Tarea 3 - List (listas Python), Diccionarios, Archivos, 'while' y 'for'

"Sistema de Trazabilidad de Contactos"

2020-12-02

Profesores: Bárbara Poblete y Alejandro Hevia. CC1002, Secciones: 5 y 7, Primavera 2020

Lea el enunciado completo antes de comenzar a trabajar

En esta tarea realizaremos un algoritmo para identificar personas potencialmente expuestas a contagio de una enfermedad de transmisión aérea (como lo es el covid-19 en la práctica, ya que se transmite principalmente por aerosol). Este proceso se denomina "**trazabilidad de contactos**" y, en su versión más simple, requiere una lista de los lugares visitados por las personas, la hora en que estuvieron ahí, y además de una lista de quiénes están contagiados (o sospechosos de estar contagiados). El propósito del proceso es identificar a los **"contactos estrechos"** que, en este caso, son personas que estuvieron al mismo tiempo y lugar que una persona infectada (o sospechosa).[1] De esta forma, los contactos estrechos pueden auto-aislarse hasta descubrir si están infectadas o no, sin exponer al resto de la comunidad a más contagios. El proceso de notificar a los contactos estrechos, se denomina **"trazabilidad de contactos"**. Al ser aplicada de forma correcta y rigurosa, esta estrategia ha demostrado ser uno de los mecanismos efectivos para ayudar al control de la propagación de la pandemia en ciertos países.

Su trabajo en esta tarea es crear un sistema que permita realizar trazabilidad de contactos por medio de identificación de contactos estrechos. Para esto se le proporcionará *un archivo de texto con las ubicaciones de todas las personas* de las cuales se tiene registro.[2] El formato del archivo es:

```
id_usuario,lugar,hora
```

Donde `id_usuario` es un `string` que identifica de forma única a un usuario, `lugar` es un `string` que identifica de forma única un lugar en que estuvo el usuario, y `hora` es un `int` (de 0-23 que representa las horas de un día). Por simplicidad se redondea a una hora completa la permanencia de cada persona en un lugar y se considera como que dos personas estuvieron en contacto estrecho si visitaron al menos un lugar a la misma hora. Note que no hay líneas repetidas en el archivo y que un usuario visita un lugar solo una vez. A continuación se muestra un ejemplo del archivo de ubicaciones, pero **su tarea debe funcionar con cualquier otro archivo** que cumpla con este formato (se revisará con un archivo diferente, por lo que pruebe con otros ejemplos):

```
jperez,mercado1,13
cgonzalez,feria2,10
jperez,cafeteria3,15
rmunoz,feria2,10
rmunoz,tienda2,11
aruiz,cafeteria3,15
rmunoz,mercado1,13
jperez,tienda2,11
```

Su sistema deberá seguir un diálogo como el que se muestra en el siguiente ejemplo de uso (en `rojo` lo que ingresa el usuario):

---

[1] Definición simplificada para el propósito de esta tarea. Técnicamente los contactos estrechos deben permanecer en una misma habitación por un lapso mayor a cierto tiempo (15 minutos como mínimo).

[2] Usamos esta solución específica por simplicidad, pero no se recomienda en la vida real monitorear de forma centralizada dónde están todos los ciudadanos en todo momento, ya que atenta contra la privacidad. Más información en la última página del enunciado.

```
Sistema Trazador de Contactos COVID19

Ingrese el nombre del archivo de ubicaciones: ubicaciones.txt
Archivo ubicaciones.txt cargado al sistema.

Ingrese el identificador de la persona sospechosa de covid (o 'fin'): rperezu
        No se registran movimientos para el usuario: rperezu

Ingrese el identificador de la persona sospechosa de covid (o 'fin'): jperez
        Contacto estrecho rmunoz
        jperez coincidio con rmunoz en mercado1 a las 13 horas
        jperez coincidio con rmunoz en tienda2 a las 11 horas

        Contacto estrecho aruiz
        jperez coincidio con aruiz en cafeteria3 a las 15 horas

Ingrese el identificador de la persona sospechosa de covid (o 'fin'): aruiz
        Contacto estrecho jperez
        aruiz coincidio con jperez en cafeteria3 a las 15 horas

Ingrese el identificador de la persona sospechosa de covid (o 'fin'): fin

Gracias por usar el Sistema Trazador de Contactos COVID19.
```

