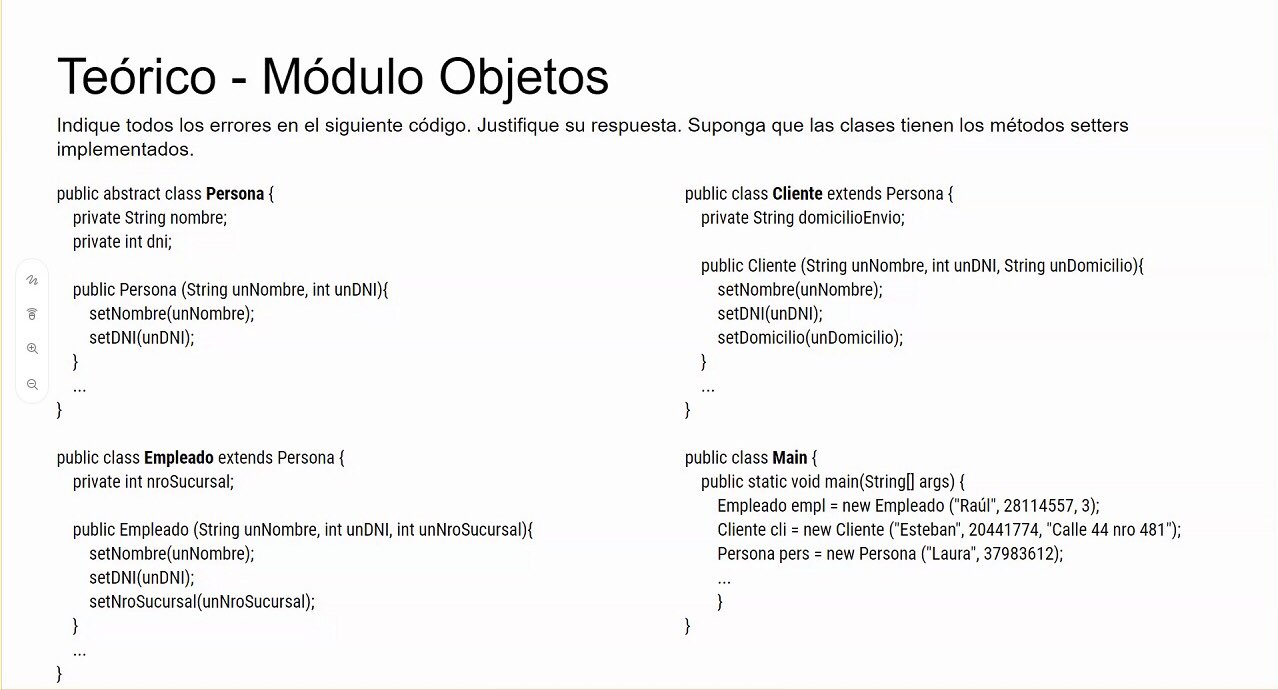
**PREGUNTAS DE FINAL TALLER:**



Programa: Compuesto por 3 clases y un programa principal

1 clase abstracta Persona (con dos atributos nombre y dni)

1 clase Empleado que extiende de Persona (con un atributo sucursal)

1 clase Cliente que extiende de Persona (con un atributo domicilio)

Errores:

1- Nunca se debe instanciar una clase abstracta (en este caso la clase persona).

2- Los atributos propios de cada clase empiezan con minúscula y al momento de invocarlos en los constructores lo hace con mayúsculas esto sería un error ya que java es un lenguaje Camel Case.

3-La clase Empleado y Cliente al momento de implementar el constructor deberían usar Super (…,…);

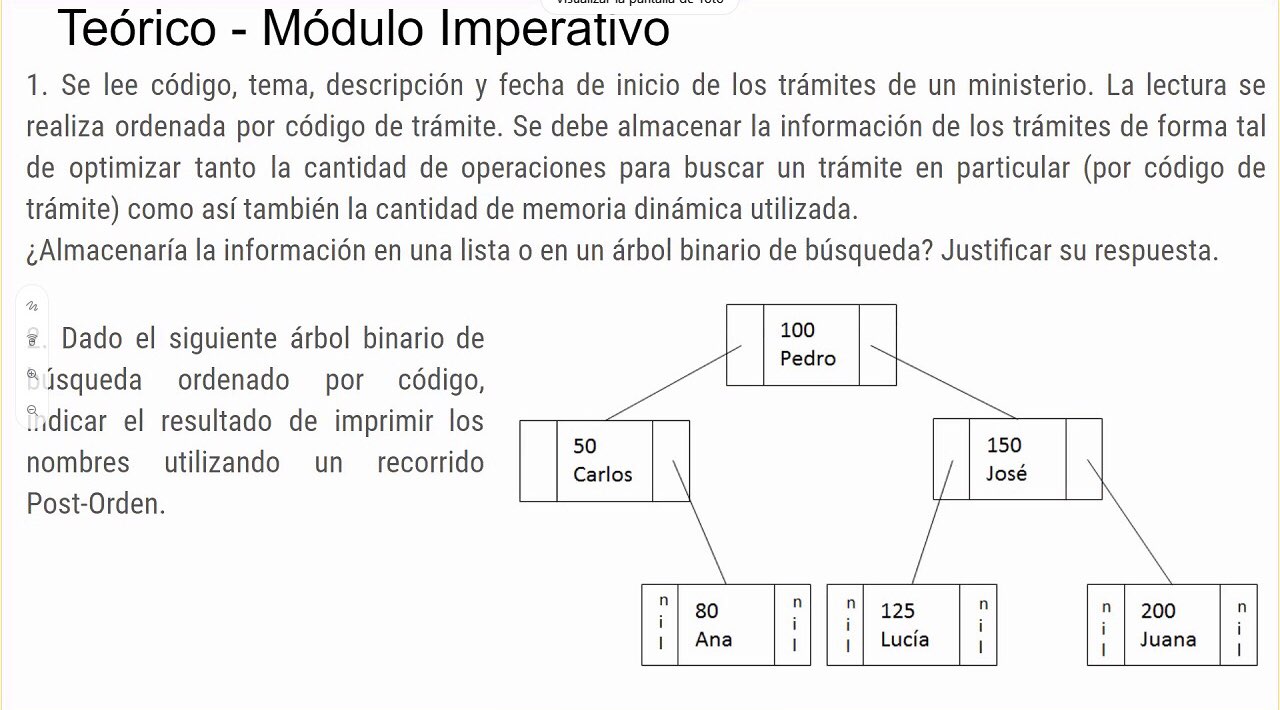
Public Nombre (x1, x2, x3) {

Super (x1, x2);

This.x3=x3;

}

// 4. set dentro del constructor , Explicar mejor.

  
ENORDEN: 50 80 100 125 150 200 (write al medio)

PRE: 100 50 80 150 125 200 (write al principio)

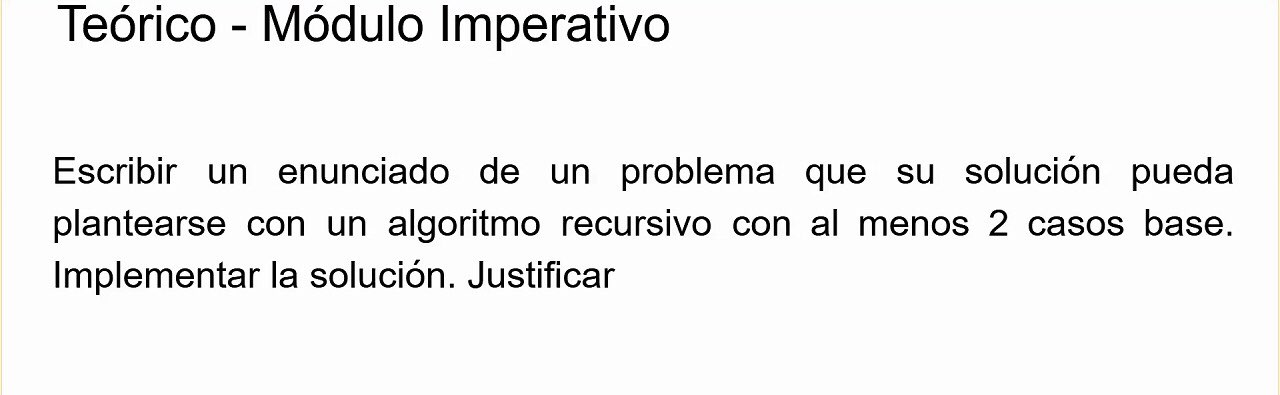
POST: 80 50 125 200 150 100(write al final)

respuesta 1:

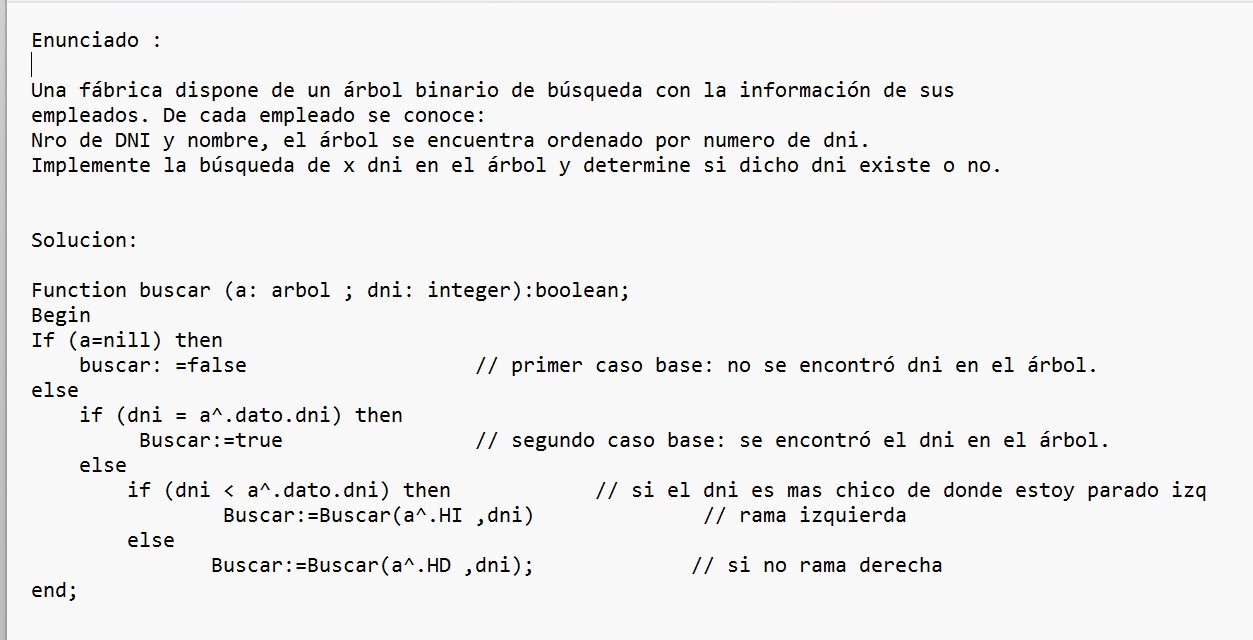
En este enunciado hay dos puntos importantes por un lado se desea optimizar la cantidad de operaciones para hacer la búsqueda y por otro lado la cantidad de memoria dinámica utilizada.

