🧠 Simulacro Examen Final de Programación Java

Parte A: Cuestiones Cortas (1 punto cada una)

1. **¿Qué es la sobrecarga de métodos? ¿En qué se diferencia de la sobrescritura?**

   La **sobrecarga (overloading)** ocurre cuando varios métodos en una misma clase tienen el mismo nombre pero diferentes parámetros (tipo, número o ambos).

   La **sobrescritura (overriding)** ocurre cuando una subclase redefine un método ya existente en su superclase. Se indica con `@Override`.

2. **Explica qué es una clase abstracta y cómo se diferencia de una interfaz.**

   Una **clase abstracta** puede contener atributos y métodos concretos y abstractos (sin implementar). No se puede instanciar y se hereda con `extends`.

   Una **interfaz** es un contrato que define métodos sin implementar (salvo métodos `default` o `static`). Se implementa con `implements`, y una clase puede implementar varias interfaces, pero solo heredar de una clase abstracta.

3. **¿Qué hace el modificador `static` en una variable o método?**

   El modificador `static` indica que el elemento pertenece a la clase en sí, no a sus objetos. Se usa, por ejemplo, para contar instancias, o para métodos auxiliares que no dependen del estado del objeto.

4. **¿Qué es la serialización de objetos? ¿Qué clase se usa para ello en Java?**

   La serialización convierte un objeto en una secuencia de bytes para almacenarlo o transmitirlo. En Java, se usa `ObjectOutputStream` para serializar y `ObjectInputStream` para deserializar. La clase debe implementar la interfaz `Serializable`.

5. **Explica qué ocurre si lanzamos una excepción que no se captura. ¿Qué palabra clave se usa para declarar esto en la firma del método?**

   (Si se lanza una excepción **no capturada**, el programa se detiene y muestra un error. Para indicar que un método puede lanzar una excepción, se usa la palabra clave `throws` en su firma.

   Ejemplo: `public void leerArchivo() throws IOException`

6. **¿Qué diferencia hay entre `ArrayList` y un array tradicional en Java?**

Un **array tradicional** tiene tamaño fijo y puede almacenar datos primitivos o referencias.

Un **ArrayList** es una colección dinámica que puede crecer o reducirse automáticamente. Solo almacena objetos y pertenece al paquete `java.util`.

7. **¿Para qué sirve el método `compareTo()` de la interfaz `Comparable`? ¿Qué valores puede devolver?**

   `compareTo()` compara el objeto actual con otro del mismo tipo. Devuelve:

   `0` si son iguales
   Un número negativo si el actual es menor
   Un número positivo si es mayor

8. **Explica brevemente qué es el polimorfismo. Pon un ejemplo simple.**

   El **polimorfismo** permite tratar objetos de diferentes clases de forma uniforme si comparten una superclase o interfaz.

   Ejemplo:

   Producto p1 = new Libro();

   Producto p2 = new Pelicula();

   List<Producto> lista = new ArrayList<>();

   lista.add(p1); lista.add(p2);

9. **¿Qué método hay que sobrescribir para ordenar objetos con `Collections.sort()`?**

   Se debe sobrescribir el método `compareTo()` si se implementa la interfaz `Comparable`.

10. **¿Qué utilidad tiene el bloque `finally` en el tratamiento de excepciones?**

    El bloque `finally` se ejecuta siempre, ocurra o no una excepción. Se usa para liberar recursos como cerrar archivos o conexiones.