**RESPUESTA A PRUEBA TÉCNICA**

1. **1. Preparación del entorno de producción:** Configura el servidor de producción con el sistema operativo y las dependencias necesarias. Esto puede incluir la instalación de un servidor web como Apache o Nginx, una base de datos como MySQL o PostgreSQL, y un entorno de ejecución como Node.js o Python.
2. **Configuración del dominio:** Configura el dominio de tu aplicación para que apunte a la dirección IP de tu servidor de producción.
3. **Preparación de la aplicación para la producción:** Esto puede incluir la minificación y la transpilación del código, la configuración de variables de entorno, y la preparación de la base de datos.
4. **Transferencia de la aplicación al servidor de producción:** Puedes usar herramientas como SCP o Rsync para transferir tu aplicación al servidor de producción, o puedes usar un sistema de control de versiones como Git.
5. **Instalación de dependencias:** Ejecuta los comandos necesarios para instalar las dependencias de tu aplicación en el servidor de producción.
6. **Configuración del servidor web:** Configura tu servidor web para servir tu aplicación. Esto puede incluir la configuración de rutas, la configuración de SSL para HTTPS, y la configuración de la caché.
7. **Iniciar la aplicación:** Inicia tu aplicación en el servidor de producción. Puedes usar un gestor de procesos como PM2 para Node.js o Gunicorn para Python para mantener tu aplicación en ejecución.
8. **Pruebas:** Realiza pruebas para asegurarte de que tu aplicación funciona correctamente en el entorno de producción.
9. **Monitoreo y mantenimiento:** Configura herramientas de monitoreo para mantener un ojo en el rendimiento de tu aplicación y para alertarte de cualquier problema.

**2.** import java.io.File

fun simularAccesoCamara(rutaArchivo: String): File {

// Simular la apertura de la cámara

println("Abriendo la cámara...")

// Simular la captura de una imagen

println("Capturando imagen...")

// Crear un archivo de imagen ficticio en la ruta especificada

val archivoImagen = File(rutaArchivo)

println("Imagen guardada en: ${archivoImagen.absolutePath}")

// Devolver el archivo de imagen

return archivoImagen

}

**3.**

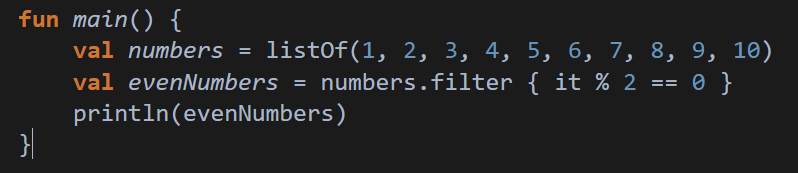
fun main() {

val numbers = listOf(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

val evenNumbers = numbers.filter { it % 2 == 0 }

println(evenNumbers)

}

****