

2do. Parcial JAVA Plan 2024 - Tema B

Complete las respuestas y presione **"Enviar"**. Solo se aceptará un sólo envío. Controle que sus datos e email sean correctos, caso contrario no recibirá devolución a vuelta de correo. Código **fuelle igual o repetido**, calificará 0 (cero) en alumnos involucrados, reservándose la Cátedra la aplicación del Reglamento de Estudio vigente. Asegúrese que su código fuente compile en la IDE de referencia. A criterio de la Cátedra, podrán realizarse **"Coloquios"** sobre el código fuente presentado.-

[Iniciar sesión en Google](#) para guardar lo que llevas hecho. [Más información](#)

*** Indica que la pregunta es obligatoria**

Correo *

avalussi.facultad@gmail.com



Ejercicio 3: Analizar la Igualdad

10 puntos

En base al proyecto Tecnologia:

<https://github.com/facundouferer/CursoDeJava/tree/Desarrollo/src/Parciales/Parcial2025/Segundo/Tecnologia>

En la clase Computadora, implementar los métodos para asegurar que dos computadoras sean consideradas iguales únicamente si tienen el mismo número de serie.

```
public boolean equals(Object o) {  
    if (this == o)  
        return true;  
    if (o == null || !(o instanceof Computadora))  
        return false;  
    Computadora Computadora = (Computadora) o;  
    return NumeroSerie.equals(computadora.NumeroSerie);  
}
```



Ejercicio 4: Buscar elemento

10 puntos

En base al proyecto Tecnologia:

<https://github.com/facundouferer/CursoDeJava/tree/Desarrollo/src/Parciales/Parcial2025/Segundo/Tecnologia>

Implemente el método buscarComputadora(String numeroSerie) en la clase InventarioComputadoras.

El método debe cumplir con los siguientes requisitos:

- El sistema cuenta con dos listas internas: Escritorio y Laptop.
- El método debe recibir un número de serie como parámetro.
- Debe buscar primero en la lista de escritorios y si encuentra un escritorio cuyo número de serie coincida exactamente con el número recibido, debe retornar ese objeto.
- Si no se encontró en escritorios, debe buscar en la lista de laptops y si encuentra una laptop con el número de serie indicado, debe retornar esa laptop.
- Si el número de serie no coincide con ningún dispositivo de ambas listas, el método debe retornar null.

```
public Computadora buscarComputadora(String numeroSerie) {  
  
    for (Escritorio NumeroSerie : Escritorio) {  
        if (Computadora.getnumeroSerie().equals(numeroSerie)) {  
            return NumeroSerie;  
        }  
    }  
  
    // Si no lo encontró antes, busca en la lista de Laptop.  
  
    for (Escritorio NumeroSerie : Laptop) {  
        if (Computadora.getnumeroSerie().equals(numeroSerie)) {  
            return NumeroSerie;  
        }  
    }  
}
```



Dado el código en imagen adjunta, marque la opción INCORRECTA.

3 puntos

```
2. import java.util.*;
3. class Cereal { }
4. public class Flakes extends Cereal {
5.     public static void main(String[] args) {
6.         List<Flakes> c0 = new List<Flakes>();
7.         List<Cereal> c1 = new ArrayList<Cereal>();
8.         List<Cereal> c2 = new ArrayList<Flakes>();
9.         List<Flakes> c3 = new ArrayList<Cereal>();
10.        List<Object> c4 = new ArrayList<Flakes>();
11.        ArrayList<Cereal> c5 = new ArrayList<Flakes>();
12.    }
13. }
```

- ☐ Compilación falla debido a error en línea 6.
- ☐ Compilación falla debido a error en línea 7.
- ☐ Compilación falla debido a error en línea 8.
- ☐ Compilación falla debido a error en línea 9.
- ☐ Compilación falla debido a error en línea 10.
- ☒ Compilación falla debido a error en línea 11.

¿Qué hace el método void close () de la Clase BufferedWriter?

1 punto

- ☐ Cierra el flujo de escritura y cierra el archivo.
- ☒ Vuelca el contenido del Buffer al archivo.
- ☐ Escribe un salto de línea en el archivo.



Dada la imagen adjunta, seleccione la afirmación INCORRECTA.