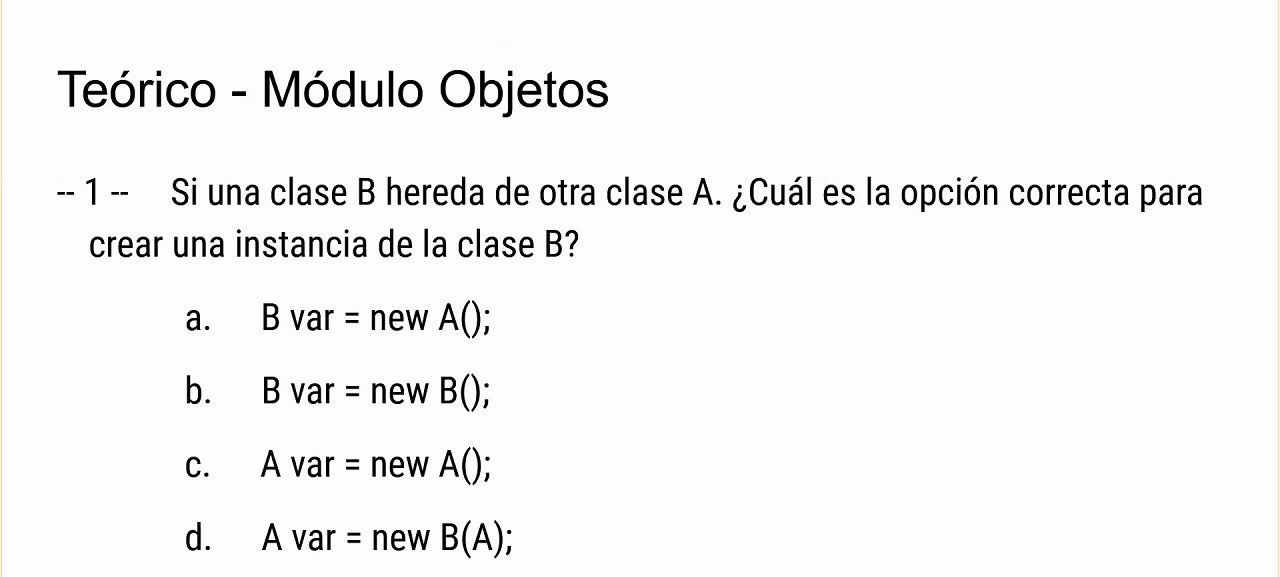
En el caso de la memoria dinámica en una lista, el proceso de búsqueda estaría alocado en memoria una única vez y en el caso del árbol el proceso de búsqueda va a estar alocada tantas veces como sea invocado el proceso de recursión, asique la utilización de la recursión siempre es menos eficiente cuando se tiene en cuanta el concepto de memoria.

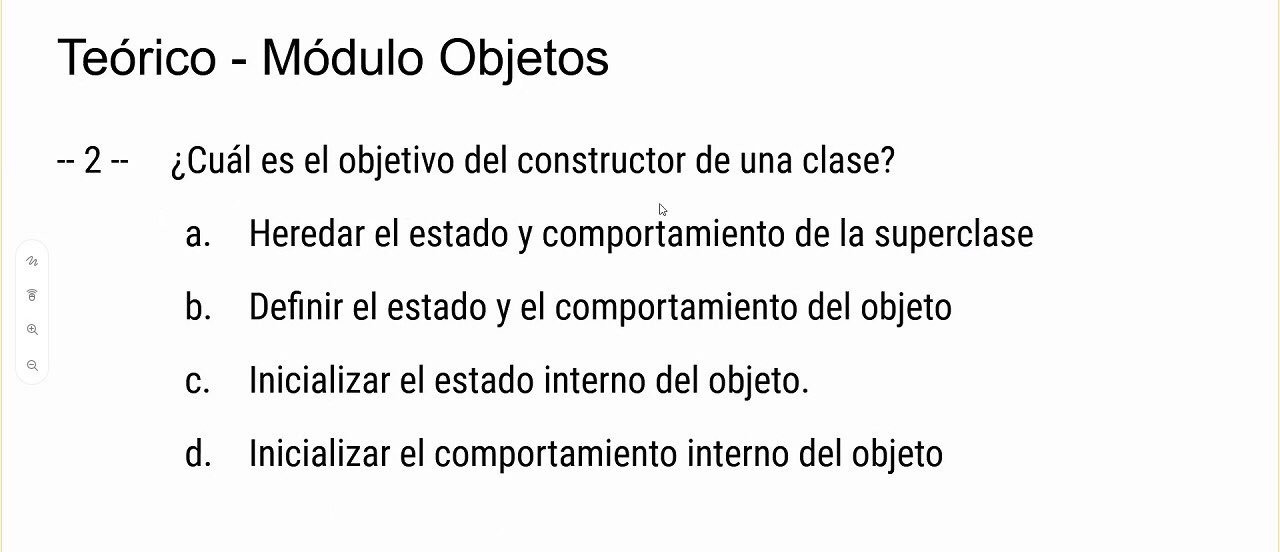
Con respecto a la cantidad de búsqueda siempre va a depender de la cantidad de elementos que tengamos ya que, si tenemos muchos elementos, el árbol de búsqueda va a ser mucho más eficiente porque podemos acortar esa búsqueda y reducir el tiempo, en el caso de la lista siempre deberíamos recorrer todos los elementos hasta llegar al elemento deseado y si fueran muchos elementos tomaría más tiempo.



