Objetivos

1. Aprender a aplicar un patrón creacional a un contexto de diseño específico.
2. Comprender los detalles del patrón "Constructor".

Contexto y planteamiento del problema

Retomando el diseño del juego del laberinto y los monstruos, y con base en el modelo de clases que se utilizó como ejemplo para el patrón Fábrica Abstracta (Maze, Room, Door, Mapsite, Wall, etc) se requiere en este laboratorio guardar un laberinto (Maze) con el fin de que pueda ser compartido por los aficionados al juego. Se trata entonces de crear una representación textual de un laberinto basada por ejemplo en XML o en JSON que luego pueda ser copiada y cargada en la estación de juego de otro aficionado diferente al creador original del laberinto, de manera que el segundo pueda jugar usándolo. De momento se considera que estos dos formatos (XML y JSON) son imprescindibles, pero se quiere dejar abierta la posibilidad de agregar otros formatos estándar en el futuro sin que ello implique alterar el código que genera la representación textual.

Características de la solución:

Se debe generar una solución, basada en el patrón Constructor de tal forma que:

1. El diseño debe respetar los principios SOLID.
2. El diseño debe basarse en la aplicación correcta del patrón Constructor .
3. El código debe funcionar correctamente como “prueba de concepto”, tal como se ha mostrado en los ejemplos. El controlador debe demostrar que: 1) es posible seleccionar dinámicamente cuál de los formatos usar para crear la representación textual exportable, 2) es posible representar los diferentes componentes de un laberinto (Maze, Room, Door, Mapsite, Wall, etc) según el formato (XML o JSON) seleccionado, 3) es posible al final visualizar por consola el texto completo que representa un laberinto según corresponda con el formato seleccionado, 4) es posible crear una representación textual para un laberinto usando cada uno de estos dos formatos y 5) el laberinto mínimo debe incluir siete Rooms interconectados con al menos dos Rooms cada uno, 6) el laberinto puede ser con bombas o encantado.
4. Incluir el modelo de clases en formato pdf.

Evaluación:

Rubro	Peso
Aplicación correcta de principios SOLID	30%
Aplicación correcta de Constructor	40%
Funcionamiento correcto del código	20%
Uso correcto de UML en el modelo de clases	10%