3 puntos

- a. Únicamente las clases que implementan la interfaz List permiten el uso de iteradores.
- b. Un iterador es un objeto que proporciona funcionalidad para recorrer todos los elementos de una colección.
- c. Un iterador permite recorrer cualquier tipo de colección hacia adelante utilizando el método next() combinado con el método hasNext() para comprobar si se ha alcanzado el final de la colección.
- d. Una colección puede recorrerse tanto con un iterador como con un ciclo for-each. Ambas formas son equivalentes.

- ☐ a
- ☐ b
- ☐ c
- ☒ d

¿Qué hace el método void flush () de la Clase BufferedWriter?

1 punto

- ☒ Cierra el flujo de escritura y cierra el archivo.
- ☐ Vuelca el contenido del Buffer al archivo.
- ☐ Escribe un salto de línea en el archivo.



Dado el código adjunto, cual sería el resultado?

3 puntos

```
3. public static void main(String[] args) {  
4.     try {  
5.         throw new Error();  
6.     }  
7.     catch (Error e) {  
8.         try { throw new RuntimeException(); }  
9.         catch (Throwable t) { }  
10.    }  
11.    System.out.println("pew");  
12. }
```

- ☒ "pew"
- ☐ No se genera salida.
- ☐ Compilación falla en línea 5.
- ☐ Compilación falla en línea 7.
- ☐ Compilación falla en línea 8.
- ☐ Compilación falla en línea 9.

¿Cuál de las sgtes. es una característica de java.lang.Exception?

2 puntos

- ☒ private
- ☒ extends Throwable
- ☐ implements Throwable
- ☐ final
- ☐ implements Externalizable



Seleccione según imagen adjunta, la opción CORRECTA.

4 puntos

Pregunta: En la definición de una interface en Java :

- a. Es necesaria emplear la palabra clave abstract.
- b. La signatura de los métodos de una interfaz tienen visibilidad public o private, pero no protected.
- c. No se permiten campos constantes.
- d. Aunque no se indique usando la palabra clave final, todos los campos son tratados como si así fuesen.

- ☒ a
- ☐ b
- ☐ c
- ☐ d

Nombres y Apellido - Legajo *

Agustin Valussi - 30353

¿Qué hace el método void write (int car) de la Clase BufferedWriter?

1 punto

- ☐ Escribe una porción de un Array en el archivo.
- ☐ Escribe un caracter en el archivo.
- ☒ Escribe una porción de una cadena de caracteres en el archivo.



Ejercicio 1: Implementar la interface

15 puntos

En base al proyecto Tecnologia:

<https://github.com/facundouferer/CursoDeJava/tree/Desarrollo/src/Parciales/Parcial2025/Segundo/Tecnologia>

Implemente en la clase Laptop el método requerido por la interface correspondiente.

El cálculo del precio final debe considerar:

- Una depreciación del 12% por cada año de uso, debido a que las laptops pierden valor más rápido que las computadoras de escritorio.
- Un descuento adicional del 15% relacionado al desgaste típico de portabilidad (batería, pantalla, bisagras, etc.).

El método debe:

- Calcular los años de uso,
- Aplicar la depreciación anual,
- Aplicar el descuento por portabilidad,

Y retornar el precio final resultante.

```
public double anioActual (double precioBase, int anioActual ) {  
    // Calcular depreciación por años de uso  
    double deprecAño = this.anioActual * 0.12;  
  
    // Calcular depreciación por desgaste tipico  
    double deprecAdic = this.anioActual * 0.15;  
  
    // Calcular costo final  
    double PrecioVenta = deprecAño * (1 + deprecAño);  
    return costoConEdad * (1 - deprecAdic);  
}
```



Seleccione opción, según imagen adjunta.

2 puntos

Pregunta: indique cuál de las siguientes definiciones de un método m, que lanza IOException, y que devuelve void, es correcta:

- a. `void m() throws IOException {}`
- b. `void m() throw IOException {}`
- c. `void m(void) throws IOException {}`
- d. `void m() {} throws IOException`

☒ a

☐ b

☐ c

☐ d

Elija la opción correcta a la pregunta :

2 puntos

¿A qué Clase pertenece la definición : "Los caracteres escritos se transforman previamente en bytes"?

☒ OutputStreamWriter

☐ InputStreamReader

☐ PublicStreamReader



Dado el código en imagen adjunta, cual sería su salida por consola?

