

Pregunta 1

- Explicar en un párrafo, con sus propias palabras, el principio de pruebas: “La ausencia de errores es una falacia”

Destaca que, aunque se detecten y corrijan errores en un sistema, esto no garantiza que el software cumplirá completamente con las necesidades o expectativas del cliente. Tampoco asegura que esté libre de defectos. Las pruebas son una herramienta para reducir el riesgo de problemas, pero no pueden eliminar por completo la posibilidad de que existan fallas o de que surjan situaciones no previstas durante su uso.

- Explicar en un párrafo, con sus propias palabras, el principio de pruebas: “Las pruebas sirven para demostrar defectos”

Resalta que el objetivo principal de las pruebas es identificar errores o problemas en el sistema. Esto significa que, al realizar pruebas, se busca evidenciar fallas en lugar de confirmar que el sistema está completamente libre de ellas. Siempre existe la posibilidad de que persistan defectos en áreas no evaluadas o en condiciones no contempladas durante las pruebas.

- Explicar en un párrafo, con sus propias palabras, el principio de pruebas: “Las pruebas exhaustivas son imposibles”

Establece que no se pueden evaluar todas las combinaciones posibles de datos, escenarios y condiciones en un sistema debido a la enorme cantidad de tiempo y recursos que esto requeriría. Por esta razón, es esencial seleccionar y priorizar los casos de prueba más relevantes y representativos para maximizar la efectividad del proceso de pruebas y enfocarse en áreas de mayor riesgo o importancia.

- Explicar en un párrafo, con sus propias palabras, el principio de pruebas: “Las pruebas deben empezar lo más pronto posible”

enfatisa la importancia de integrar las pruebas desde las primeras etapas del desarrollo de software. Esto permite identificar y abordar posibles errores en fases iniciales, evitando que se propaguen y se vuelvan más costosos de corregir. Las pruebas pueden iniciarse incluso sin código, evaluando requisitos, diseños o especificaciones, lo que ayuda a mejorar la calidad del producto desde el principio.

- Explicar en un párrafo, con sus propias palabras, el principio de pruebas: “Los defectos se aglutinan”

Señala que los errores tienden a concentrarse en ciertas áreas del sistema, en lugar de estar distribuidos de manera uniforme. Esto significa que un pequeño número de módulos o componentes suele contener la mayoría de los problemas. Identificar estas áreas críticas permite enfocar los esfuerzos de prueba en los puntos más propensos a fallas, optimizando el tiempo y los recursos

- Explicar en un párrafo, con sus propias palabras, el principio de pruebas: “La paradoja del pesticida”

establece que, si siempre se realizan las mismas pruebas en un sistema, con el tiempo estas dejarán de ser efectivas para detectar nuevos errores, ya que los defectos previamente cubiertos ya estarán corregidos. Para evitar esto, es necesario actualizar o ampliar los casos de prueba, adaptándolos a los cambios en el sistema y a las nuevas funcionalidades, asegurando así que las pruebas sigan siendo útiles y relevantes

- Explicar en un párrafo, con sus propias palabras, el principio de pruebas: “Las pruebas dependen del contexto”

indica que las estrategias y enfoques de prueba deben ajustarse a las características específicas de cada sistema. Las pruebas de un videojuego, por ejemplo, serán muy diferentes de las pruebas de un sistema bancario, ya que los objetivos, usuarios, riesgos y prioridades varían en cada caso. Por ello, es fundamental analizar el propósito del software, su entorno de uso y las necesidades de sus usuarios para diseñar pruebas adecuadas y efectivas.

