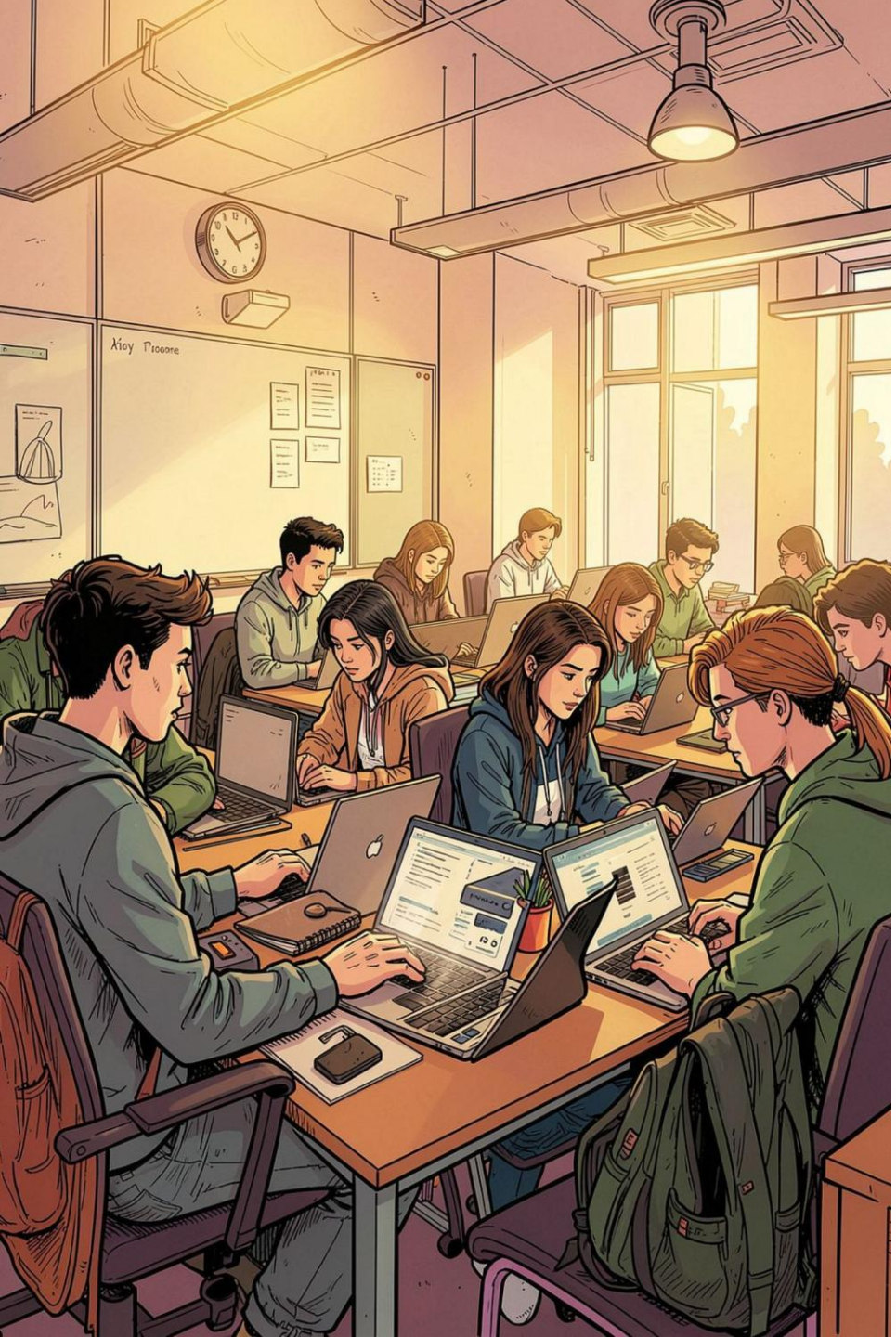


Desarrollo Formal de Software

Introducción, Articulación Curricular y Diagnóstico

INGENIERÍA DE SISTEMAS
UNIVERSIDAD DEL PACÍFICO

Mg. Yowanna Karina Caicedo Guerrero

Made with GAMMA



Las respuestas que han dado tienen elementos importantes. Desarrollo Formal de Software no significa solamente desarrollar un programa, ni tampoco significa solamente hacer verificaciones.