3 puntos

```
public class test {  
    public static void main(String args[]) {  
        int i=1, j=1;  
        try {  
            i++;  
            j--;  
            if(i == j)  
                i++;  
        }  
        catch(ArithmeticException e) {  
            System.out.print(0);  
        }  
        catch(ArrayIndexOutOfBoundsException e) {  
            System.out.print(1);  
        }  
        catch(Exception e) {  
            System.out.print(2);  
        }  
        finally {  
            System.out.print(3);  
        }  
        System.out.print(",4");  
    }  
}
```

- a. 0,4
- b. 1,4
- c. 2,4
- d. 3,4

- ☐ a
- ☐ b
- ☒ c
- ☐ d



Ejercicio 2: Excepciones

15 puntos

En base al proyecto Concesionario:

<https://github.com/facundouferer/CursoDeJava/tree/Desarrollo/src/Parciales/Parcial2025/Segundo/Tecnologia>

Implemente una excepción que deberá lanzarse al intentar crear un objeto de tipo Escritorio que posea menos de 5 puertos disponibles.

La excepción debe ser verificada, por lo que el código que crea un Escritorio deberá manejarla mediante try/catch o declararla con throws.

La clase de la excepción debe incluir:

Un constructor que reciba un mensaje personalizado.

Un constructor por defecto con un mensaje predeterminado indicando que un escritorio debe tener al menos 5 puertos.



//En la clase PuertosInsuficientesException.java

```
public class PuertasInsuficientesException extends Exception {  
    /**  
     * Constructor que permite crear la excepción con un mensaje de error específico.  
     */  
    public PuertasInsuficientesException (String message) {  
        super(message);  
    }  
}
```

//En la clase Auto

Vehiculo nuevoVehiculo;

// Según el tipo elegido, se piden los datos específicos y se crea el objeto
// correspondiente.

```
if (tipo == 1) {  
    System.out.print("Número de puertas: ");  
    int puertas = menu.leerOpcion();  
    nuevoVehiculo = new Auto(marca, modelo, anio, patente, km, puertas);  
} else if (tipo == 2) {  
    System.out.print("Capacidad de carga (toneladas): ");  
    double carga = menu.leerDouble();  
    nuevoVehiculo = new Camion(marca, modelo, anio, patente, km, carga);  
} else {  
    System.out.println("Tipo de vehículo no válido.");  
    return;  
}
```

flota.add(nuevoVehiculo);

System.out.println("Vehículo registrado exitosamente.");

```
} catch (PuertasInsuficientesException e) { // Maneja el error segun cantidad de puertas.  
    System.out.println("Error al crear el vehículo: " + e.getMessage());  
} catch (Exception e) { // Maneja cualquier otro error inesperado.  
    System.out.println("Ocurrió un error inesperado: " + e.getMessage());  
}  
}
```



¿Qué interfaz proporciona capacidad de almacenar datos usando pares (clave , valor)?

1 punto

- a. **Java.util.Map.**
- b. **Java.util.Set.**
- c. **Java.util.List.**
- d. **Java.util.Collection.**

- ☐ a
- ☒ b
- ☐ c
- ☐ d

Dado el código fuente en imagen adjunta, seleccione la respuesta **INCORRECTA**.

3 puntos

```
public class Person implements Serializable {  
    private int age;  
    private String name;  
    private Address country;  
}
```

- ☐ Address debe implementar Serializable.
- ☐ Address debe implementar Serializable, sino se lanza `NotSerializableException`.
- ☐ No todas las subclases de Person podrían ser serializables.
- ☐ El tipo de datos de "country" debe ser serializable.
- ☒ No se produce error de compilación.



¿A qué Clase pertenece la sgte. definición : " abre un archivo de texto en modo lectura"?

1 punto

- ☐ FileWriter
- ☒ FileReader
- ☐ FileReader

Dado el código en imagen adjunta, cual sería el resultado por Consola?

5 puntos

```
2. class Noodle {
3.     String name;
4.     Noodle(String n) { name = n; }
5. }
6. class AsianNoodle extends Noodle {
7.     public boolean equals(Object o) {
8.         AsianNoodle n = (AsianNoodle)o;
9.         if(name.equals(n.name)) return true;
10.        return false;
11.    }

12.    public int hashCode() { return name.length(); }
13.    AsianNoodle(String s) { super(s); }
14. }
15. public class Soba extends AsianNoodle {
16.     public static void main(String[] args) {
17.         Noodle n1 = new Noodle("bob"); Noodle n2 = new Noodle("bob");
18.         AsianNoodle a1 = new AsianNoodle("fred");
19.         AsianNoodle a2 = new AsianNoodle("fred");
20.         Soba s1 = new Soba("jill"); Soba s2 = new Soba("jill");
21.         System.out.print(n1.equals(n2) + " " + (n1 == n2) + " | ");
22.         System.out.print(a1.equals(a2) + " " + (a1 == a2) + " | ");
23.         System.out.println(s1.equals(s2) + " " + (s1 == s2));
24.     }
25.     Soba(String s) { super(s); }
26. }
```

- ☐ Compilación falla.
- ☐ tru e true | true true | true true
- ☐ true false | true false | true false
- ☒ false false | true false | true false
- ☐ false false | true false | false false
- ☐ false false | false false | false false



Según imagen adjunta, seleccione la opción CORRECTA.