Pregunta 3

Las APIs son componentes fundamentales en el desarrollo de software moderno, permitiendo la integración y comunicación entre diferentes aplicaciones y servicios. En este contexto:

- a) Explique 3 riesgos de no realizar pruebas de seguridad y desempeño en nuestra API
- b) Explique como las pruebas de contrato aumentan la fiabilidad de nuestra API
- c) Explique la importancia de las pruebas punto a punto entre APIs

Parte A)

- Exposición a vulnerabilidades de seguridad: Sin pruebas de seguridad, la API puede quedar abierta a ataques como inyecciones SQL, robo de datos, o acceso no autorizado. Esto podría comprometer datos sensibles y la reputación de la organización.
- Bajo rendimiento bajo carga: Si no se evalúa el desempeño, la API podría fallar o volverse extremadamente lenta cuando se somete a un alto volumen de solicitudes. Esto afectaría negativamente la experiencia del usuario y podría interrumpir servicios críticos.
- Comportamiento impredecible en condiciones extremas: Sin pruebas de estrés y escalabilidad, no se puede garantizar que la API funcione correctamente ante picos de uso o situaciones inesperadas, lo que puede provocar errores graves o interrupciones del sistema.

Parte B)

Las pruebas de contrato verifican que la API cumpla con las expectativas definidas en el contrato (especificaciones como formatos de datos, endpoints, tipos de respuesta, etc.). Esto asegura que los consumidores de la API siempre obtendrán respuestas consistentes y predecibles, incluso si se realizan cambios en la API. Al detectar discrepancias entre lo que la API promete y lo que realmente entrega, estas pruebas reducen la posibilidad de errores en la integración, aumentando la confianza tanto en el proveedor como en los consumidores de la API.

Parte C)

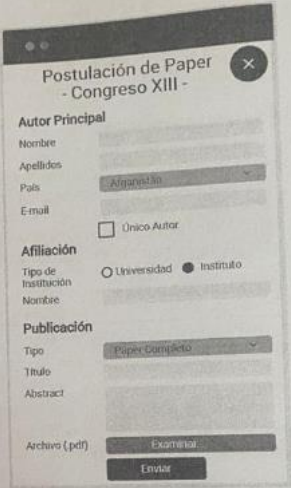
Las pruebas punto a punto aseguran que todas las interacciones entre APIs (o servicios que dependen unas de otras) funcionen como un todo, validando que los datos se envíen y reciban correctamente a lo largo de todo el flujo. Estas pruebas son cruciales para identificar problemas de integración, como incompatibilidades de formato, errores en la transferencia de datos o latencias inesperadas. Además, garantizan que las APIs trabajen juntas de manera fluida, proporcionando una experiencia robusta y confiable para los usuarios finales.

Pregunta 4

Se tiene una aplicación para postular una investigación (Paper) en el Congreso XII. Para poder entrar a la postulación, se debe utilizar la pantalla siguiente:

Los campos deben cumplir lo siguiente:

- Nombre y Apellidos de Autor admite cada uno una cadena de mínimo 2 caracteres y máximo 50 caracteres. Recibe solo caracteres alfabéticos, espacios en blanco y guiones.
- País es una lista desplegable con 195 países. Por defecto está el primer valor de la lista.
- El e-mail debe cumplir el formato de un correo electrónico válido (Sugerencia: pueden usar expresiones regulares o una regla descrita)
- Único Autor es opcional. Pero si se deja en blanco, cargará una nueva pantalla donde ingresar a los otros autores.
- Tipo de Institución tiene "Instituto" seleccionado por defecto.
- Nombre de Institución es una cadena de mínimo 6 caracteres hasta 30 caracteres. Se admiten caracteres alfanuméricos y espacios en blanco.
- El Tipo de Publicación es "Paper Completo" o "Short Paper". Según la selección, la postulación se enviará a diferentes comités.
- Título de Publicación es una cadena de mínimo 10 caracteres hasta 40 caracteres. Se admiten caracteres alfanuméricos y espacios en blanco.
- Abstract de Publicación es una cadena de mínimo 80 caracteres hasta 300 caracteres. Se admiten caracteres alfanuméricos, espacios en blanco, guiones y símbolos matemáticos.
- Archivo de Publicación recibe sólo archivos en formato PDF con un tamaño máximo de 30 MB.



