción personalizada en Java?

- a) try
- b) throw
- c) finally

d) catch

49. ¿Cuál de los siguientes comandos elimina el directorio target antes de iniciar el proceso de construcción?

- a) mvn site
- b) mvn build
- c) mvn answer
- d) mvn clean

50. ¿Cuál es el comando utilizado para ver el historial de cambios en Git?

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

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

- a) Una expresión lambda es una forma de escribir una clase anónima en Java 8. <=INCORRECTA
- b) Una expresión lambda es una forma de escribir una función anónima en Java 8. <=CORRECTA
- 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.

52. ¿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
- c) Uno
- d) Otro número

53. 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

54. ¿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

55. ¿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.

56. 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

57. ¿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. ← Se puede
- c) Ambos pueden contener métodos predeterminados. ←No estamos seguros
- 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.

58. ¿Cuál no es un objetivo de Maven?

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

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

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

60. ¿Qué es un repositorio remoto en Git?

- 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.

61. 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"));`

62. ¿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

63. ¿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

64. ¿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.

65. ¿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.

66. ¿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

67. ¿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.

68. 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";

69. 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.

70. 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

71. ¿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

72. ¿Cuál de los siguientes es cierto acerca de una subclase concreta?

- 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.

73. ¿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

74. 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

```

- a) 3
- b) 9

c) 14

d) 6

75. ¿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;

76. ¿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.

77. ¿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.

78. ¿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.

79. 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.

80. ¿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) @Controller, @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 crean un bean y exponen la serialización de las clases mientras que @Controller. @Component requiere de una inicialización manual.

81. ¿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.

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

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

83. 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));
    }
}

```

- 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]

84. ¿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.

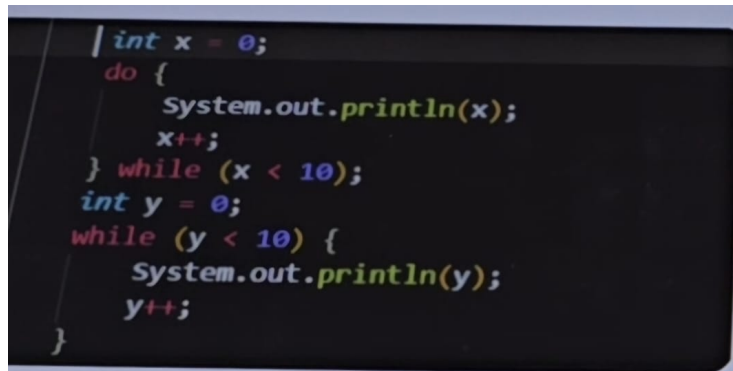
85. ¿Qué utilidad de línea de comandos basada en MS Windows le permitirá ejecutar el intérprete de Java sin abrir la ventana de la consola?

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

86. ¿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.

87. ¿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

88. ¿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:

89. ¿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

90. ¿Cómo se manejan las excepciones en Java?

- a) Las excepciones el 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.

91. ¿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

92. 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

```

package my.city;
import my.school.*;
public class School {
    public static void main(String[] args) {
        System.out.println(Classroom.globalKey);
        Classroom room = new Classroom(101, "Mrs. Anderson");
        System.out.println(room.roomNumber);
        System.out.println(room.floor);
        System.out.println(room.teacherName);
    }
}

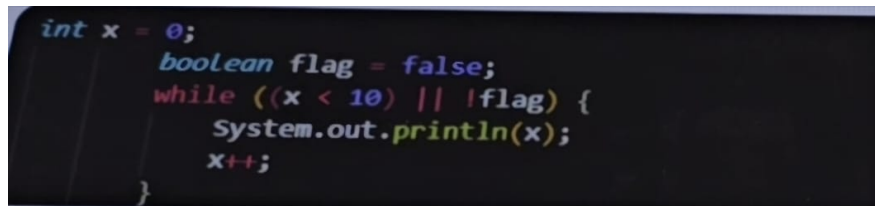
```

Aquí dejo el código que muestra las líneas que tienen error.

93. ¿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 ← Es correcto
- d) Un método que no tiene cuerpo

94. ¿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 ← Es correcto
- d) muestra los números del 0 al 9

95. ¿Qué son las anotaciones en java?