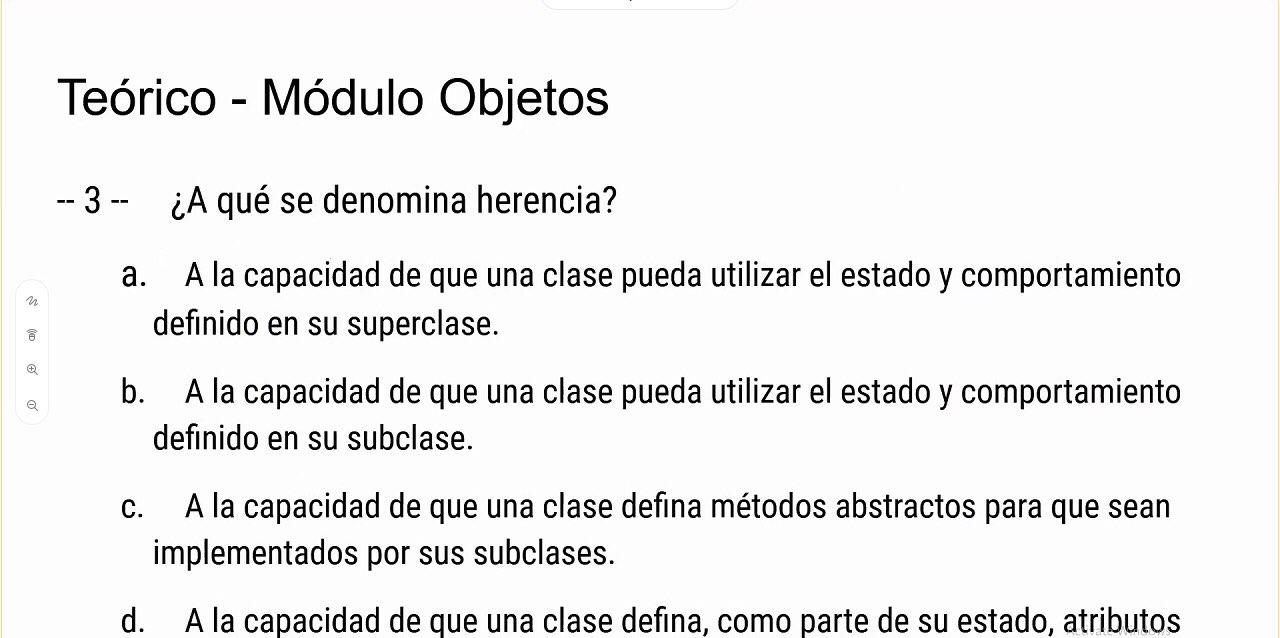
respuesta: ✓

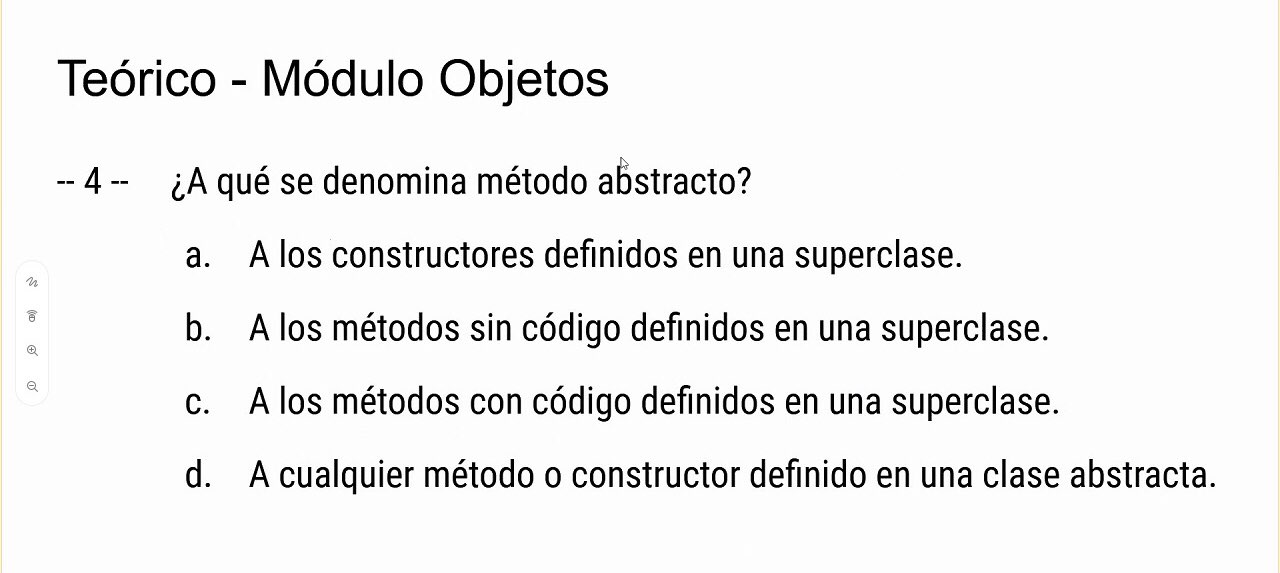


****

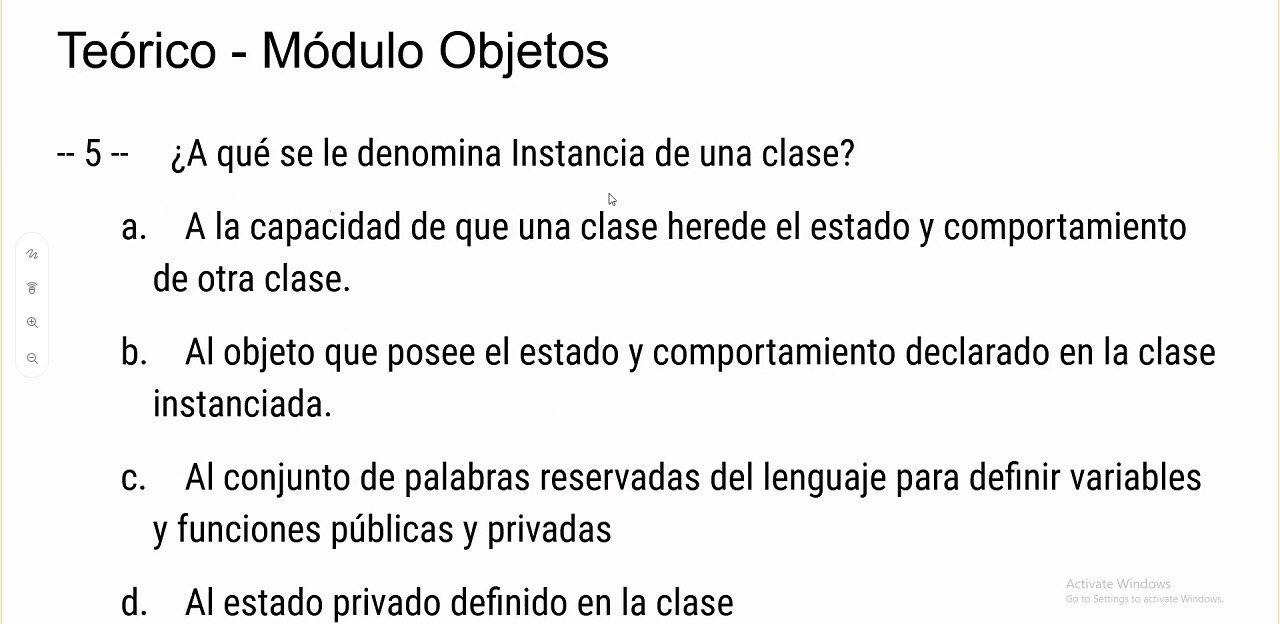
****

respuesta:

  
  
  
respuesta: ✓

A- A la capacidad de que una clase pueda utilizar el estado y comportamiento definido en su superclase.  
  
  
  
  
  
respuesta: ✓

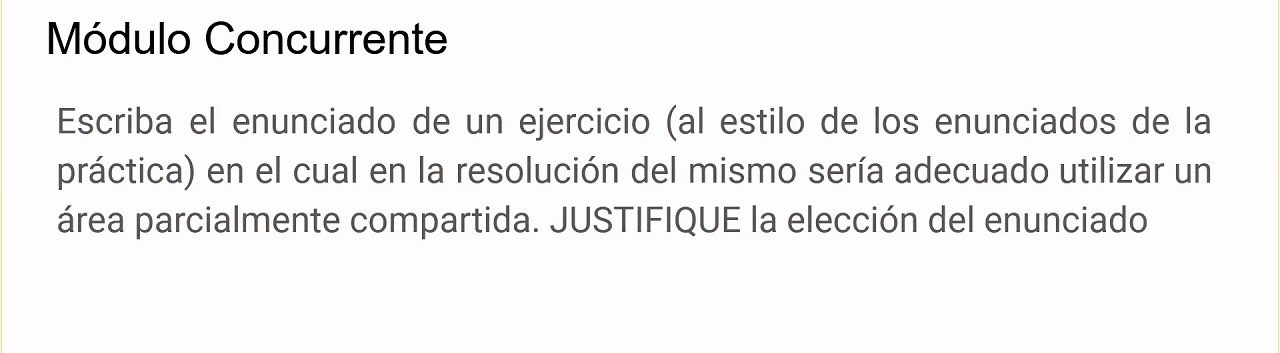
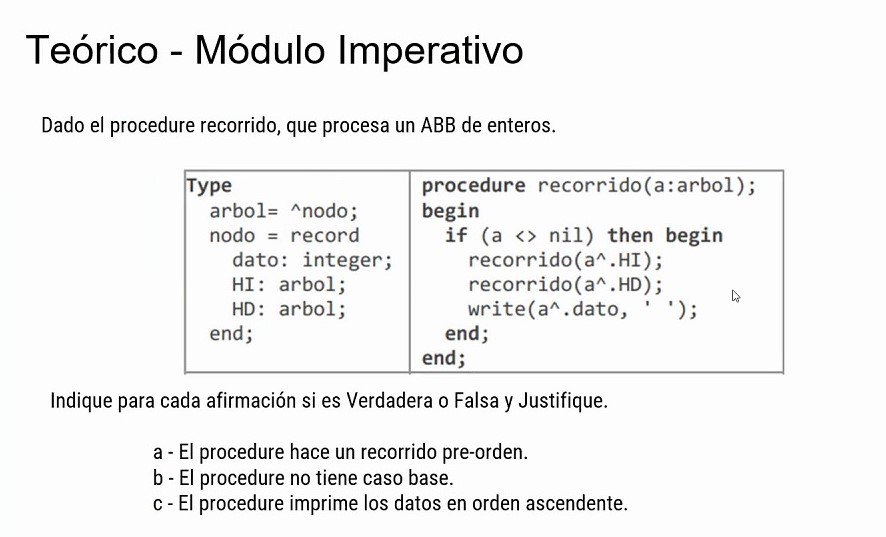
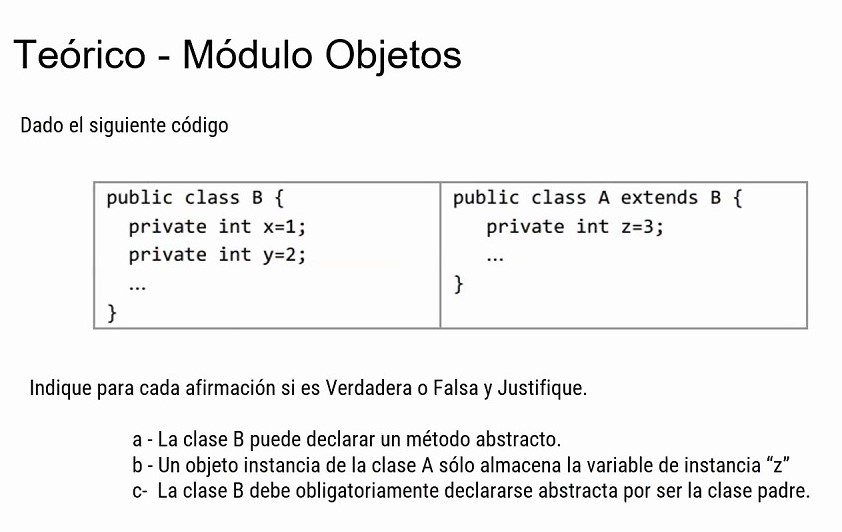
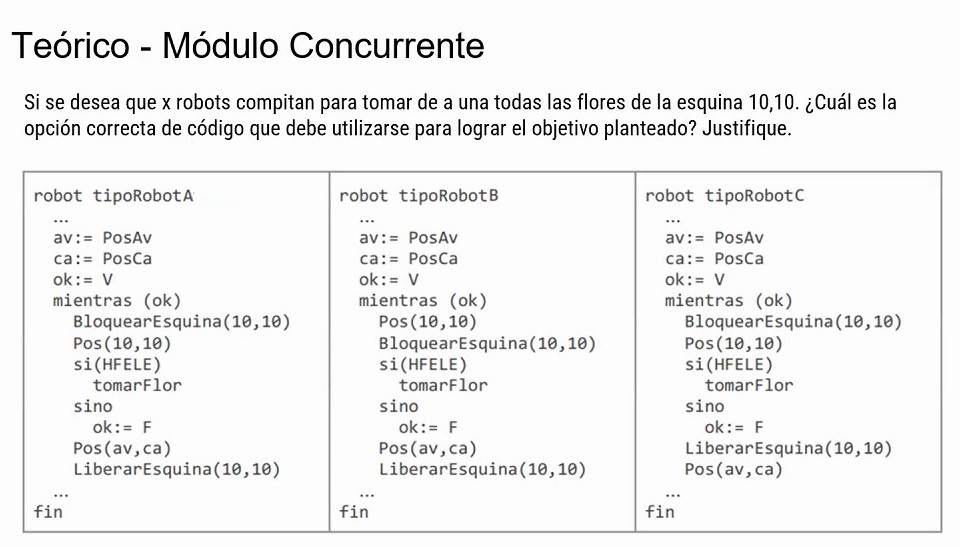
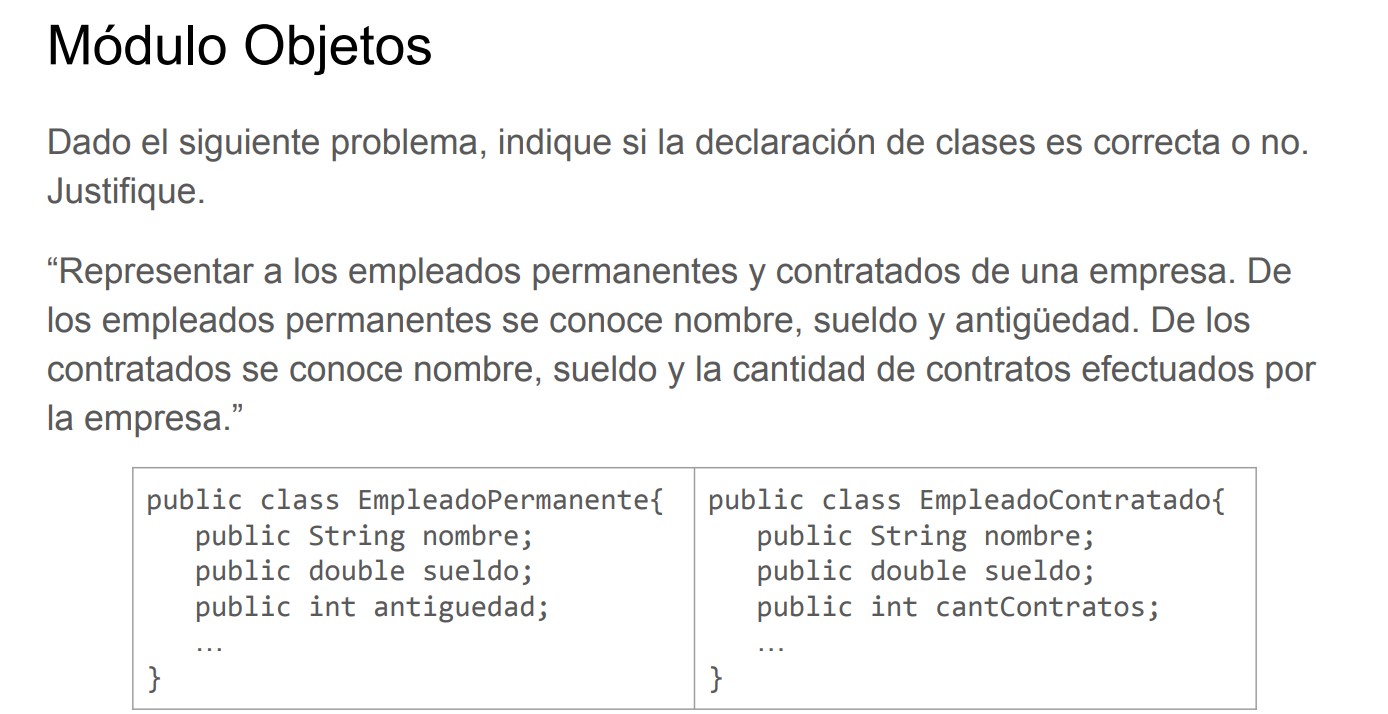
B- A los métodos sin código definidos en una superclase.

