📝 Evaluación Kotlin – Desarrollo Mobile

Duración: 70 minutos **Puntaje total:** 60 puntos

🔹 Pregunta 1: Colecciones (10 puntos)

Dada la siguiente colección de números:

```
val numeros = listOf(1, 4, 12, 4, 5, 45, 8, 10)
```

Realice las siguientes tareas:

- Cree una lista con todos los elementos **impares**. (**4 puntos**)
- Encuentre el **primer elemento** que sea **mayor a 11**. (**3 puntos**)
- Sume el **primer** y el **último elemento** de la lista. (**3 puntos**)

📌 **Nota:** Debe mostrar los resultados en consola utilizando `println`.

🔹 Pregunta 2: Ciclos (10 puntos)

Considere la siguiente clase y variable:

```
data class Alumno(
   val nombre: String,
   val nota1: Int,
   val nota2: Int,
   val nota3: Int
)
```

```
val listaAlumnos = listOf(
    Alumno("Alex", 55, 60, 40),
    Alumno("Iris", 30, 45, 62),
    Alumno("Juan", 50, 57, 63),
    Alumno("Nina", 68, 65, 60)
)
```

Utilice un **ciclo** para recorrer la lista y calcular el **promedio de notas** de cada alumno.

El promedio debe mostrarse en pantalla junto con el **nombre del alumno**.

🔹 Pregunta 3: Funciones de orden superior (10 puntos)

Implemente una función `operacion` que reciba función de suma y multiplicación como parámetro:

```
fun operacion(x: Int, y: Int, f: (Int, Int) -> Int): Int
```

Realice lo siguiente:

- Declare la función `operacion`. (**3 puntos**)
- Declare las funciones `suma` y `multiplicación`. (**3 puntos**)
- Utilice `operacion` para **sumar 2 + 2** y **multiplicar 2 × 2**. (**4 puntos**)

🔹 Pregunta 4: Condicionales (10 puntos)

Dada la variable:

```
val edad = 15
```

Realice lo siguiente:

- Cree un programa que clasifique a la persona como **mayor de edad** o **menor de edad**. (**6 puntos**)
- Valide que la edad ingresada sea **positiva** y **menor a 150**. (**4 puntos**)

🔹 Pregunta 5: Clases y Herencia (20 puntos)

Supongamos que usted debe desarrollar una plataforma estudiantil utilizando **Programación orientada a objeto**, **herencia** y **polimorfismo**.

Para ello realice lo siguiente:

- Cree la clase `Persona` con los atributos: `nombre`, `edad` y `rut`. Incluya el método `mostrarInfo()` para mostrar la información de esos parámetros. (**4 puntos**)
- Cree dos clases hijas:
    - `Estudiante`: que incluye `matrícula` y `carrera`.
    - `Profesor`: que incluye `especialidad`.

  Ambas clases deben heredar atributos y métodos de `Persona`. (**6 puntos**)

- Añada los métodos `estudiar()` y `explicar()` en las clases hijas `Estudiante` y `Profesor` `respectivamente.` Cada uno debe mostrar un mensaje personalizado. (**4 puntos**)
- Sobreescriba el método `mostrarInfo()` en ambas clases para incluir los nuevos atributos. (matrícula, carrera y especialidad según corresponda) (**6 puntos**)