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

IIC2115 – Programación como Herramienta para la Ingeniería (II/2023)

Actividad 1

Aspectos generales

- **Objetivo:** evaluar individualmente el aprendizaje sobre el uso de técnicas de POO y estructuras de datos en un problema práctico de manejo y consulta de información.
- **Lugar de entrega:** jueves 24 de agosto a las 23:59 hrs. en el repositorio privado.
- **Formato de entrega:** archivo Python Notebook (**A1.ipynb**) con la solución de la Actividad. El archivo debe estar ubicado en la carpeta **A1**. Utilice múltiples celdas de texto y código para facilitar el trabajo del cuerpo docente. Entregas que no cumplan el formato de entrega tendrán un descuento de 0,5 ptos.
- **Entregas atrasadas:** El descuento por atraso se realizará de acuerdo a lo definido en el programa del curso.
- **Issues:** Las discusiones en las *issues* del Syllabus que sean relevantes para el desarrollo de la evaluación, serán destacadas y se considerarán como parte de este enunciado. 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 esta actividad deberán desarrollar un sistema de consulta de información, construido a partir de los datos almacenados en un archivo en formato `.json`. Específicamente, deberán leer la información desde el archivo y almacenarla de manera adecuada a su estructura y contenido, para posteriormente responder una serie de consultas a partir del desarrollo de algoritmos que recorran de manera eficiente la información almacenada. Tanto las estructuras de datos utilizadas para almacenar la información, como los mecanismos utilizados para responder las consultas, deben ser modelados mediante POO, considerando atributos y comportamientos adecuados. En otras palabras, serán mal evaluadas las soluciones que estén estructuradas en base a "código suelto" distribuido en diferentes celdas.

Archivo `personas.json`

Se entregará un archivo `.json` con la información necesaria para construir de sistema. Cada elemento en el archivo representará a una persona, que será identificada mediante una clave (rut de la persona). El valor asociado a dicha clave será un diccionario que contiene los atributos de cada persona. Estos se describen a continuación:

- Nombre: no necesariamente es único en una familia.
- Apellido: se mantiene a lo largo de las generaciones.
- Sexo: "m" para masculino, "f" para femenino.
- Rut Padre: rut del padre. Puede ser `null`, que correspondería a no tener registro del padre.
- Rut Madre: rut de la madre. Puede ser `null`, que correspondería a no tener registro de la madre.

A continuación se muestra la estructura del archivo:

```
{
"16446564-2": {
  "nombre": "Daniel",
  "apellido": "Carrillo",
  "sexo": "m",
  "rut padre": null,
  "rut madre": null
```

```
},
"14543256-k": {
  "nombre": "Maria",
  "apellido": "Magga",
  "sexo": "f",
  "rut padre": null,
  "rut madre": null
}
}
```


Por simplicidad, el archivo entregado como ejemplo contendrá únicamente 3 familias, cuyos miembros heredarán siempre el apellido, ya sea de la madre o del padre, para mantenerlo a través de cada generación. Sin embargo, hay individuos con apellidos ajenos a las 3 familias, los que son cónyuges de algún familiar y de los que no se cuenta con registro de los padres. Para la corrección, **no pueden asumir** que siempre existirán 3 familias en el archivo a cargar, pero sí pueden asumir que el resto de características se mantendrán para cualquier archivo utilizado. Para trabajar con el archivo de personas, puede utilizar la librería **json** de Python.

Estructura de la solución

El programa realizado deberá cargar inicialmente la información desde el archivo de personas, entregando la ruta donde este se encuentra. Luego de realizar el proceso de lectura y estructuración de la información, el archivo no puede ser leído nuevamente.

Recién terminado el proceso anterior, podrá comenzar la ejecución de los algoritmos para responder a las consultas. Es decir, no podrá hacer cálculos ni almacenar información relacionada con las consultas al momento de leer la información del archivo.

Las consultas deben responderse una a una, especificando claramente el proceso que permite hacerlo. Los resultados deben ser impresos en pantalla en un formato claro y legible, indicando de manera expresa la solución. Puede asumir que los datos de entrada a la consulta siempre serán entregados de manera correcta. Con el fin de garantizar esto, cada consulta debe ir acompañada de un ejemplo de ejecución, que muestre imprima un resultado valido.

Consultas

- Antepasado en común: dado el rut de dos personas, se debe retornar el antepasado más cercano en común, y la distancia de parentesco entre estas 2 personas. Por ejemplo, dos primos tienen de antepasado común a uno de sus abuelos y la distancia entre los 2 primos es de 4: de un primo hasta su padre, del padre hasta el abuelo, del abuelo al hermano del padre, y desde el hermano del padre hasta el otro primo. En caso de no existir algún antepasado en común, debe indicarse de manera explícita.

- Descendiente más lejano: dado el rut de una persona, se debe retornar el descendiente más lejano de esta, entendiéndose por más lejano como la persona que se encuentra a la mayor distancia de parentesco directo.

- Familia nuclear más grande: se debe encontrar la familia nuclear más grande (hermanos, madre, padre), retornando su tamaño y los nombres de cada uno de sus miembros. En caso de existir nombre repetidos entre estos, cada uno debe mostrarse solo una vez.

- Personas más jóvenes: dado un numero entero k, se debe retornar a las k personas de mayor rut.

- Agregar persona: dados los datos de una persona, se debe agregar la persona al sistema. Esto implica actualizar la estructura utilizada para almacenar la información, ubicando correctamente en ella a la nueva persona. Si la persona a agregar no esta conectada a ninguna familia, se creara una nueva familia, donde su único miembro es la persona agregada.

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 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/`).