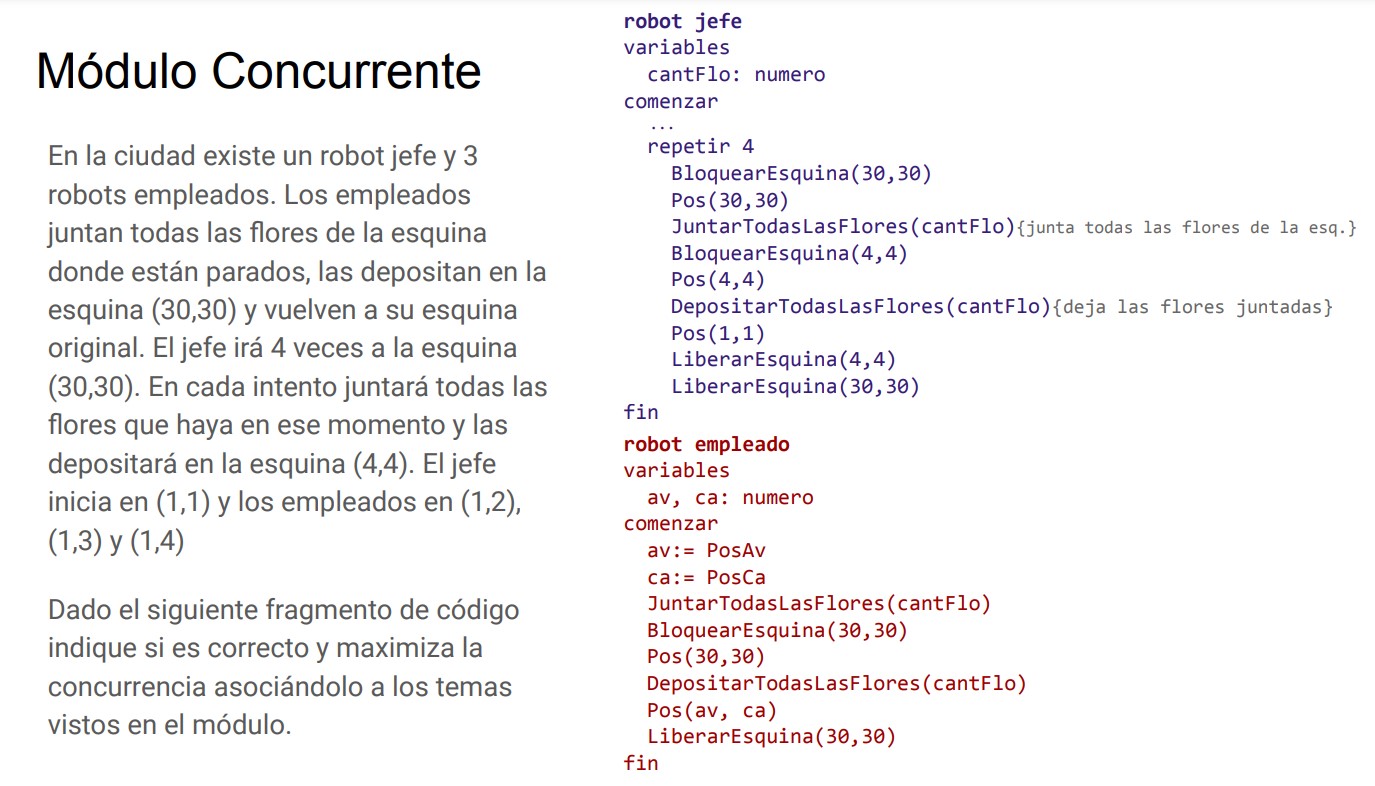
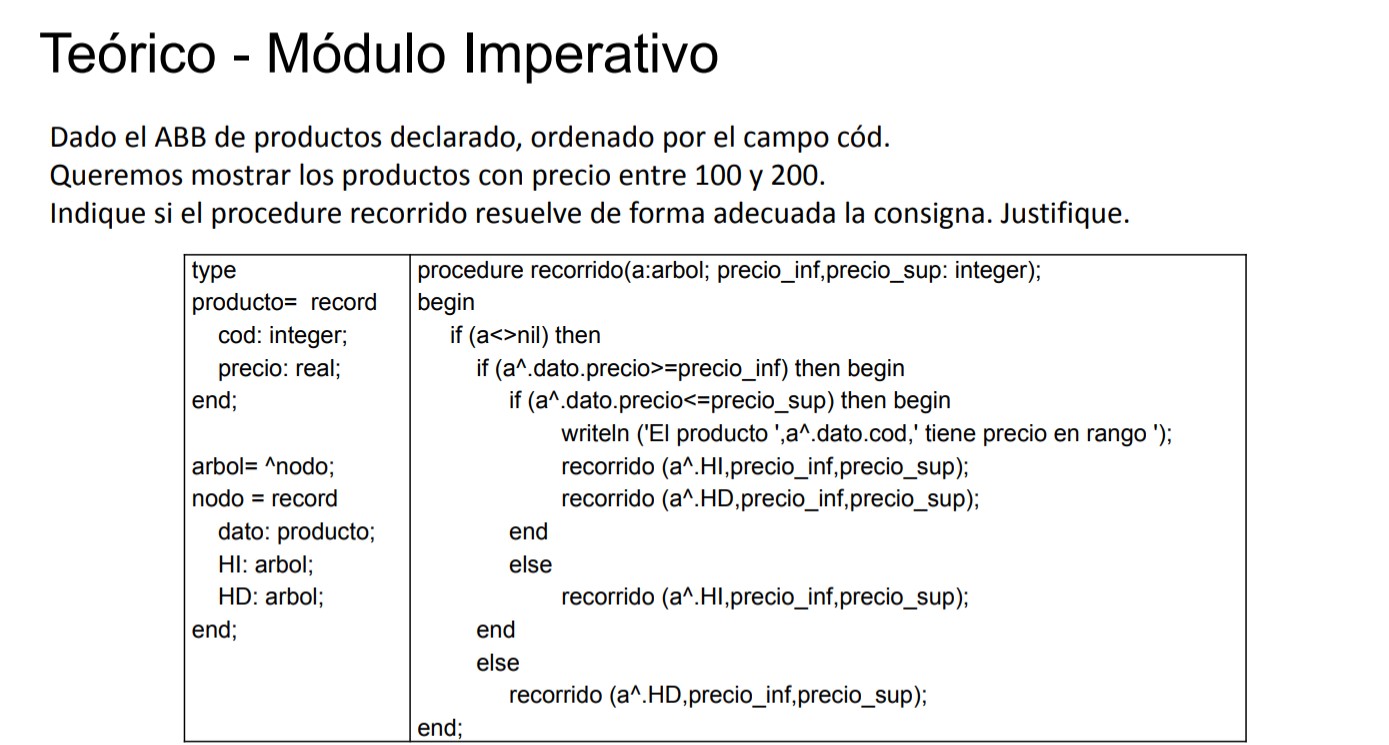


