DESARROLLO FORMAL DE SOFTWARE

FORMATO DE LA SEGUNDA ENTREGA

CONSTRUCCIÓN, CONTRATOS Y VERIFICACIÓN DE FUNCIONALIDADES

Instrucciones. Diligencie en Word. Escriba explicaciones concretas y use fragmentos de código breves con referencia al archivo o línea. El código completo se adjunta o se comparte mediante enlace.

1. Identificación del proyecto

Nombre del proyecto	
Programa académico	Ingeniería de Sistemas
Asignatura	Desarrollo Formal de Software
Grupo / semestre	IS06D1 / V
Docente	Yowanna Karina Caicedo Guerrero

Integrantes del grupo (máximo 3)

No.	Nombre Completo	Código estudiantil

2. Continuidad con la primera entrega

Elemento	Respuesta / evidencia
Descripción breve del sistema	
Problema o necesidad que aborda	
Usuarios o actores principales	
Cambios desde la primera entrega (si los hubo)	

3. Funcionalidades seleccionadas

Registre las tres funcionalidades que serán trabajadas durante el proyecto.

N.º	Funcionalidad de la primera entrega	¿Se mantiene? ¿Qué se implementó?	Propósito / resultado esperado
1			
2			
3			

3.1 Funcionalidad 1

Elemento	Respuesta / evidencia
Nombre y propósito	
Actor/usuario	
Entradas y tipos de datos	
Resultado esperado	

A. Contrato PRE/POST

Elemento	Especificación	Cómo se refleja en el código	Archivo/línea
PRE: condiciones previas			
POST: resultado esperado			
Aserción o comprobación, si aplica			

B. Código relevante

Aspecto	Fragmento o referencia	Explicación
Implementación principal		
Validación de PRE/entradas		
Resultado o comprobación de POST		
Condicional, si existe		
Ciclo, si existe		
Función/método o algoritmo, si existe		

C. Verificación razonada

No.	Condición verificada	Cómo se comprueba	Resultado esperado	Conclusión
1				
2				
3				

D. Ejecuciones

Caso	Entrada / estado inicial	Resultado esperado	Resultado observado	¿Coinciden? Explique
Válido/normal				
No válido o de borde				
Adicional, si corresponde				

Captura de ejecución válida:	[PEGAR CAPTURA AQUÍ]
Captura de ejecución no válida o de borde:	[PEGAR CAPTURA AQUÍ]

E. Análisis de estructuras y correctitud

Concepto	Explicación basada en el código	Referencia/evidencia	Si no aplica, justifique
Secuencia			
Condicional y ramas			
Ciclo: inicialización, condición, actualización y salida			
Invariante del ciclo			
Terminación / progreso hacia la salida			
Correctitud parcial			
Correctitud total			
Función/método y contrato			
Algoritmo y propósito			
Recursividad: caso base/recursivo			
Objetos: estado, contrato e invariante			

F. Problemas y correcciones

Elemento	Respuesta / evidencia
Diferencia o defecto encontrado	
Corrección realizada, si aplica	
Nueva ejecución y resultado	
Si no hubo diferencias, evidencia que lo respalda	

G. Conclusión de esta funcionalidad

Elemento	Respuesta / evidencia
¿Qué quedó verificado?	
¿Qué limitación o riesgo permanece?	
¿Qué falta mejorar?	

3.2 Funcionalidad 2

Elemento	Respuesta / evidencia
Nombre y propósito	
Actor/usuario	
Entradas y tipos de datos	
Resultado esperado	

A. Contrato PRE/POST

Elemento	Especificación	Cómo se refleja en el código	Archivo/línea
PRE: condiciones previas			
POST: resultado esperado			
Aserción o comprobación, si aplica			

B. Código relevante

Aspecto	Fragmento o referencia	Explicación
Implementación principal		
Validación de PRE/entradas		
Resultado o comprobación de POST		
Condicional, si existe		
Ciclo, si existe		
Función/método o algoritmo, si existe		

C. Verificación razonada

No.	Condición verificada	Cómo se comprueba	Resultado esperado	Conclusión
1				
2				
3				

D. Ejecuciones

Caso	Entrada / estado inicial	Resultado esperado	Resultado observado	¿Coinciden? Explique
Válido/normal				
No válido o de borde				
Adicional, si corresponde				

Captura de ejecución válida:	[PEGAR CAPTURA AQUÍ]
Captura de ejecución no válida o de borde:	[PEGAR CAPTURA AQUÍ]

