ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL
PROGRAMACIÓN ORIENTADA A OBJETOS
EXAMEN SEGUNDO PARCIAL - 2025 - 2T

NOMBRE: **PARALELO:**

(5 puntos)Tema 1. Indique si es verdadero (V) o falso(F):

Una clase puede heredar de múltiples interfaces.	
Una clase puede ser abstract y final a la vez.	
El método findViewById se utiliza para vincular los elementos definidos en el archivo XML con las variables de la Activity.	
Las excepciones verificadas deben manejarse obligatoriamente con try-catch o throws.	
NullPointerException es una excepción verificada.	

(20 puntos) Tema 2

Analice el código mostrado a continuación y responda las preguntas de acuerdo con las instrucciones.

```
interface Alerta {
        String nivel();
        String recomendacion();
}
```

```
abstract class EventoClimatico implements Alerta {
        protected String provincia;

        public EventoClimatico(String provincia) {
            this.provincia = provincia;
        }

        public void mostrar() {
             System.out.println(descripcion() + " | " + provincia);
        }

        protected abstract String descripcion();

        public void emitirAlerta() {
           System.out.println("Nivel: " + nivel());
           System.out.println("Recomendación: " +
recomendacion());
        }
}
```

```
class LluviaIntensa extends EventoClimatico {
    private int mm;

    public LluviaIntensa(String provincia, int mm) {
        super(provincia);
        this.mm = mm;
    }

    @Override
    protected String descripcion() {
        return "Lluvia intensa (" + mm + " mm)";
    }

     @Override
```

```
class Sequia extends EventoClimatico {
    private int diasSinLluvia;

    public Sequia(String provincia, int   diasSinLluvia)
{
        super(provincia);
        this.diasSinLluvia = diasSinLluvia;
    }

    @Override
    protected String descripcion() {
        return "Sequía (" + diasSinLluvia + " días sin
lluvia)";
    }
```

```
    public String nivel() {
        if (mm >= 80) {
            return "ALTA";
        } else {
            return "MEDIA";
        }
    }

    @Override
    public String recomendacion() {
        if (mm >= 80) {
            return "Evitar zonas inundables";
        } else {
            return "Precaución al conducir";
        }
    }
}
```

```
    @Override
    public String nivel() {
        if (diasSinLluvia >= 30) {
            return "ALTA";
        } else {
            return "MEDIA";
        }
    }

    @Override
    public String recomendacion() {
        if (diasSinLluvia >= 30) {
            return "Ahorrar agua";
        } else {
            return "Monitorear consumo";
        }
    }
}
```

```
public class Main {
        public static void main(String[] args) {

            EventoClimatico e1 = new LluviaIntensa("Guayas", 90);
            EventoClimatico e2 = new Sequia("Manabí", 20);

            // Descomenta SOLO UNA:
            // System.out.println(e1.nivel() + " - " + e2.nivel()); //A
            // e2.mostrar();   //B
            // System.out(((Alerta) e1).recomendacion()); //C
            // ((EventoClimatico) e2).emitirAlerta(); //D
        }
}
```

Si se descomenta cada línea, ¿qué salida **real** produce el programa?

A) System.out.println(e1.nivel() + " - " + e2.nivel());

a) MEDIA - MEDIA
b) ALTA - MEDIA
c) ALTA - ALTA
d) MEDIA - ALTA

B) e2.mostrar();

a) Sequía (20 días sin lluvia) | Manabí
b) Sequía (30 días sin lluvia) | Manabí
c) Sequía (20 días sin lluvia) | Guayas
d) Nivel: MEDIA

C) System.out.println(((Alerta) e1).recomendacion());

a) Imprime: Evitar zonas inundables
b) Imprime: Precaución al conducir
c) No imprime nada
d) Da error de compilación

D) ((EventoClimatico) e2).emitirAlerta();

a)
Nivel: ALTA
Recomendación: Ahorrar agua
b)
Nivel: MEDIA
Recomendación: Monitorear consumo
c)
Nivel: BAJA
Recomendación: Evitar zonas inundables
d) Da error porque no se puede hacer casting

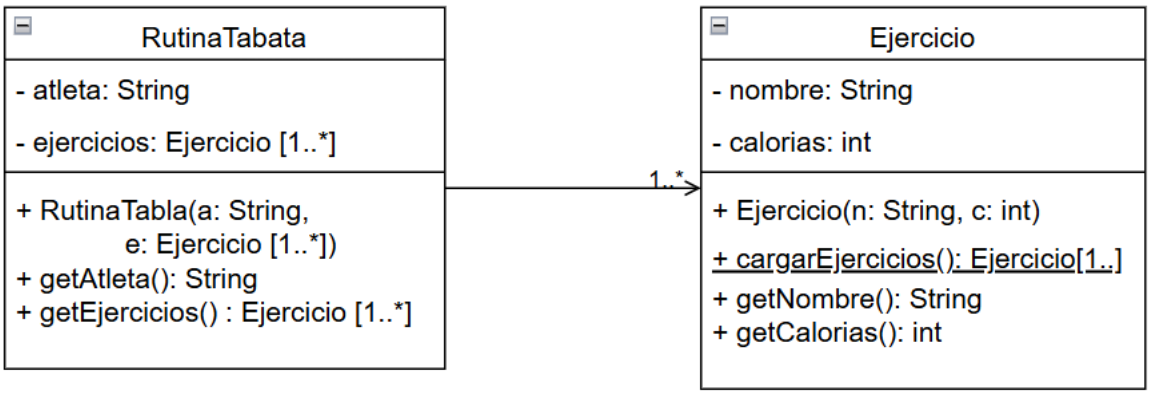
(75 puntos) TEMA 3. Desarrollo - Simulador de Rutinas Tábata

El método Tábata consiste en realizar 4 minutos de entrenamiento intenso:

- 8 rondas de:
  - 20 segundos de ejercicio
  - 10 segundos de descanso

Se desea crear un simulador que permita asignar distintos tipos de ejercicios para ejecutar una rutina Tabata completa.