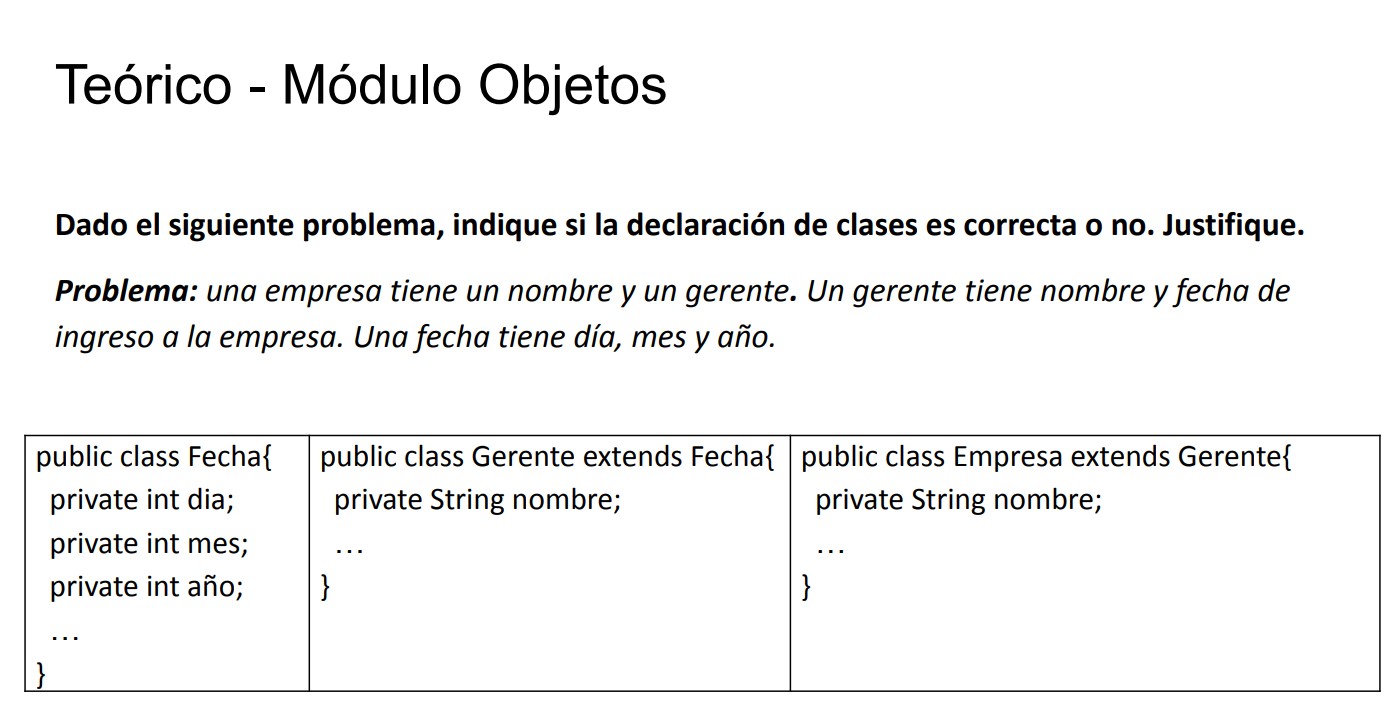
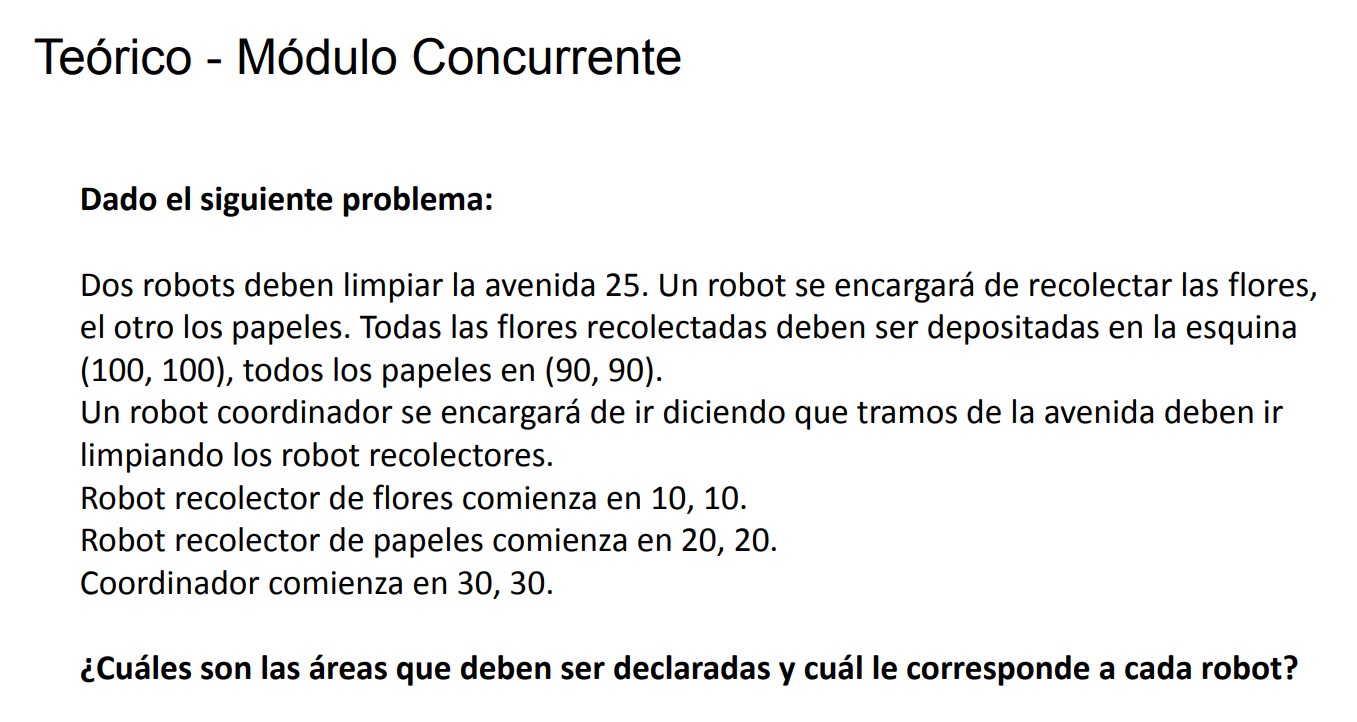
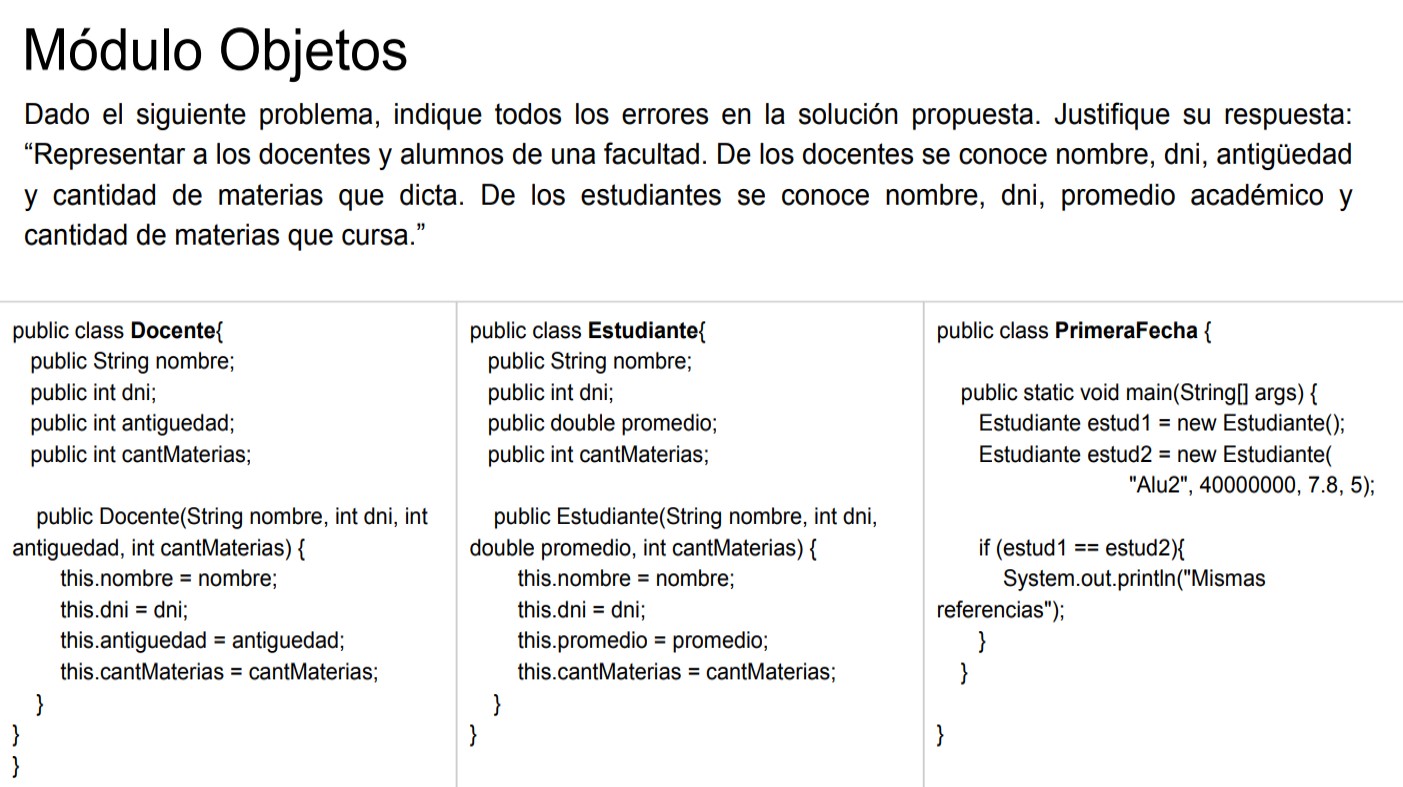
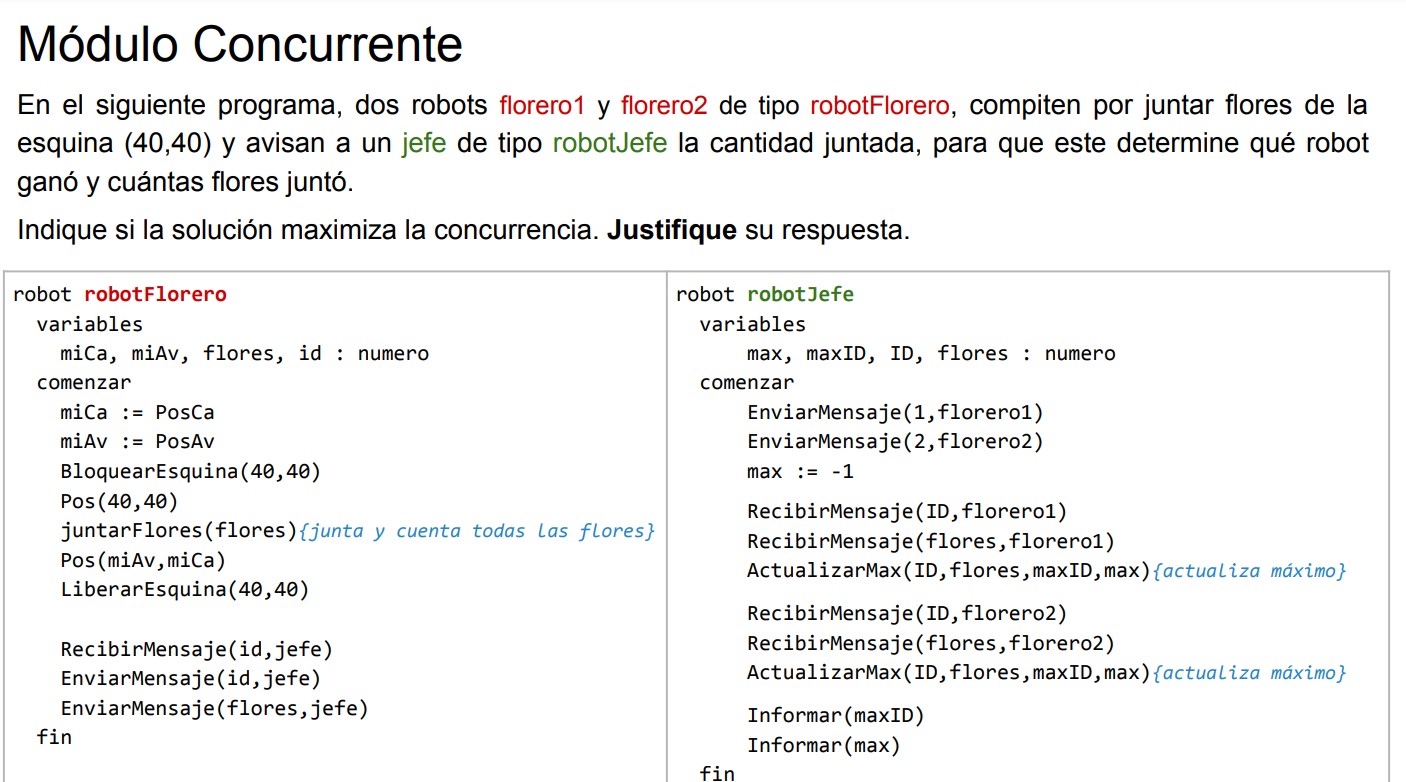
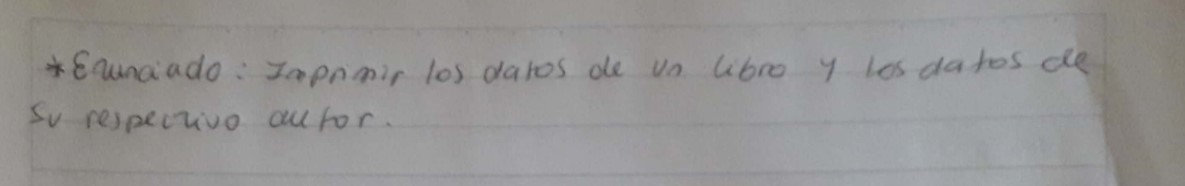
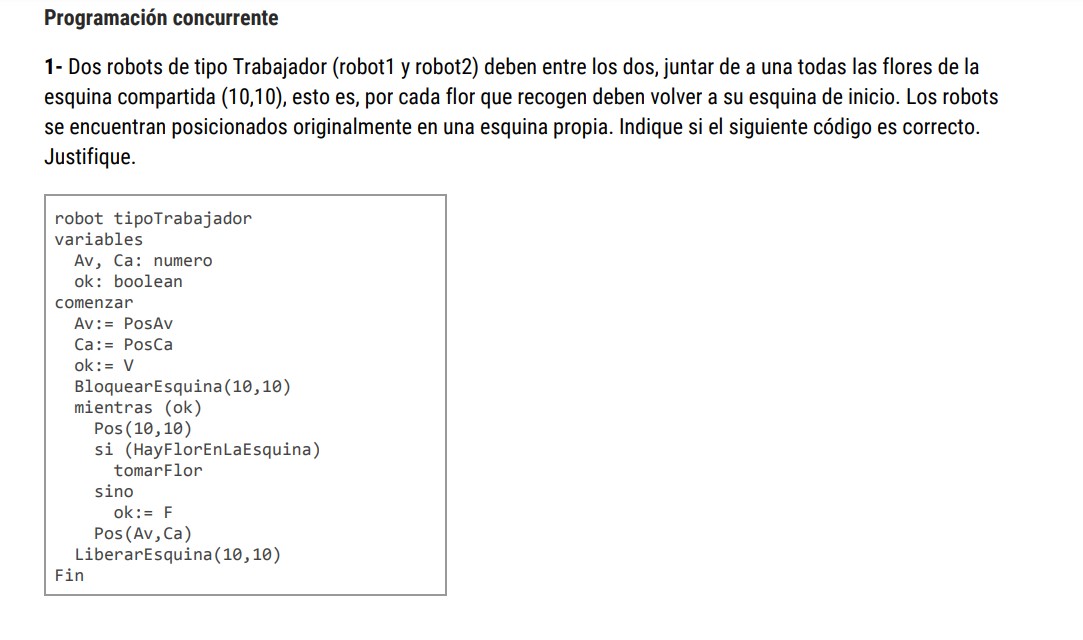
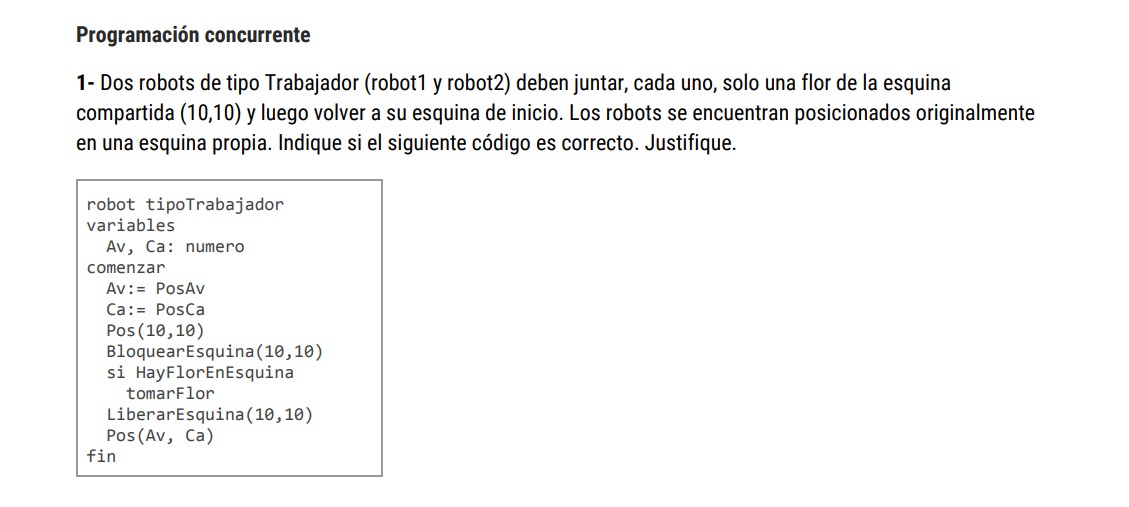
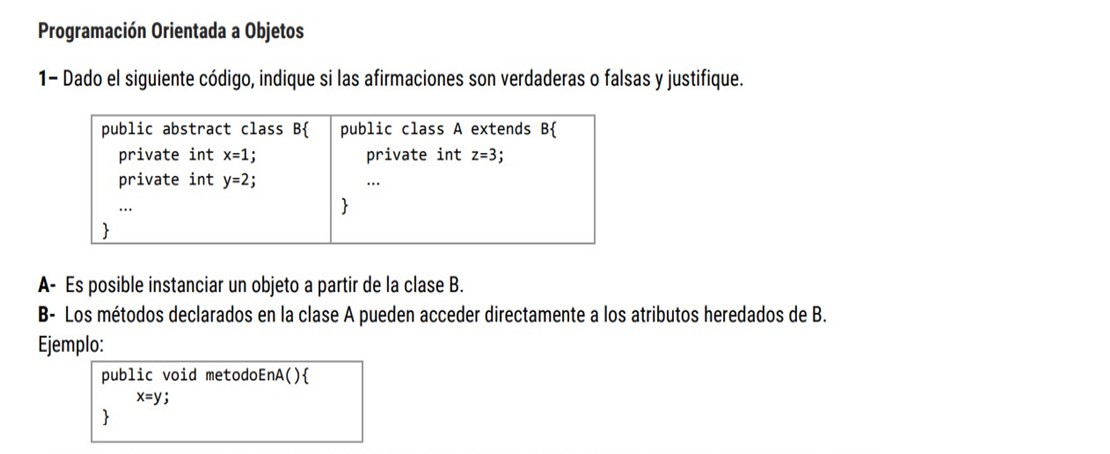
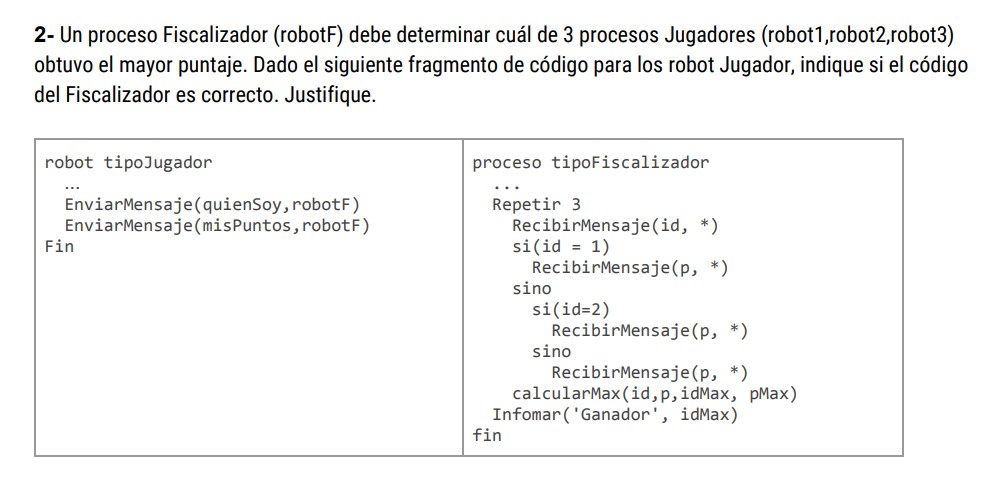
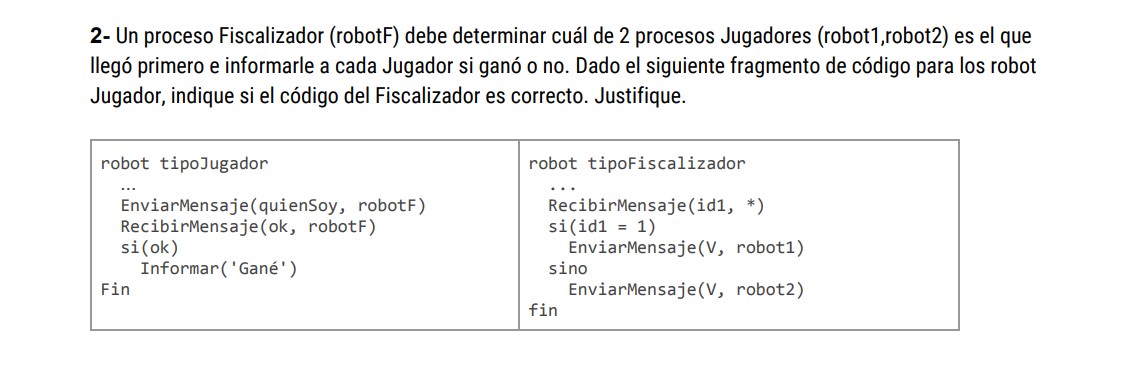
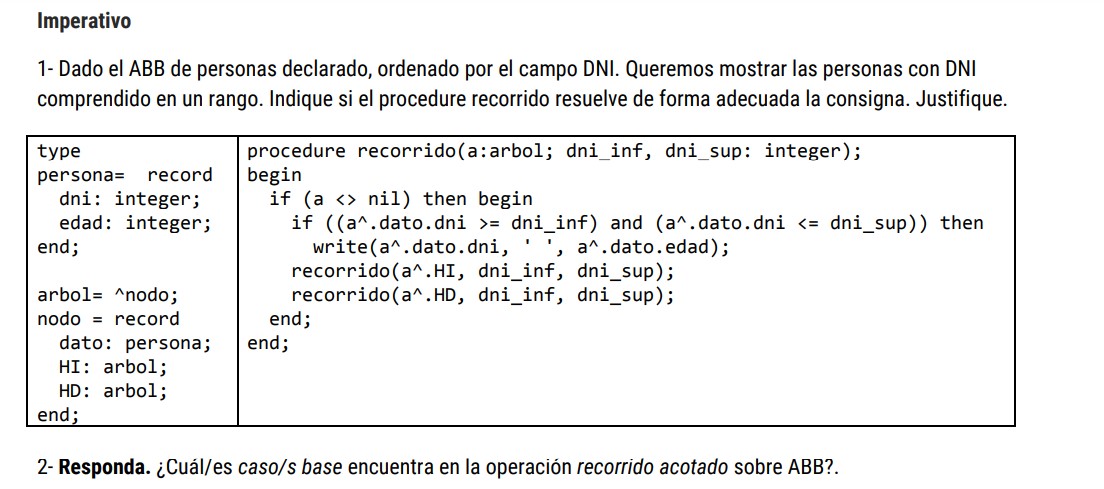
respuesta: ✓