E. Análisis de estructuras y correctitud

Concepto	Explicación basada en el código	Referencia/evidencia	Si no aplica, justifique
Secuencia			
Condicional y ramas			
Ciclo: inicialización, condición, actualización y salida			
Invariante del ciclo			
Terminación / progreso hacia la salida			
Correctitud parcial			
Correctitud total			
Función/método y contrato			
Algoritmo y propósito			
Recursividad: caso base/recursivo			
Objetos: estado, contrato e invariante			

F. Problemas y correcciones

Elemento	Respuesta / evidencia
Diferencia o defecto encontrado	
Corrección realizada, si aplica	
Nueva ejecución y resultado	
Si no hubo diferencias, evidencia que lo respalda	

G. Conclusión de esta funcionalidad

Elemento	Respuesta / evidencia
¿Qué quedó verificado?	
¿Qué limitación o riesgo permanece?	
¿Qué falta mejorar?	

3.3 Funcionalidad 3

Elemento	Respuesta / evidencia
Nombre y propósito	
Actor/usuario	
Entradas y tipos de datos	
Resultado esperado	

A. Contrato PRE/POST

Elemento	Especificación	Cómo se refleja en el código	Archivo/línea
PRE: condiciones previas			
POST: resultado esperado			
Aserción o comprobación, si aplica			

B. Código relevante

Aspecto	Fragmento o referencia	Explicación
Implementación principal		
Validación de PRE/entradas		
Resultado o comprobación de POST		
Condicional, si existe		
Ciclo, si existe		
Función/método o algoritmo, si existe		

C. Verificación razonada

No.	Condición verificada	Cómo se comprueba	Resultado esperado	Conclusión
1				
2				
3				

D. Ejecuciones

Caso	Entrada / estado inicial	Resultado esperado	Resultado observado	¿Coinciden? Explique
Válido/normal				
No válido o de borde				
Adicional, si corresponde				

Captura de ejecución válida:	[PEGAR CAPTURA AQUÍ]
Captura de ejecución no válida o de borde:	[PEGAR CAPTURA AQUÍ]

E. Análisis de estructuras y correctitud

Concepto	Explicación basada en el código	Referencia/evidencia	Si no aplica, justifique
Secuencia			
Condicional y ramas			
Ciclo: inicialización, condición, actualización y salida			
Invariante del ciclo			
Terminación / progreso hacia la salida			
Correctitud parcial			
Correctitud total			
Función/método y contrato			
Algoritmo y propósito			
Recursividad: caso base/recursivo			
Objetos: estado, contrato e invariante			

F. Problemas y correcciones

Elemento	Respuesta / evidencia
Diferencia o defecto encontrado	
Corrección realizada, si aplica	
Nueva ejecución y resultado	
Si no hubo diferencias, evidencia que lo respalda	

G. Conclusión de esta funcionalidad

Elemento	Respuesta / evidencia
¿Qué quedó verificado?	
¿Qué limitación o riesgo permanece?	
¿Qué falta mejorar?	

4. Matriz integradora de conceptos del segundo corte

Señale la funcionalidad y evidencia donde aparece cada concepto. Si no aplica, justifique.

Concepto	Funcionalidad 1 / evidencia	Funcionalidad 2 / evidencia	Funcionalidad 3 / evidencia	Observación
Secuencia				
Condicional (obligatorio en una funcionalidad)				
Ciclo (obligatorio en otra funcionalidad)				
Invariante y terminación				
Correctitud parcial				
Correctitud total				
Funciones/métodos y contratos				
Verificación de algoritmos				
Recursividad, si aplica				
Verificación orientada a objetos, si aplica				

5. Código

Elemento	Respuesta / evidencia
Lenguaje y entorno	
Enlace al código fuente y permisos de acceso	

6. Conclusiones generales

Elemento	Respuesta / evidencia
Resultados principales de la verificación	
Defectos encontrados y correcciones	
Aspectos pendientes para el proyecto final	

7. Lista de control

- ☐ Se mantienen las tres funcionalidades o se justifican cambios.
- ☐ PRE/POST y código están relacionados.
- ☐ Hay condicional en una funcionalidad y ciclo en otra.
- ☐ Se analiza invariante/terminación y correctitud cuando corresponde.
- ☐ Se revisan funciones, algoritmos, recursividad y objetos según aplique.
- ☐ Cada funcionalidad incluye ejecuciones, capturas y comparación esperado/observado.
- ☐ Se registran correcciones o la ausencia de diferencias.
- ☐ El enlace del código funciona y el PDF es legible.

8. Información de entrega

Campo	Indicación
Modalidad	Grupal, máximo 3 integrantes.
Documento	Formato diligenciado, exportado a un único PDF.
Código	Adjunto o enlace accesible; las capturas no sustituyen el código.
Lugar	AVAS.
Fecha	10 de noviembre de 2026 hasta las 7:00am