Se trata de construir software de manera sistemática y precisa, utilizando especificaciones formales y técnicas de verificación para establecer y comprobar que el software cumple determinadas propiedades y requisitos.

¿Dónde estamos en la línea de software?

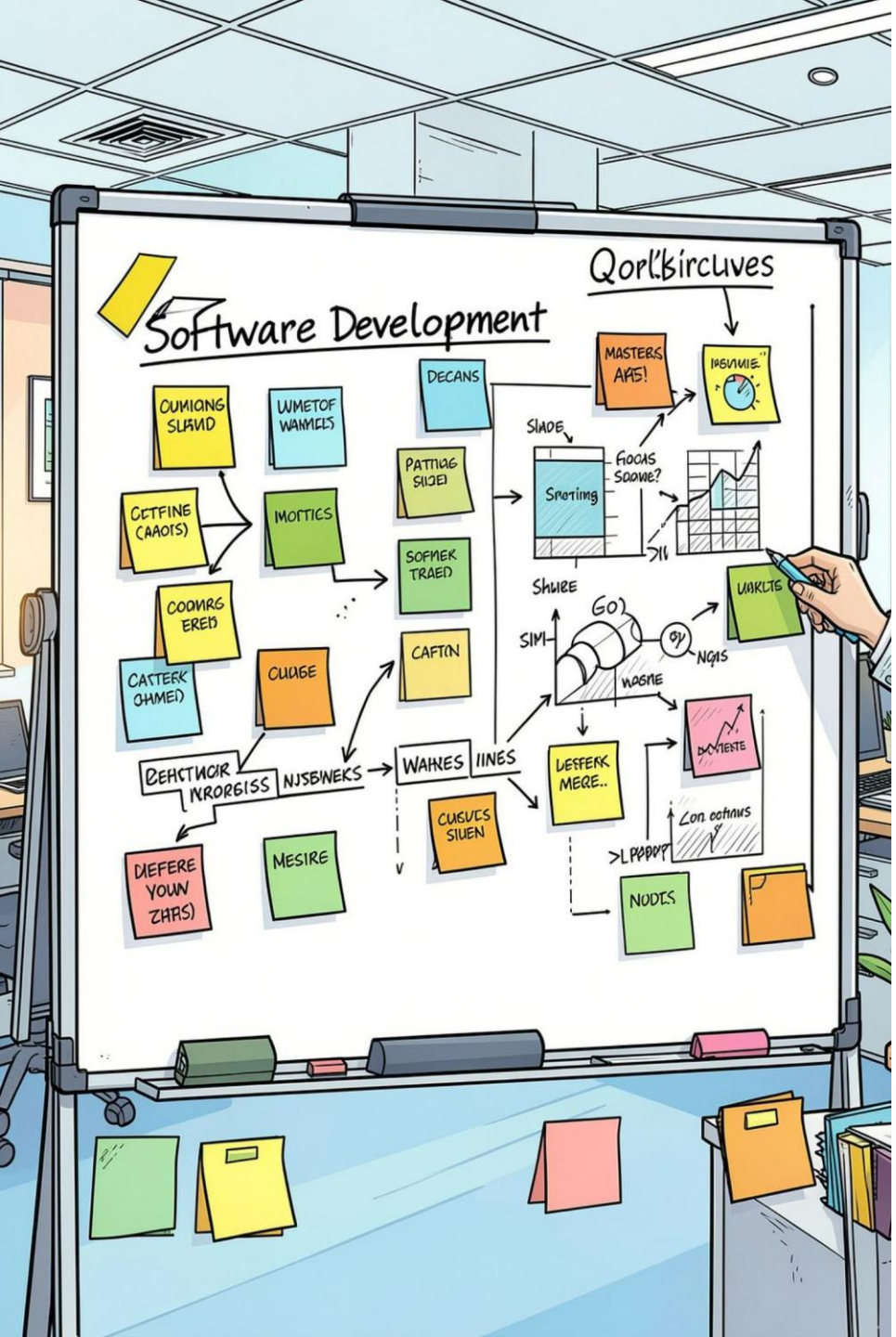
Esta asignatura ocupa un lugar central en la trayectoria curricular de Ingeniería de Sistemas.