- a) Una forma de declarar variables en Java.
- b) Una forma de declarar métodos abstractos en Java.
- c) Un mecanismo para etiquetar y procesar código de forma especial.
- d) Comentarios especiales que se utilizan para documentar el código.

96. ¿Qué son las expresiones regulares en Java?

- a) Un mecanismo para validar y manipular fechas y horas.

- b) Un mecanismo para validar y manipula cadenas de caracteres.
- c) Un mecanismo para validar y manipular números enteros.
- d) Un mecanismo para validar y manipular números decimales.

97. ¿Qué es una anotación en Spring?

- a) Una biblioteca de terceros que se utiliza para extender la funcionalidad de Spring
- b) Una clase que se utiliza para definir la estructura de una tabla de base de datos.
- c) Una etiqueta que se utiliza para anotar una clase o un método y proporcionar información adicional al contenedor de Spring.
- d) Un archivo de configuración XML que se utiliza para configurar una aplicación de Spring.

98. ¿Cuál es la salida?

```
import java.util.ArrayList;

public class OtherExample {
    public static void main(String[] args) {
        var list = new ArrayList<String>();
        list.add("Austin");
        list.add("Boston");
        list.add("San Francisco");
        var c :long = list.stream()
            .filter(a -> a.length() > 10) //línea x
            .count();
        System.out.println(c + " " + list.size());
    }
}
```

- a) Ninguna de las anteriores
- b) 1 3
- c) 1 1
- d) El código no compila en la línea x

e) 2 3

99. ¿Cuál es la salida de la siguiente aplicación?

```
interface Speak { default int talk(){ return 7;} }
interface Sing { default int talk(){ return 5; }}

public class Performance implements Speak, Sing {
    public int talk(String... x) {
        return x.length;
    }

    public static void main(String[] notes) {
        System.out.println(new Performance().talk());
    }
}
```

- a) 5
- b) 7
- c) El código compila sin problemas la salida no se puede determinar hasta el tiempo de ejecución.
- d) Ninguna de las anteriores.
- e) El código no compila

100. ¿Qué conjunto de modificadores, cuando son agregados a un método default dentro de una interfaz, evitan que sea sobrescrito por la clase que lo implementa?

- a) private
- b) const
- c) final
- d) private static
- e) Ninguna de las anteriores
- f) Static

101. ¿Qué tipo de excepción se produce cuando se intenta realizar una operación incompatible con el tipo de datos en Java?

- a) `ArrayIndexOutOfBoundsException`
- b) `ArithmeticException`
- c) `IllegalArgumentException`
- d) `ClassCastException`

102. ¿Qué es un starter?

- a) Es una herramienta que nos facilita la creación y configurar un proyecto cargando las dependencias necesarias para un objetivo.
- b) Es el inicializador de un proyecto en Java.
- c) Componente encargado de realizar las operaciones de balanceador.

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

```
public class Test2 extends Thread{
    public static void main(String[] args) {
        protected Thread t = new Thread(new Test2());
        Thread t2 = new Thread(new Test2());

        t.start();
        t2.start();
    }

    public void main(){
        for (int i = 0; i < 2; i++)
            System.out.println(Thread.currentThread().getName() + " ");
    }
}
```

La respuesta es: **NO COMPILA**

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

`docker build`

2. ¿Seleccione la respuesta que considere correcta, dado el siguiente bloque de código?

```
import java.util.Arrays;
```

```
public class Example {
```

```
    public static void main( String [] args ) {  
        int [][] matrix= {{ { 1, 2}, {3, 4}}, {{5, 6}, {7, 8}}};  
        int [] flattened= Arrays.stream(matrix)  
                                .flatMapToIn(layer)Arrays.string(layer)  
                                .flatMapToIn(Arrays::stream))  
                                .boxed().map(n -> new int[(n)).toArray(int[] []::new);  
  
        System.out.println(Arrays.deepToString(flattened));  
    }  
}
```

[[1], [2], [3], [4], [5], [7], [8]]

[[1, 2]], [[3, 4]], [[5, 6]], [[7, 8]]

[[1, 2], [3,4], [5,6], [7,8]]