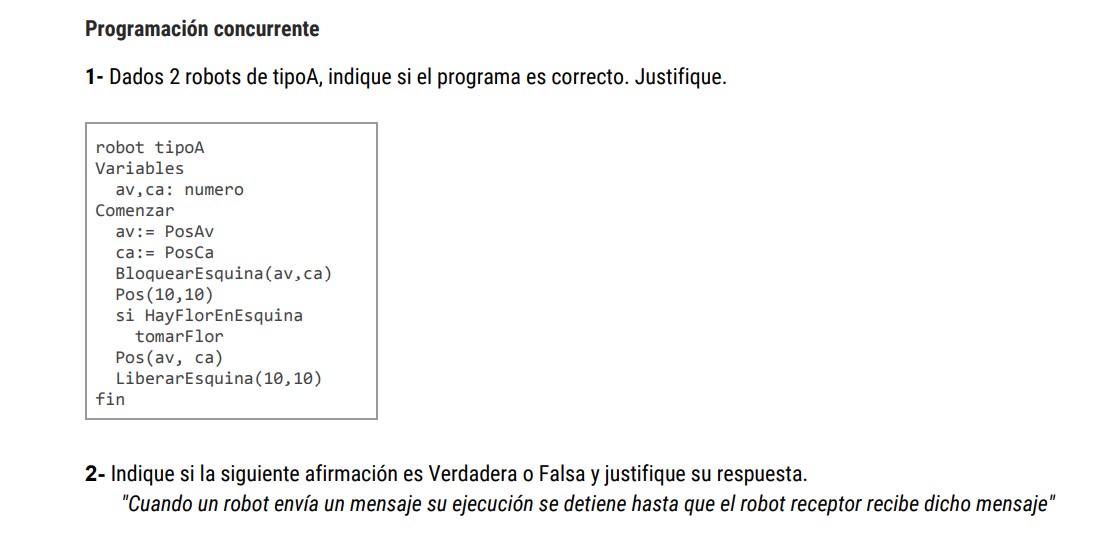
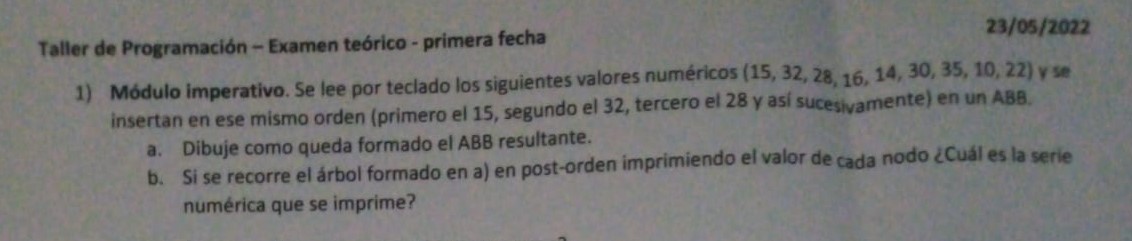
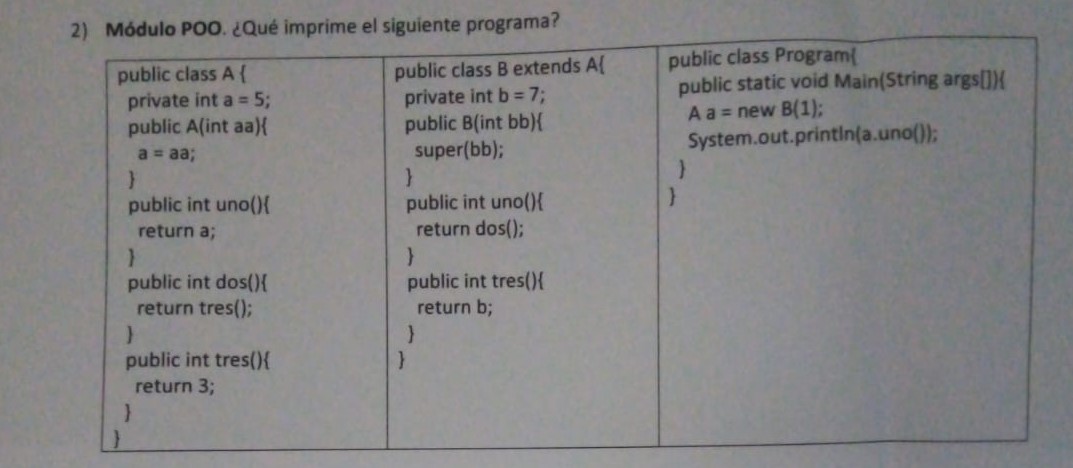
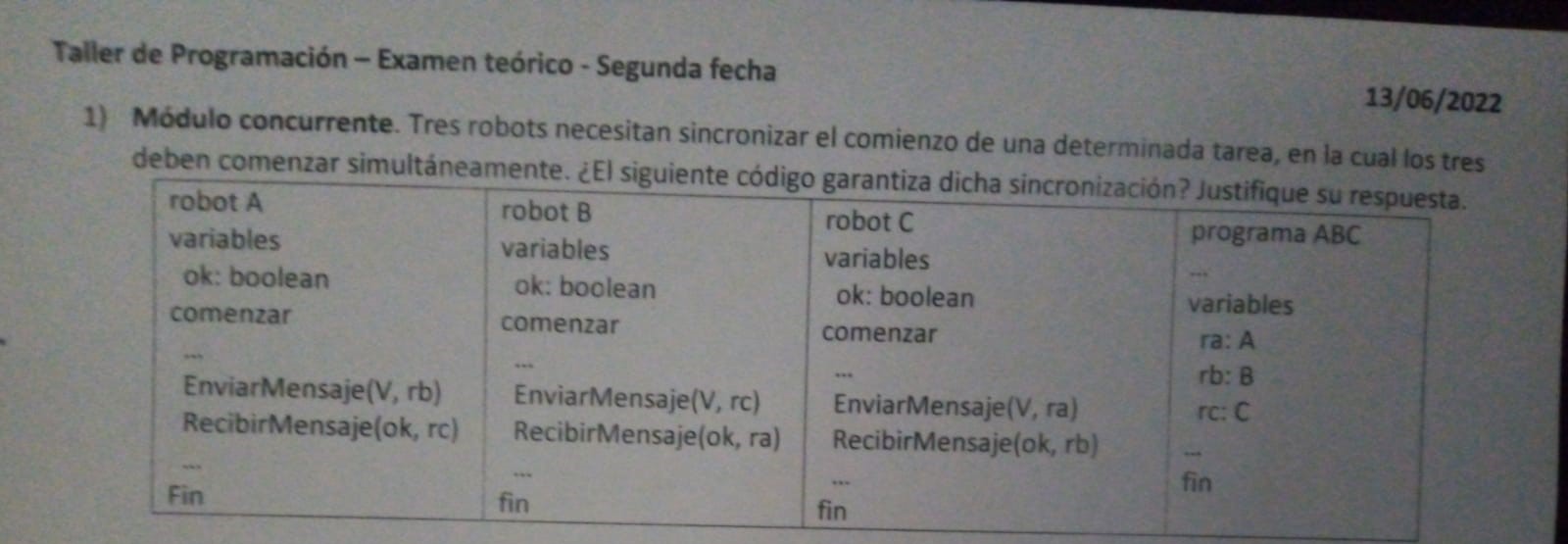
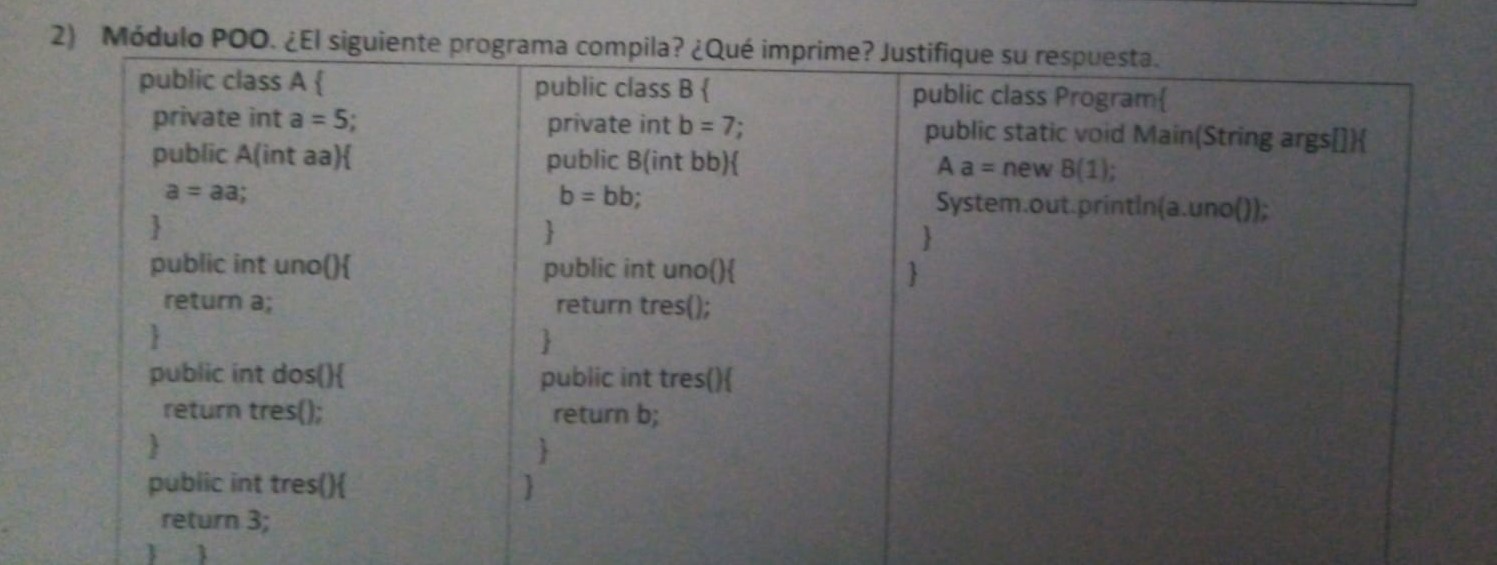
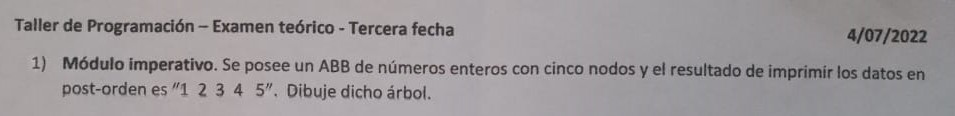
B- Al objeto que posee el estado y comportamiento declarado en la clase instanciada.