Desarrollo Formal de Software es el puente entre la especificación teórica y la construcción verificable. Aquí se integran los conocimientos previos para producir software correcto y de calidad.

Construir el Software

Made with GAMMA



¿Qué vamos a hacer en esta asignatura?

Esta asignatura cierra el ciclo del proyecto integrador: pasar de la documentación a la construcción real del software.

1

Construir

Programar una funcionalidad pequeña y significativa del proyecto.

2

Especificar

Definir con precisión qué debe cumplir el programa.

3

Verificar

Comprobar que el software se comporta según lo especificado.

4

Analizar

Evaluar y mejorar la calidad del software construido.

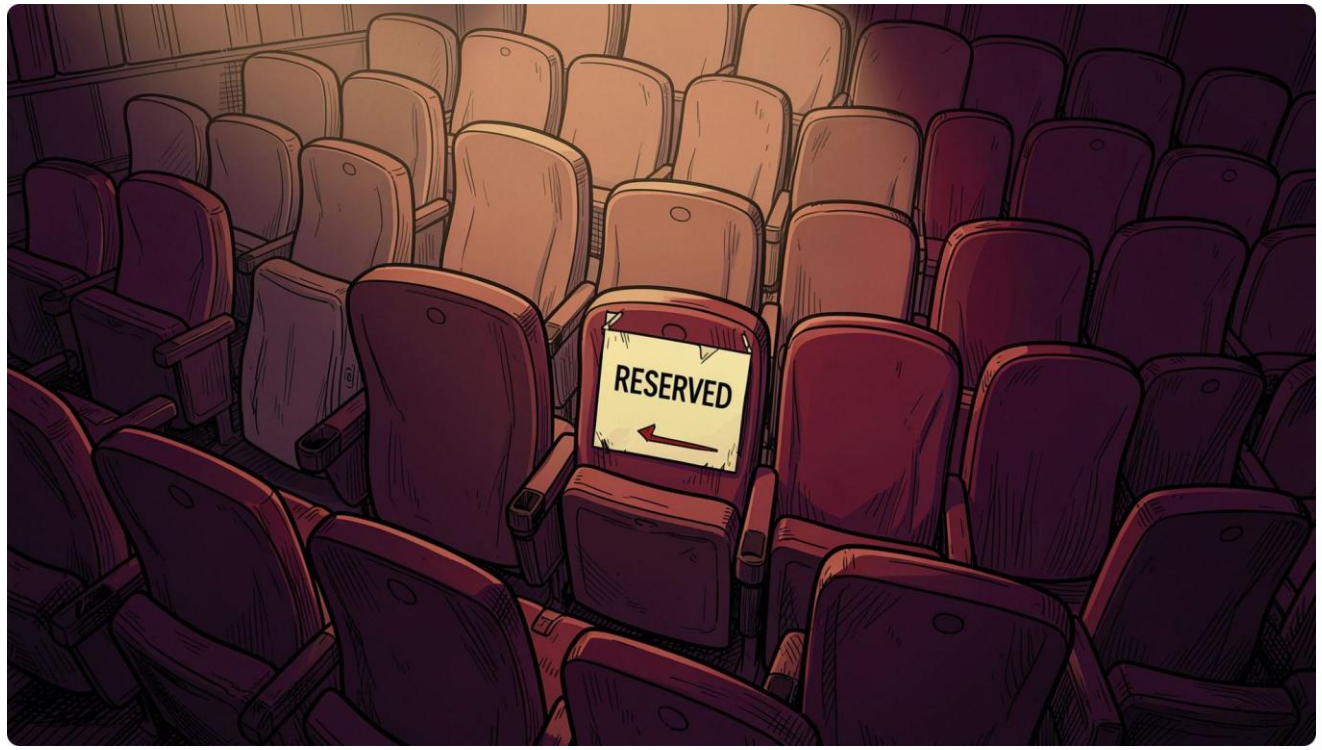
Made with GAMMA

¿Cómo sabemos que un software es correcto?



Ejemplo: Reservar un cupo

Considera la función `reservar(cupos)`. La operación solo debe permitir una reserva cuando existan cupos disponibles.