A continuación, se muestra el diagrama de la capa modelo de la aplicación



- Una **rutina** tiene el nombre del atleta y una lista de ejercicios para esa rutina**.**
- Un Ejercicio tiene un nombre y el número de calorías estimadas que consume.

Existe un archivo llamado **ejercicios.txt** que contiene la información de los ejercicios y tiene la siguiente estructura:

Nombreejercicio,calorias

Ejemplo de archivo

```
Burpees,6
Squat Jumps,5
Mountain Climbers,4
High Knees,5
Jumping Jacks
Push-ups,3
Sprints,7
```

Considerando el diagrama de clases proporcionado, desarrolle lo indicado a continuación.

En el **paquete excepciones:**

3.1 Cree una excepción de tipo **unchecked** llamada **PocosEjerciciosException** con un constructor que recibe el mensaje de la excepción.

En el **paquete modelo:**

Clase Ejercicio

3.2 Implemente el método estático **cargarEjercicios** que lee el archivo ejercicios.txt y devuelve una lista de tipo Ejercicio.

Para el manejo de las excepciones considere lo siguiente:

- Si el archivo a leer no existe, muestre un mensaje en la consola indicando: "*Archivo ejercicios.txt no encontrado*".
- Si se produce un IOException muestre el mensaje "*Error al procesar el archivo*" más el mensaje del error.

- Recuerde capturar también los errores **durante el procesamiento del contenido del archivo** para que todas las líneas del archivo sean leídas. Si se origina un error, muestre por consola el mensaje "*Error procesando la línea* ", el contenido de la línea y el mensaje de error.

Si al finalizar la lectura del archivo la lista tiene menos de 2 ejercicios lance la excepción **PocosEjerciciosException,** con el mensaje: "*Pocos ejercicios para seleccionar*".

Recuerde incluir en la firma del método que se puede lanzar la excepción.

Clase RutinaTabata

3.3 Implemente el código necesario para que la **RutinaTabata** se comporte como un hilo:

Itera 8 veces (cada iteración es una ronda).

- En cada ronda:
    - Itera la lista de ejercicios y para cada ejercicio:
        - Imprime "Atleta <nombre> Ronda [n]: <nombre del ejercicio>".
        - Duerme el hilo 20 segundos
        - Imprime Atleta <nombre> Ronda [n]: Descanso iniciado".
        - Duerme el hilo 10 segundos
        - Imprime Atleta <nombre> Ronda [n]: Descanso finalizado".
        - Acumula las calorías del ejercicio

Al finalizar todas las rondas, debe mostrar un mensaje con el formato:

Atleta <nombre> - Total de calorías quemadas: <acumulado> kcal

Clase Main

- 3.4 Llame al método **cargarEjercicios** para obtener la lista de tipo **Ejercicio**, capture las excepciones necesarias.
- 3.5 Con la lista del paso anterior llame al método **asginarRutinas** que devuelve una lista de rutinas, e inicie la ejecución de las rutinas.
    - Recuerde que cada **Rutina** se comporta como un hilo.

Ejemplo de ejecución de las rutinas para Camila y Jaime, cada una con dos ejercicios.

```
Atleta <Camila> Ronda [1] <Mountain Climbers>
Atleta <Jaime> Ronda [1] <Burpees>
Atleta <Jaime> Ronda [1] descanso iniciado.
Atleta <Camila> Ronda [1] descanso iniciado.
Atleta <Jaime> Ronda [1] descanso terminado.
Atleta <Camila> Ronda [1] descanso terminado.
….
Atleta <Camila> Ronda [8] <High Knees>
Atleta <Jaime> Ronda [8] <Squat Jumps>
Atleta <Jaime> Ronda [8] descanso iniciado.
Atleta <Camila> Ronda [8] descanso iniciado.
Atleta <Camila> Ronda [8] descanso terminado.
Atleta <Jaime> Ronda [8] descanso terminado.
Atleta <Camila> - Total de calorías quemadas:72 kcal
Atleta <Jaime> - Total de calorías quemadas:88 kcal
```

Contenido de la clase Main

```
public class Main {
    public static void main(String[] args) {
        //COMPLETAR: llame a cargarRutinas
        //COMPLETAR: llame a asignarRutinas e inicie la ejecución de cada rutina









    }

    //ASUMA QUE ESTE MÉTODO YA ESTÁ IMPLEMENTADO
    public static ArrayList<RutinaTabata> asignarRutinas(ArrayList<Ejercicio> listaEj){
        //código que crea las rutinas
        return lista;
    }
}
```