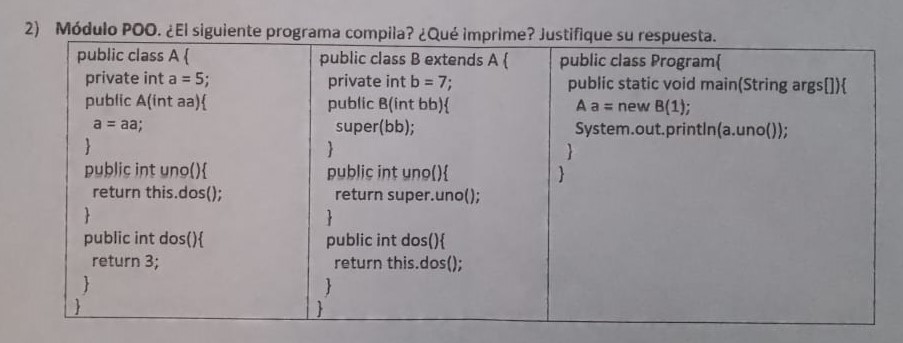
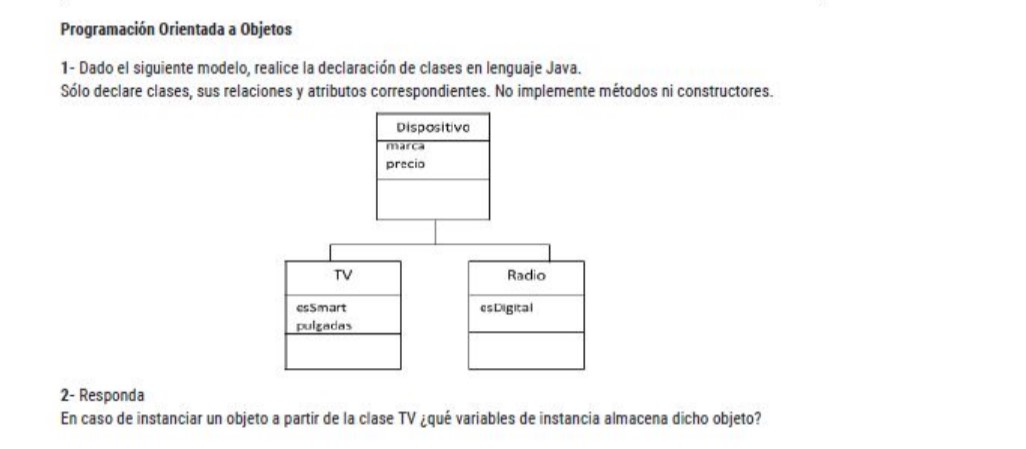
(Un objeto es siempre instancia de una clase, para que exista un objeto debe existir una clase).