2 puntos

Pregunta: Respecto a las excepciones en Java, podemos afirmar ...

- a. Todas las subclases de la clase estándar de Java `RuntimeException` son excepciones comprobadas.
- b. Todas las subclases de la clase estándar de Java `Exception` son excepciones comprobadas.
- c. `Error` es una subclase directa de `Throwable`, mientras que `Exception` es una subclase directa de `Error`.
- d. Tanto `Error` como `Exception` son subclases directas de `Throwable`.

- ☒ a
- ☐ b
- ☐ c
- ☐ d

Responda según imagen.

3 puntos

Pregunta: Dado el siguiente código, ¿Cuál de las siguientes afirmaciones es correcta?

```
Set < Object > objetos = new HashSet<Object>();  
String obj1 = "JAVA";  
int obj2 = 5;  
Boolean obj3 = new Boolean(true);  
objetos.add(obj3);  
objetos.add(obj1);  
objetos.add(obj2);  
objetos.add(obj3);  
for(Object object : objetos) {  
    System.out.print(object);  
}
```

- a. Error en tiempo de ejecución.
- b. Se muestran por pantalla `JAVA` y `true` en un orden no determinado.
- c. Se muestran por pantalla `JAVA` y `true` en el orden exacto en el que fueron insertadas en la colección.
- d. Se muestran por pantalla `JAVA` y `true` en un orden no determinado y, además, "`true`" se muestra dos veces.

- ☐ a
- ☐ b
- ☐ c
- ☒ d



Seleccione según imagen adjunta, la opción CORRECTA.

3 puntos

Pregunta: ¿Qué código hay que añadir en la posición indicada en el código para que compile?

```
public class ExceptionTest {  
    class TestException extends Exception {}  
    public void runTest() throws TestException {}  
    public void test() /* Código a añadir */ {  
        runTest();  
    }  
}
```

a. No hay que añadir código alguno.
b. throws Exception
c. catch (Exception e)
d. throws RuntimeException

- ☐ a
- ☒ b
- ☐ c
- ☐ d

Elija la opción correcta a la pregunta :

2 puntos

¿A qué Clase pertenece la definición : "Lee bytes y los transforma a caracteres"?

- ☒ OutputStreamWriter
- ☐ InputStreamReader
- ☐ PublicStreamReader

¿Qué Clase provee una representación/asociación abstracta de ficheros y directorios?

1 punto

- ☒ ObjectInputStream
- ☐ File
- ☐ FileInputStream



¿A qué Clase pertenece la sgte. definición : " abre un archivo de texto en modo escritura"?

1 punto

- ☐ FileReader
- ☒ FileWriter
- ☐ FileOutput

¿Qué hace el método void newLine () de la Clase BufferedWriter?

1 punto

- ☐ Cierra el flujo de escritura y cierra el archivo.
- ☐ Vuelca el contenido del Buffer al archivo.
- ☒ Escribe un salto de línea en el archivo.

Seleccione la opción CORRECTA, según la afirmación en la imagen :

2 puntos

Pregunta: Un Set es una estructura:

- a. Que almacena cada elemento individual una sola vez como mínimo. No mantiene un orden específico.
- b. Que almacena cada elemento individual una sola vez como mínimo. Mantiene un orden específico.
- c. Que almacena cada elemento individual una sola vez como máximo. No mantiene un orden específico.
- d. Que almacena cada elemento individual una sola vez como máximo. Mantiene un orden específico.

- ☐ a
- ☐ b
- ☐ c
- ☒ d

[Enviar](#)

Página 1 de 1

[Borrar formulario](#)

Nunca envíes contraseñas a través de Formularios de Google.

Este contenido no ha sido creado ni aprobado por Google. - [Términos del Servicio](#) - [Política de Privacidad](#)

¿Parece sospechoso este formulario? [Informe](#)

Google Formularios