Se pide para esta pantalla:

- Arma la tabla de clases equivalentes
- Listar 1 caso de prueba valido y 2 casos de prueba no válidos, puedes usar la tabla de la parte "a" para identificar las clases de equivalencia para cada caso.

Condición de Entrada	Clases Válidas	Clases No Válidas
Nombre y Apellidos de Autor	- Cadena alfabética de 2 a 50 caracteres - Permite espacios y guiones (e.g., "Juan Pérez", "Ana-López")	- Menos de 2 caracteres - Más de 50 caracteres - Incluye caracteres no alfabéticos (e.g., "Juan@123")
País	- Selección de cualquier país de la lista (195 opciones)	- Valor no perteneciente a la lista - Campo vacío
E-mail	- Formato válido (e.g., "autor@dominio.com")	- Formato incorrecto (e.g., "autor@dominio", "autor@@dominio.com") - Más de 100 caracteres
Único Autor	- Marcado o sin marcar (opcional)	- N/A (es opcional)
Tipo de Institución	- "Universidad" o "Instituto" (preseleccionado por defecto "Instituto")	- Selección inválida (e.g., "Empresa", "Escuela") - Campo vacío
Nombre de Institución	- Cadena alfanumérica de 6 a 30 caracteres (e.g., "Universidad Nacional", "Inst. Superior")	- Menos de 6 caracteres - Más de 30 caracteres - Contiene caracteres no válidos (e.g., "%Universidad&")

Tipo de Publicación	- "Paper Completo" o "Short Paper"	- Selección inválida (e.g., "Resumen", "Preliminar") - Campo vacío
Título de Publicación	- Cadena alfanumérica de 10 a 40 caracteres (e.g., "Estudio del ADN en Perú")	- Menos de 10 caracteres - Más de 40 caracteres - Incluye caracteres no válidos (e.g., "@Estudio_ADN")
Abstract de Publicación	- Cadena alfanumérica de 80 a 300 caracteres (e.g., "Este estudio analiza el impacto del cambio climático...")	- Menos de 80 caracteres - Más de 300 caracteres - Incluye caracteres no válidos excesivos (e.g., "!!! Resumen !!!")
Archivo de Publicación	- Archivo en formato PDF con tamaño máximo de 30 MB	- Archivo en formato no PDF (e.g., Word, Excel) - Archivo mayor a 30 MB - Campo vacío

Prueba Valido

Objetivo	Verificar que el formulario de postulación se registre correctamente cuando se ingresan datos válidos.
Precondición	El sistema debe estar operativo y permitir la postulación de un Paper. El usuario debe estar en la pantalla correcta para postular el Paper.
Datos de Entrada	Nombre y Apellidos de Autor: "María López" País: "Perú" E-mail: "maria.lopez@universidad.com" Único Autor: Marcado Tipo de Institución: "Universidad" Nombre de Institución: "Universidad Nacional" Tipo de Publicación: "Paper Completo" Título de Publicación: "Estudio Avanzado en Matemáticas" Abstract de Publicación: "Este trabajo presenta un análisis profundo de las matemáticas aplicadas en la ingeniería moderna, abordando los conceptos teóricos y prácticos." Archivo de Publicación: "paper.pdf" (20 MB)
Resultados Esperados	El sistema debe registrar correctamente la postulación y devolver un mensaje de éxito indicando que la postulación fue completada correctamente.

Prueba No valido 1

Objetivo	Verificar que el sistema muestra los mensajes de error correspondientes cuando se ingresan datos inválidos en los campos del formulario.
Precondición	El sistema debe estar operativo y permitir la postulación de un Paper. El usuario debe estar en la pantalla de postulación.
Datos de Entrada	Nombre y Apellidos de Autor: "Juan" País: "País Incorrecto" (no pertenece a la lista) E-mail: "juan.dominio.com" (formato inválido) Único Autor: No marcado Tipo de Institución: "Instituto" Nombre de Institución: "ABC" Tipo de Publicación: "Paper Completo" Título de Publicación: "Matemáticas" Abstract de Publicación: "Estudio." (menos de 80 caracteres) Archivo de Publicación: "archivo.docx" (no es PDF)
Resultados Esperados	El sistema debe mostrar los siguientes mensajes de error: "El nombre del autor debe tener al menos 2 caracteres."