1

¿Qué debe cumplirse antes?

¿Qué condición debe ser verdadera antes de llamar a la función?

2

¿Qué debe pasar después?

¿Qué relación existe entre el estado anterior y el posterior?

3

¿Qué nunca debería ocurrir?

¿Qué propiedad debe mantenerse siempre invariable?

Made with GAMMA

Una posible solución

Estas condiciones son las **especificaciones formales** de la operación. Más adelante aprenderemos a incorporarlas al programa y verificarlas automáticamente.

Precondición

`cupos > 0`

Debe existir al menos un cupo disponible antes de reservar.

Postcondición

`cupos_nuevos = cupos_anteriores - 1`

Tras la reserva, el número de cupos se reduce exactamente en uno.

Invariante

`cupos_nuevos ≥ 0`

El número de cupos nunca puede ser negativo en ningún momento.

Made with GAMMA



Pregunta para comenzar

Antes de escribir código, reflexiona sobre cómo verificamos la corrección de un programa.

¿Basta con ejecutarlo una vez?

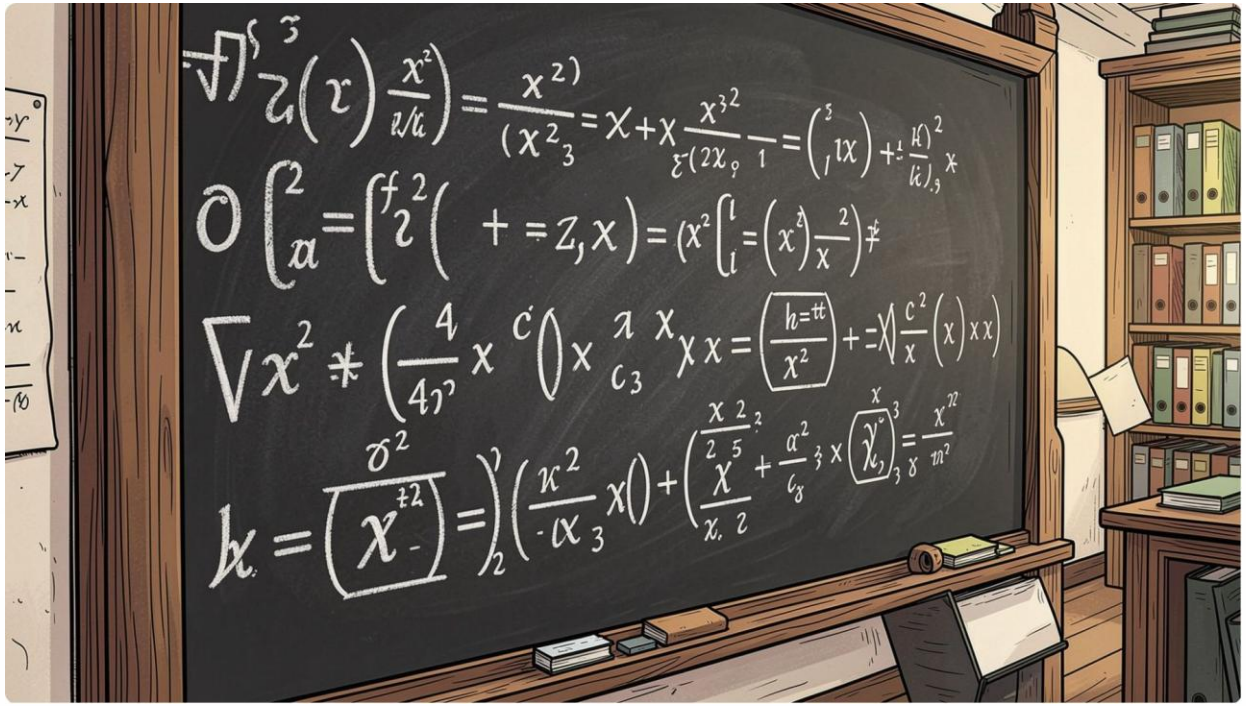
¿Basta con que no aparezca un error?

¿Basta con probar algunos casos?

¿Qué tendríais que demostrar?

Made with GAMMA

¿Qué significa «formal»?



