Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2115 – Programación como Herramienta para la Ingeniería (II/2025)

Laboratorio 1

Aspectos generales

- **Objetivo:** evaluar individualmente el aprendizaje sobre el uso de técnicas de POO y estructuras de datos en un problema práctico de modelación y simulación.
- **Lugar de entrega:** Parte 1 lunes 18/08 a las 17:30, Parte 2 domingo 24/08 a las 23:59, ambas en el repositorio privado.
- **Formato de entrega:** archivos Python Notebook `L1_1.ipynb` y `L1_2.ipynb` con las soluciones de las partes 1 y 2 del laboratorio. Los archivos deben estar ubicados en la carpeta `L1`. Entregas que no cumplan el formato tendrán un descuento de 0,5 ptos.
- **Entregas atrasadas:** el descuento por atraso para la Parte 1 es de 1 punto cada 10 minutos o fracción. El descuento por atraso para la Parte 2 es de 1 punto por cada hora o fracción.
- **Issues:** Las discusiones en las *issues* del Syllabus que sean relevantes para el desarrollo de la evaluación, serán destacadas. Así mismo, el uso de librerías externas que solucionen aspectos fundamentales del problema no podrán ser utilizadas. Solo se podrán utilizar las que han sido aprobadas en las *issues*, previa consulta de los estudiantes.
- **Entregas con errores de sintaxis y/o que generen excepciones en todas las ejecuciones serán calificados con nota 1.0.**

Introducción

En este laboratorio usted deberá desarrollar un sistema inspirado en el clásico videojuego Sim Tower, donde se busca estudiar el funcionamiento de una torre urbana con múltiples pisos y espacios de distinto uso. El

objetivo es poder analizar, entre otros, cómo varía la ocupación de la torre, los ingresos que se generan y el impacto que tienen distintas configuraciones sobre su operación.

La torre puede albergar diversos tipos de espacios (residenciales, oficinas, comercios, entre otros) y recibir distintos tipos de personas (residentes, trabajadores, visitantes, entre otros), cuyos comportamientos afectan el desempeño global. Se espera que el sistema permita representar y simular estas dinámicas de manera que sea posible responder preguntas como: ¿cuántas personas puede albergar la torre y cuántas la están ocupando efectivamente?, ¿qué ingresos se generan según el tipo de espacio?, ¿cómo cambia la operación al modificar la proporción de espacios residenciales y comerciales?, o ¿qué ocurre con el uso de recursos limitados como los ascensores en distintos horarios?

La forma de modelar y organizar la información queda a su criterio. Lo importante es que el sistema propuesto sea capaz de responder de manera coherente a este tipo de preguntas, integrando elementos de programación orientada a objetos y estructuras de datos, y acompañando su diseño con pruebas explícitas que permitan verificar el funcionamiento de las simulaciones.

Parte 1

En esta primera parte se espera que el sistema desarrollado sea capaz de abordar preguntas relacionadas con la estructura básica y la organización de la torre. Para ello, su programa debe permitir representar la información de manera tal que usted pueda estudiar las siguientes situaciones:

1. **Capacidad y ocupación:** dada una torre con múltiples pisos y tipos de espacios, se desea conocer cuántas personas puede albergar en total y cuántas la están ocupando efectivamente.

2. **Ingresos generados:** algunos espacios producen ingresos (por ejemplo, tiendas o restaurantes) y otros gastos (por ejemplo, servicios comunitarios). Se desea calcular tanto los ingresos totales de la torre como los ingresos/gastos desagregados por tipo de espacio.

3. **Heterogeneidad de usos:** se busca distinguir qué proporción de la torre corresponde a cada tipo de espacio y cómo varía la ocupación en cada categoría.

4. **Otro aspecto no trivial:** se espera que proponga, justifique e implemente la medición de algún otro aspecto no trivial sobre la configuración de la torre.

La forma en que usted decida organizar su solución (clases, atributos, estructuras de datos, etc.) debe permitir responder a estas preguntas de manera consistente. No existe una jerarquía correcta específica, sino que se espera que la propuesta surja de su propio razonamiento.

Parte 2

En esta segunda parte se espera que el sistema pueda **simular el comportamiento dinámico de la torre** y entregar información que permita estudiar distintos escenarios de funcionamiento. Este comportamiento no solo se refiere a las personas, sino también al crecimiento de la infraestructura, al cambio de uso o abandono de los espacios, entre otros.
En particular, el programa debe ser capaz de abordar las siguientes situaciones:

1. **Movimiento de personas:** observar cómo varía la distribución de personas entre los distintos espacios de la torre a lo largo del tiempo, considerando que las personas pueden desplazarse entre pisos.

2. **Uso de recursos comunes:** analizar cómo afecta a la operación de la torre la existencia de recursos limitados, como ascensores con capacidad máxima, en horarios en que múltiples grupos de personas requieren utilizarlos.

3. **Efectos de configuraciones distintas:** comparar el comportamiento de la torre en al menos tres configuraciones diferentes (por ejemplo, una torre predominantemente residencial, otra centrada en oficinas y otra de carácter mixto), observando métricas como ocupación, ingresos o tiempos de espera.

4. **Visualización de la torre:** generar una representación simple de la estructura de la torre y de su estado durante la simulación. La visualización no necesita ser sofisticada: puede ser un diagrama en texto, un esquema en consola o una figura básica que muestre la distribución de pisos y unidades, junto con algún indicador de ocupación o flujo de personas.

5. **Checkpoints:** el sistema debe ser robusto ante posibles interrupciones de la simulación y permitir su recuperación en base a un archivo que contenga un *checkpoint* de su estado.

