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

IIC2115 – Programación como Herramienta para la Ingeniería (II/2018)

Laboratorio 2

Objetivo

- Aplicar correctamente en distintos contextos estructuras de datos
- **Lenguaje a utilizar:** Python 3.6
- **Lugar:** repositorio compartido en GitHub.
- **Fecha límite de entrega: miércoles 12 de septiembre, a las 23:59 hrs.**
- **Formato de entrega:** dos archivos con la solución propuesta, uno en formato **.py** y el otro en **.ipynb**. Deben ser exactamente iguales (el ayudante revisará cualquiera de ellos). Adicionalmente, incluya un archivo **README.md** con información útil para el ayudante corrector.

Introducción

Un día cualquiera, mientras te encontrabas descansando en tu hogar, escuchaste que tocaban la puerta. Para tu sorpresa, la persona que se encontraba frente a ti decía ser tu hermano perdido, hecho bastante curioso, ya que acababas de recibir una millonaria herencia de un familiar lejano. Decides desconfiar del extraño y te propones crear un programa para verificar si lo que dice tu supuesto "hermano" es cierto.

Encuentra a tu familia

El programa que crearás tiene por nombre "Encuentra a tu familia". Deberás utilizar la información disponible en el archivo *personas.json* para armar los distintos arboles genealógicos de cada familia, para luego poder realizar distintas consultas y averiguar la real procedencia de tu "hermano".

Archivo: *personas.json*

Se entregará un archivo *.json* con la información inicial del sistema. Cada elemento en el archivo representará a una persona que será identificada mediante una *key* (rut de la persona). El valor de dicha *key* 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.
- Genero: "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",
  "genero": "m",
  "rut padre": null,
  "rut madre": null
},
"14543256-k": {
  "nombre": "Maria",
  "apellido": "Magga",
  "genero": "f",
  "rut padre": null,
  "rut madre": null
}
}
```

Por simplicidad, en el archivo entregado como ejemplo habrán 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. **No puede asumir** que siempre habrán 3 familias en los archivos a cargar, pero si puede asumir que el resto de características se mantendrán para cualquier archivo utilizado para corregir. Para trabajar con el archivo de personas, puede utilizar la libreria **json** de Python, cuya documentación se encuentra **aqui**
Con este archivo, deberá armar un grafo correspondiente a los árboles genealógicos de todas las familias presentes en él, sobre el que deberá realizar una serie de consultas.

Programa en consola

Deberá crear un programa en consola que permita cargar el archivo de personas (esto es, darle la ruta del archivo *json* de personas al programa) y realizar todas las consultas pedidas.

El funcionamiento del programa en consola es el siguiente: Se deben realizar todas las consultas antes de obtener sus resultados. Esto quiere decir que se debe permitir hacer consultas hasta que se especifique que no hay mas consultas por realizar. En ese momento se deben retornar las respuestas a todas las consultas, en el orden en el que se pidieron. Ademas, su programa debe ser resistente a errores, es decir, no se debe caer en ningún momento, y en particular, al ingresar algún *input* erróneo.

Consultas

Las consultas que debe permitir el sistema se especifican a continuación:

- Antepasado en común: Dado el rut de dos personas, se debe retornar el antepasado más cercano en común, y la distancia entre esas 2 personas en el árbol genealógico (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 en consola.

- Descendiente más lejano: Dado el rut de una persona, se debe retornar el descendiente más lejano de esa persona. Entiéndase por más lejano como la persona que se encuentra más abajo en el árbol genealógico de la familia.

- Cantidad de personas por generación: Dado el número de la generación y una familia en particular, se debe retornar el rut de todas las personas que pertenecen a esa generación (al menos con el árbol que se tenga en el momento), sin incluir cónyuges. Una generación son todas las personas que se encuentran en el mismo "nivel" en el árbol genealógico.

- Personas más jóvenes: Dado un numero entero k, se debe retornar a las k personas de mayor rut. (no necesariamente es una hoja del árbol ni tiene correspondencia con la generación). Para esta consulta se recomienda construir un grafo auxiliar ordenado.

- Familias con más generaciones: Dado un número entero k, esta consulta debe retornar a las k familias con más generaciones. Cada familia retornada debe ser una lista en donde cada persona de la familia es representada como un elemento de la lista. No asuma que sólo hay 3 familias. La función debe poder retornar valores para cualquier k .

- Mostrar Familia: Dado el apellido de una familia, debe imprimir de forma ordenada una representación de su árbol genealógico, mostrando a todos los miembros, incluyendo a aquellos que no poseen el apellido, pero que son padre o madre de algún miembro (los cónyuges). La consulta debe mostrar como opción a todas las familias presentes en el archivo abierto (que podría ser distinto al entregado y tener más familias, por ejemplo). *hint: en internet hay varios códigos que muestran árboles de forma ordenada como strings.*

- Agregar persona: Dados los datos de una persona, se debe agregar la persona al sistema. Esto implica modificar el grafo para que la persona se ubique correctamente dentro de él. Si la persona a agregar no esta conectada a ninguna familia, se creara una nueva familia donde su unico miembro es la persona agregada.

Política de Integridad Académica

Los alumnos de la Escuela de Ingeniería deben mantener un comportamiento acorde al Código de Honor de la Universidad:

"Como miembro de la comunidad de la Pontificia Universidad Católica de Chile me comprometo a respetar los principios y normativas que la rigen. Asimismo, prometo actuar con rectitud y honestidad en las relaciones con los demás integrantes de la comunidad y en la realización de todo trabajo, particularmente en aquellas actividades vinculadas a la docencia, el aprendizaje y la creación, difusión y transferencia del conocimiento. Además, velaré por la integridad de las personas y cuidaré los bienes de la Universidad."

En particular, se espera que mantengan altos estándares de honestidad académica. Cualquier acto deshonesto o fraude académico está prohibido; los alumnos que incurran en este tipo de acciones se exponen a un procedimiento sumario. Ejemplos de actos deshonestos son la copia, el uso de material o equipos no permitidos en las evaluaciones, el plagio, o la falsificación de identidad, entre otros. Específicamente, para los cursos del Departamento de Ciencia de la Computación, rige obligatoriamente la siguiente política de integridad académica en relación a copia y plagio: Todo trabajo presentado por un alumno (grupo) para los efectos de la evaluación de un curso debe ser hecho individualmente por el alumno (grupo), sin apoyo en material de terceros. Si un alumno (grupo) copia un trabajo, se le calificará con nota 1.0 en dicha evaluación y dependiendo de la gravedad de sus acciones podrá tener un 1.0 en todo ese ítem de evaluaciones o un 1.1 en el curso. Además, los antecedentes serán enviados a la Dirección de Docencia de la Escuela de Ingeniería para evaluar posteriores sanciones en conjunto con la Universidad, las que pueden incluir un procedimiento sumario. Por "copia" o "plagio" se entiende incluir en el trabajo presentado como propio, partes desarrolladas por otra persona. Está permitido usar material disponible públicamente, por ejemplo, libros o contenidos tomados de Internet, siempre y cuando se incluya la cita correspondiente.