«Formal» no significa redactar un documento con formato oficial. Significa **expresar con precisión matemática** las propiedades que el software debe cumplir.

Precondición

Qué debe cumplirse *antes* de ejecutar una operación.

Postcondición

Qué debe cumplirse *después* de ejecutarla.

Invariante

Qué condición debe mantenerse *siempre*.

Made with GAMMA

Actividad grupal: Préstamo de un Libro

Situación: un usuario solicita un libro de la biblioteca. El préstamo solo puede realizarse si el usuario está registrado y el libro está disponible.

En grupos de 3 a 5 personas, responder a las siguientes preguntas y preparar una justificación para cada respuesta.

¿Qué debe cumplirse antes?

¿Qué debe cumplirse después?

¿Qué nunca debería ocurrir?

¿Qué errores o casos límite podrían presentarse?

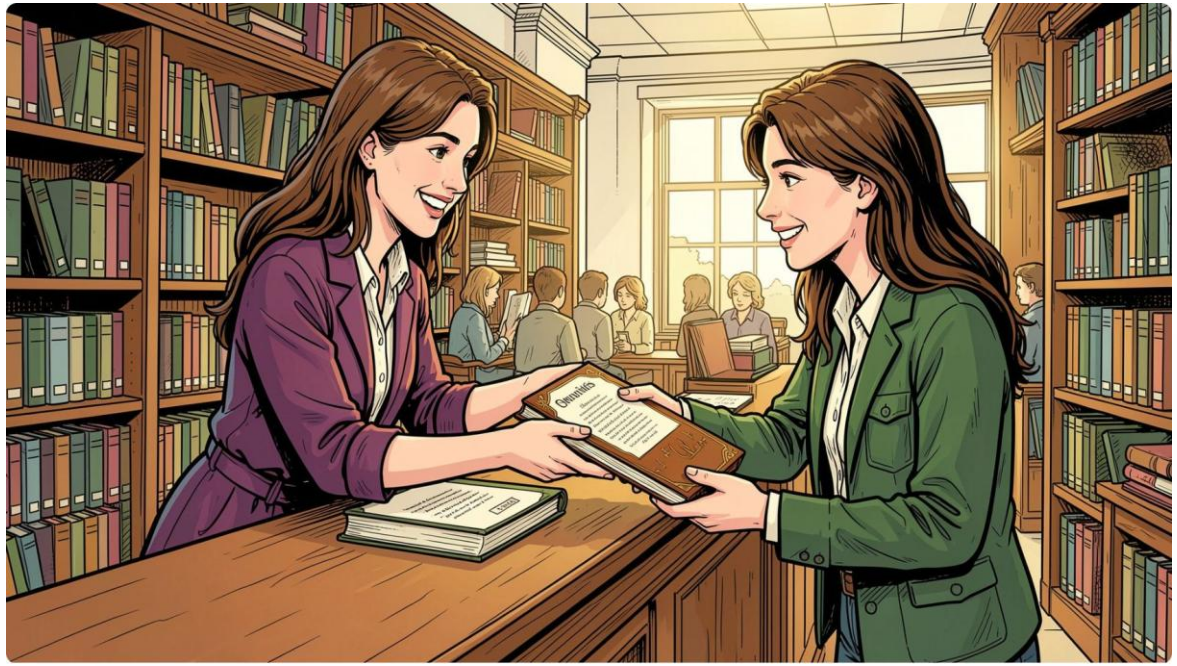


Solución orientadora de la actividad

Respuestas esperadas

- **Precondiciones:** usuario registrado; libro existente; libro disponible.
- **Postcondiciones:** préstamo registrado; libro pasa a estado «no disponible».
- **Propiedades:** no prestar el mismo ejemplar simultáneamente; la disponibilidad no puede tomar valores inválidos.
- **Casos límite:** libro no disponible; usuario no registrado; datos incompletos; solicitud duplicada.