Para cada situación probada, debe quedar explícito cuál era el comportamiento esperado y qué resultado se obtuvo efectivamente al ejecutar la simulación, de modo que se pueda verificar la coherencia del sistema.

Consideraciones sobre la metodología de trabajo

En este laboratorio no solo interesa el resultado final, sino también el proceso seguido para construir la solución. Por ello, se espera que el desarrollo de su programa se organice a partir de una **división progresiva del problema en subproblemas más simples**, siguiendo un enfoque de *divide y vencerás*. La subdivisión debe reflejarse tanto en el diseño de clases e interacciones como en la organización del notebook: cada celda debe ser temáticamente coherente (por ejemplo, una clase compleja, una familia de clases, los tests asociados

a una clase, etc.), evitando reunir todo el código en una única celda. Todos los los supuestos y simplificaciones que se utilicen deben quedar claramente explicitados en el notebook. Además, siempre que sea posible, se recomienda complementar con diagramas.

Otro componente clave es el uso de **prompts estructurados** para interactuar con herramientas de IA. Cada vez que se utilice un asistente para generar código, se debe registrar en una celda de texto el prompt utilizado, incluyendo al menos:

- Propósito y requisitos,
- Entradas y salidas,
- Restricciones y supuestos,
- Hitos intermedios,
- Convenciones.

Este nivel de detalle permite mantener claridad sobre lo que se espera del código y facilita evaluar cómo la IA fue utilizada en el proceso de desarrollo.

Finalmente, todo el código debe estar acompañado de **tests explícitos**. Un test, en este contexto, es simplemente un fragmento de código que demuestra que otro fragmento funciona como se espera, considerando casos no triviales. No se exige un test unitario por cada método, pero sí que se diseñen pruebas convincentes que permitan validar comportamientos relevantes. Para cada caso probado, debe quedar visible el resultado esperado y el resultado obtenido, de modo que cualquier lector pueda verificar si el comportamiento es correcto.

Es importante enfatizar que **adjuntar el historial de interacción con un asistente de IA no constituye por sí mismo una justificación válida ni suficiente**. Dicho historial puede incluirse como material adicional si se desea, pero no sustituye en ningún caso los requisitos de formato, estructuración y testeo descritos anteriormente, los cuales son obligatorios y forman parte de la evaluación.

Política de Integridad Académica

Los/as estudiantes de la Escuela de Ingeniería de la Pontificia Universidad Católica de Chile deben mantener un comportamiento acorde a la Declaración de Principios de la Universidad. En particular, se espera que mantengan altos estándares de honestidad académica. Cualquier acto deshonesto o fraude académico está prohibido; los/as estudiantes que incurran en este tipo de acciones se exponen a un Procedimiento Sumario. Es responsabilidad de cada estudiante

conocer y respetar el documento sobre Integridad Académica publicado por la Dirección de Docencia de la Escuela de Ingeniería.

Específicamente, para los cursos del Departamento de Ciencia de la Computación, rige obligatoriamente la siguiente política de integridad académica. Todo trabajo presentado por un/a estudiante para los efectos de la evaluación de un curso debe ser hecho **individualmente** por el/la estudiante, **sin apoyo en material de terceros**. Por "trabajo" se entiende en general las interrogaciones escritas, las tareas de programación u otras, los trabajos de laboratorio, los proyectos, el examen, entre otros.

En particular, si un/a estudiante copia un trabajo, o si a un/a estudiante se le prueba que compró o intentó comprar un trabajo, **obtendrá nota final 1.1 en el curso** y se solicitará a la Dirección de Pregrado de la Escuela de Ingeniería que no le permita retirar el curso de la carga académica semestral.

Por "copia" se entiende incluir en el trabajo presentado como propio, partes hechas por otra persona. En caso que corresponda a "copia" a otros estudiantes, la sanción anterior se aplicará a todos los involucrados. En todos los casos, se informará a la Dirección de Pregrado de la Escuela de Ingeniería para que tome sanciones adicionales si lo estima conveniente.

También se entiende por copia extraer contenido sin modificarlo sustancialmente desde fuentes digitales como Wikipedia o mediante el uso de asistentes inteligentes como ChatGPT, Gemini o Copilot. Se entiende que una modificación sustancial involucra el análisis crítico de la información extraída y en consecuencia todas las modificaciones y mejoras que de este análisis se desprendan. Cualquiera sea el caso, el uso de fuentes bibliográficas, digitales o asistentes debe declararse de forma explícita, y debe indicarse cómo el/la estudiante mejoró la información extraída para cumplir con los objetivos de la actividad evaluativa.

Obviamente, está permitido usar material disponible públicamente, por ejemplo, libros o contenidos tomados de Internet, **siempre y cuando se incluya la referencia correspondiente**.

Lo anterior se entiende como complemento al Reglamento del Estudiante de la Pontificia Universidad Católica de Chile (`https://registrosacademicos.uc.cl/reglamentos/estudiantiles/`). Por ello, es posible pedir a la Universidad la aplicación de sanciones adicionales especificadas en dicho reglamento.

Compromiso del Código de Honor

Este curso suscribe el Código de Honor establecido por la Universidad, el que es vinculante. Todo trabajo evaluado en este curso debe ser propio. En caso que exista colaboración permitida con otros/as estudiantes, el trabajo deberá referenciar y atribuir correctamente dicha contribución a quien corresponda. Como estudiante es un debe conocer el Código de Honor (`https://www.uc.cl/codigo-de-honor/`).