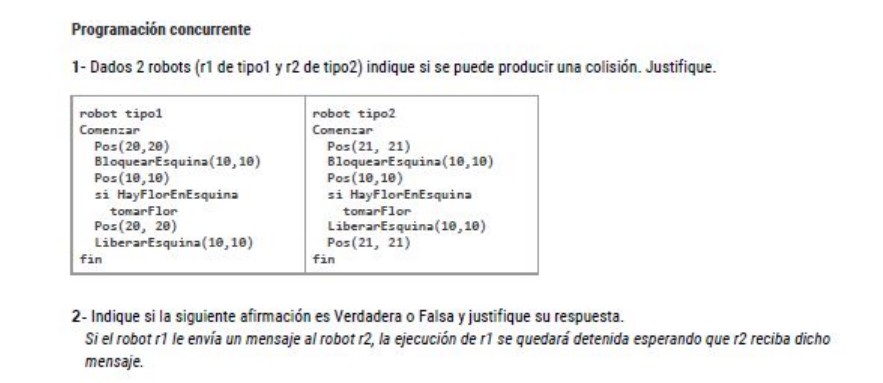
  
  
  
  
  
  
  
  
  
  
  
  


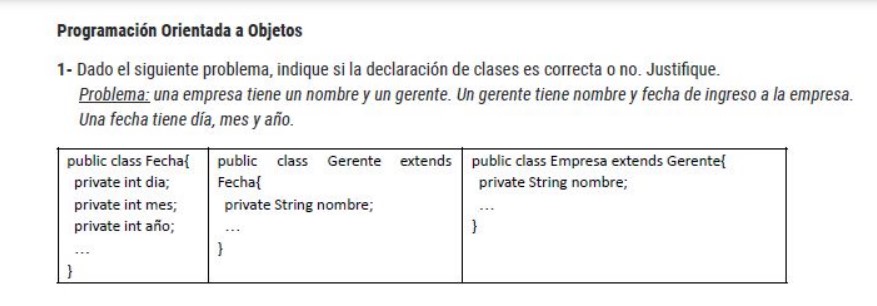
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  


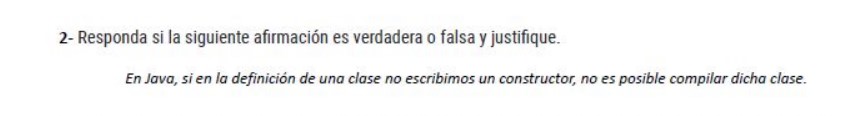
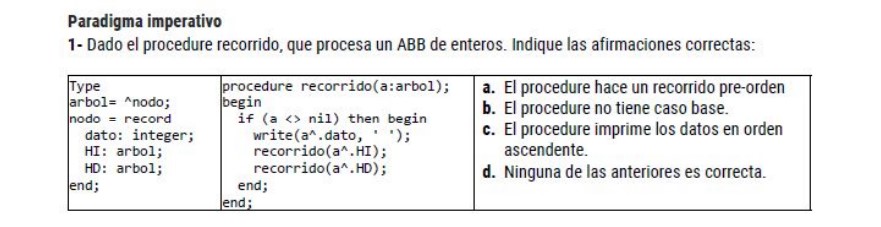
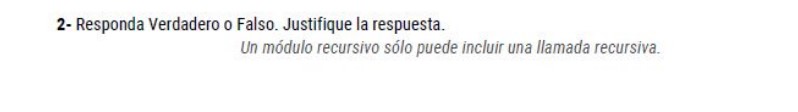
  
  
  
  
  
  
  
  
  
  
  
  
  


  
  
  
  
  
respuesta 1: ✓

public abstract class Dispositivo { // podría no ser abstracta!  
private int marca;   
private doublé precio;   
}  
  
public class TV extends Dispositivo {   
private boolean esSmart;   
private doublé pulgadas;   
}   
  
public class Radio extends Dispositivo {   
private boolean esDigital;   
}   
  
respuesta 2: ✓  
  
un objeto TV almacena: marca, precio (heredadas de la superclase), esSmart, pulgadas (v.i propias).  
  
  
  
respuesta 1: ✓

sí, puede producirse una colisión. el robot tipo 2 Libera la esquina 10,10 antes de salir de ella, y como el robot tipo 1 tiene instrucción de bloquear y acceder a la misma esquina, puede encontrarse con que está ocupada por robot tipo2. Siempre se deben Liberar el recurso compartido una vez que se sale de él.

respuesta 2: ✓  
  
Falso. el envió de mensajes es asincrónico, o sea, que el robot r1 ejecuta la instrucción de envío de mensaje y continúa ejecutando las siguientes instrucciones que tenga pendientes sin esperar a que r2 reciba el mensaje. Pasa lo contrario en caso de que se encuentre una instrucción de RecibirMensaje… en ese caso el robot detiene la ejecución hasta que le llegue el mensaje. el Recibir es sincrónico.  
  
  
  
  
respuesta 1: ✓

La declaración de clases es correcta: La clase Empresa hereda las variables de instancia de la clase Gerente, necesita en su constructor hacer un llamado al constructor de clase superior Gerente para poder instanciarse. La clase Gerente hereda v.i. de la clase Fecha. Por lo que está bien que se declare como “extends Fecha”. La clase Fecha solo tiene atributos propios, sin heredar de ninguna clase.  
  
  
respuesta 2: ✓  
  
2- Falsa. Si la clase no declara ningún constructor, java incluye uno sin parámetros y sin código (constructor nulo).  
  
  
  
  
  
  
  
   
respuesta:

Falsa. Un modulo recursivo puede tener varias llamadas recursivas por ejemplo en una búsqueda dicotómica de forma recursiva ….