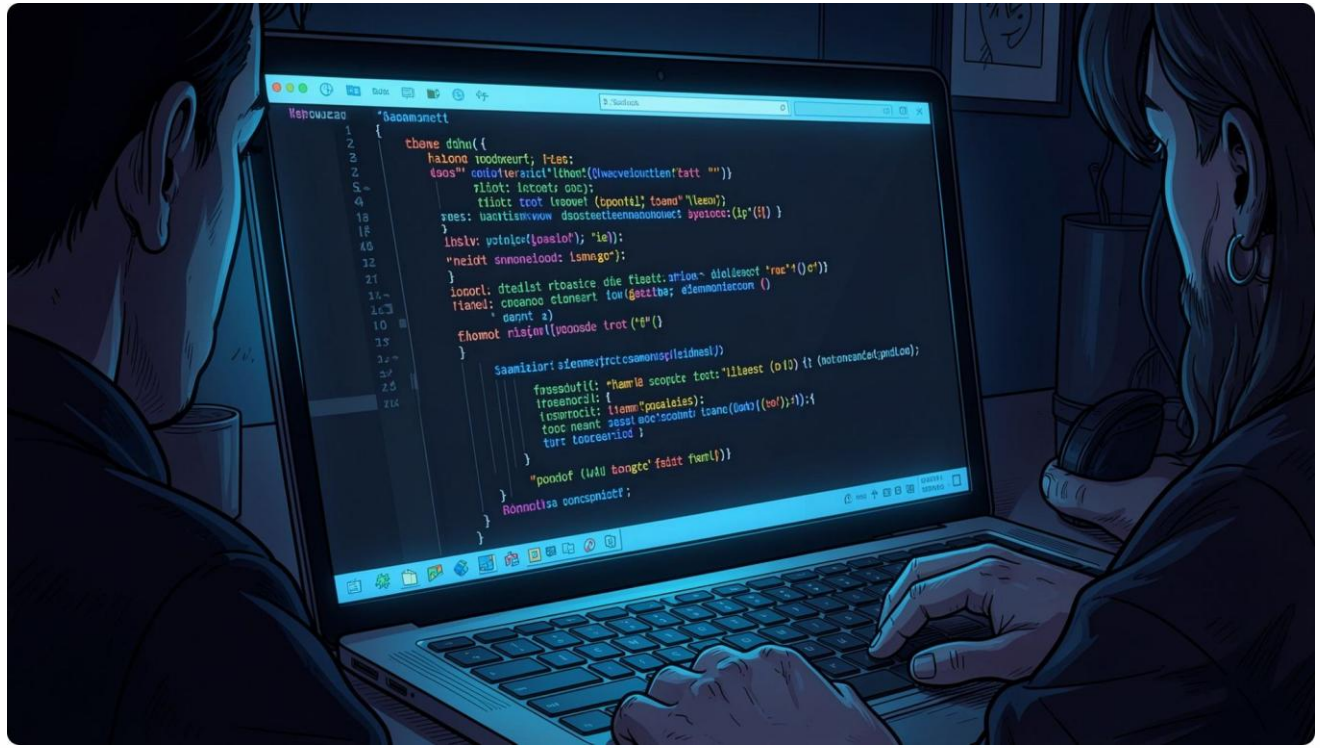
Las respuestas pueden variar; lo que se evalúa es la justificación y el razonamiento.



Made with GAMMA

Diagnóstico de Programación

No es un examen ni tiene nota. Pueden trabajar en **C++, Java o Python**.



Problema: función `reservar(cupos)`

> Si `cupos > 0`: realizar la reserva
> Disminuir el número de cupos

> Si `cupos = 0`: no realizar la reserva.

> ¿Qué debería ocurrir si `cupos < 0`?

Este diagnóstico nos permite conocer vuestro punto de partida.

Made with GAMMA

Consulta

Cada grupo traer el proyecto/documentación desarrollada en Análisis, Diseño e Ingeniería de Software, u otro de su preferencia.

- ✓ Deben identificar **2 o 3 funcionalidades** de su proyecto que puedan convertirse en un pequeño programa.

Por ejemplo:

Sistema de supletorios

•Registrar solicitud.
•Consultar solicitud.
•Validar solicitud.



- ✓ Traer para cada funcionalidad

Elemento	Respuesta
Nombre	Registrar solicitud
¿Qué hace?	…
Entradas	…
Salidas	…
¿Qué debe cumplirse antes?	…
¿Qué debe cumplirse después?	…
¿Qué no debería ocurrir?	…

- ✓ Traer lo que tengan disponible

Made with GAMMA