	"El país seleccionado no es válido." "El e-mail debe tener un formato válido." "El nombre de la institución debe tener entre 6 y 30 caracteres." "El abstract debe tener al menos 80 caracteres." "El archivo debe estar en formato PDF."
--	---

Prueba No valido 2

Objetivo	Verificar que el sistema muestra los mensajes de error cuando se ingresan datos inválidos en campos específicos.
Precondición	El sistema debe estar operativo y permitir la postulación de un Paper. El usuario debe estar en la pantalla de postulación.
Datos de Entrada	Nombre y Apellidos de Autor: "Ana-López" País: "Perú" E-mail: "ana.lopez@universidad.com" Único Autor: Marcado Tipo de Institución: "Escuela" (valor no permitido) Nombre de Institución: "Universidad Tecnológica" (válido) Tipo de Publicación: "Resumen" (valor no permitido) Título de Publicación: "Estudio Teórico del Cambio Climático en Sudamérica y sus implicaciones en el mundo académico" (más de 40 caracteres) Abstract de Publicación: "Resumen" (menos de 80 caracteres) Archivo de Publicación: "paper.pdf" (50 MB, excede el límite)
Resultados Esperados	El sistema debe mostrar los siguientes mensajes de error: "El tipo de institución seleccionado no es válido." "El tipo de publicación seleccionado no es válido." "El título de la publicación no debe exceder los 40 caracteres." "El abstract debe tener al menos 80 caracteres." "El archivo debe ser menor a 30 MB."

Pregunta 5

Se tiene la clase Main donde se han definido 6 pruebas para la clase ListaDeReproduccion como se ve en la Columna "A". Cada prueba devuelve un booleano que indica si ha sido superada. Se usa además la clase Cancion, sólo con funcionalidades limitadas a lo que se necesita en las pruebas de ListaDeReproduccion.

Usando las 2 primeras pruebas, se ha programado la clase ListaDeReproduccion como se ve en la Columna "B". Se pide:

- (1.5 puntos) Programar el método "agregarCancion" y ajustar el método "cuales" a fin de que se pase la prueba 3 (test3) sin romper las pruebas anteriores
- (1.5 puntos) Ajustar los métodos "agregarCancion" y "cuales" a fin de que se pase la prueba 4 (test4) sin romper las pruebas anteriores.
- (1.5 puntos) Ajustar los métodos "agregarCancion" y "cuantoDura" a fin de que se pase la prueba 5 (test5) sin romper las pruebas anteriores.
- (1.5 puntos) Ajustar el método "agregarCancion" y programar el método "cuantas" a fin de que se pase la prueba 6 (test6) sin romper las pruebas anteriores.

Nota: Puede usar los números de línea para indicar dónde se insertaría el nuevo código de la clase Menu y/o si se debe cambiar o retirar alguna línea del código original de la clase Menu. La sintaxis tiene que hacer sentido (no es estricta).

e) En este ejemplo, indicar que tipo de elemento (Test Driver, Test Stub, Mock Object, Clases a probar) son:

- La clase Main:
- La clase Cancion:

Columna A	Columna B
<pre> public class Main{ public static void main(String args[]) { Main pruebas = new Main(); pruebas.correrPruebas(); } public void correrPruebas() { System.out.println("Prueba 1:"+test1()); System.out.println("Prueba 2:"+test2()); System.out.println("Prueba 3:"+test3()); System.out.println("Prueba 4:"+test4()); System.out.println("Prueba 5:"+test5()); } public boolean test1() { ListaDeReproduccion mlLista=new ListaDeReproduccion(); return (mlLista.cuantoDura()==0); } public boolean test2() { ListaDeReproduccion mlLista=new ListaDeReproduccion(); String salida=mlLista.cuales(); return salida.equals("No hay canciones"); } public boolean test3() { ListaDeReproduccion mlLista=new ListaDeReproduccion(); Cancion cancion1=new Cancion("Color de Esperanza",260); mlLista.agregarCancion(cancion1); String salida=mlLista.cuales(); return salida.equals("1. Color de Esperanza"); } public boolean test4() { ListaDeReproduccion mlLista=new ListaDeReproduccion(); Cancion cancion1=new Cancion("Color de Esperanza",260); mlLista.agregarCancion(cancion1); Cancion cancion2=new Cancion("Pies Descalzos",206); mlLista.agregarCancion(cancion2); String salida=mlLista.cuales(); return salida.equals("1. Color de Esperanza, 2. Pies Descalzos"); } public boolean test5() { ListaDeReproduccion mlLista=new ListaDeReproduccion(); Cancion cancion1=new Cancion("Color de Esperanza",260); mlLista.agregarCancion(cancion1); Cancion cancion2=new Cancion("Pies Descalzos",206); mlLista.agregarCancion(cancion2); return (mlLista.cuantoDura()==466); } public boolean test6() { ListaDeReproduccion mlLista=new ListaDeReproduccion(); Cancion cancion1=new Cancion("Color de Esperanza",260); mlLista.agregarCancion(cancion1); Cancion cancion2=new Cancion("Pies Descalzos",206); mlLista.agregarCancion(cancion2); return (mlLista.cuantoDura()==526); } } class Cancion{ String titulo; float duracion; Cancion(String titulo, float duracion) { this.titulo=titulo; this.duracion=duracion; } } </pre>	<pre> 1 class ListaDeReproduccion 2 { 3 String[] canciones = new String[10]; 4 float duracion; 5 6 public float cuantoDura() { 7 return 0; 8 } 9 10 public String cuales() { 11 return "No hay canciones"; 12 } 13 } </pre>

a) Ajuste para pasar test3

Método agregarCancion:

En este punto, necesitamos implementar un método agregarCancion que permita añadir canciones al arreglo canciones y mantener su duración.

Insertar el siguiente código en la clase ListaDeReproduccion después de la línea 12:

```
public void agregarCancion(Cancion cancion) {  
    for (int i = 0; i < canciones.length; i++) {  
        if (canciones[i] == null) {  
            canciones[i] = cancion.titulo;  
            duracion += cancion.duracion;  
            break;  
        }  
    }  
}
```

Ajuste al método cuales:

Modificar el método cuales para devolver los títulos de las canciones añadidas, separadas por comas:

```
public String cuales() {  
    String resultado = "";  
    for (int i = 0; i < canciones.length; i++) {  
        if (canciones[i] != null) {  
            if (!resultado.isEmpty()) {  
                resultado += ", ";  
            }  
            resultado += (i + 1) + ". " + canciones[i];  
        }  
    }  
    return resultado.isEmpty() ? "No hay canciones" : resultado;  
}
```

b) Ajuste para pasar test4

El método agregarCancion ya es adecuado para esta prueba. Solo debemos asegurarnos de que el método cuales devuelve correctamente la lista numerada de canciones. Esto ya se cumple con las modificaciones previas.

c) Ajuste para pasar test5

Método cuantoDura:

Modificar este método para que devuelva la duración total de las canciones en la lista:

```
public float cuantoDura() {  
    return duracion;  
}
```

d) Ajuste para pasar test6

Método cuantas:

Añadir este nuevo método para contar cuántas canciones hay en la lista:

```
public int cuantas() {  
    int contador = 0;  
    for (int i = 0; i < canciones.length; i++) {  
        if (canciones[i] != null) {  
            contador++;  
        }  
    }  
    return contador;  
}
```

e) Clasificación de elementos:

La clase Main: Test Driver (ejecuta pruebas sobre la clase bajo prueba).

La clase Cancion: Stub (sirve como dependencia básica para las pruebas)