Su tarea debe además cumplir con los siguientes requisitos de implementación:

La información que se lee del archivo con ubicaciones debe ser almacenada en una variable global de tipo `list` (lista Python, no use listas recursivas). Esta lista debe almacenar valores de tipo **`usuario`**, que es un la estructura mutable definida de la siguiente forma:

```
import estructura

# persona: id_usuario (str), lugares_visitados (dict)
estructura.crear("usuario","id_usuario lugares_visitados")
```

En donde **`id_usuario`** es una variable de tipo `string` y contendrá un identificador del usuario (ej. **`"jperez"`**) y **`lugares_visitados`** es de tipo `dict` (diccionario de Python) que corresponde a un diccionario con los lugares visitados (en donde la llave del diccionario es el nombre del lugar, e.j. **`"mercado1"`** y su valor es la hora en que se visitó el lugar). Por ejemplo, en el caso de la persona **`jperez`** del archivo de ejemplo, su estructura de tipo `usuario` que se debe agregar a la lista es igual a:

```
usuario('jperez', {'mercado1': 13, 'cafeteria3': 15, 'tienda2': 11})
```

Además, Ud. **debe implementar** las siguientes funciones, incluyendo sus recetas de diseño. Luego, utilizando estas funciones deberá implementar su programa principal, llamado **`tarea3.py`** que ejecute el diálogo indicado:

1. **`cargarUbicacionesAListaarchivo`**: este procedimiento recibe un string con un nombre de un archivo de texto como parámetro. Al ejecutarse, debe leer una por una las líneas del archivo y al finalizar. El efecto de ejecutar este procedimiento el el de llenar correctamente con estructuras de tipo **`usuario`** la lista global. Por ejemplo, para el archivo de prueba indicado más arriba la lista deberá ser igual a:

```
[usuario('jperez',{'mercado1':'13','cafeteria3':'15','tienda2':'11'}),
 usuario('cgonzalez',{'feria2':'10'}),
 usuario('rmunoz',{'feria2':'10','tienda2':'11','mercado1':'13'}),
 usuario('aruiz',{'cafeteria3':'15'})]
```

Notar que el archivo puede tener varias líneas para el mismo usuario. En el ej. '`jperez`' tiene 3 líneas, uno en `mercado1` a las 13 horas, otro en `cafeteria3` a las 15 horas y otro en `tienda2` a las 11 horas, sin embargo la lista global sólo debe crear una estructura de tipo `usuario` para `jperez`.

1. **`indiceId(id_usuario):`** esta función recibe un string con el identificador de un usuario (ej. '`jperez`') y devuelve el índice en `la lista global` donde está almacenada la estructura usuario correspondiente. En el ejemplo anterior, **`indiceId('jperez')`** debe retornar **`0`**.

   Bonus: Si implementa un test para esta función, tendrá 0.5 puntos extras. Note que no es tan directo pues esta función usa una lista como variable global.

2. **`contactoEstrecho(usuario1,usuario2):`** esta función recibe *dos estructuras de tipo* **`usuario`** (usuario1 y usuario2) y retorna `True` si hubo contacto estrecho entre ambos. Es decir, si usuario1 y usuario2 estuvieron en el mismo lugar a la misma hora. En el ejemplo:
   `contactoEstrecho(usuario('cgonzalez',{'feria2':10}),usuario('rmunoz',{'feria2':10,'tienda2':11,'mercado1':13}))` retorna **`True.`** Para esta función debe incluir tests además de la receta de diseño.

3. **`imprimeContactosEstrechosEntre(usuario1,usuario2):`** este procedimiento recibe dos estructuras usuario e imprime en pantalla líneas que muestran dónde y cuándo tuvieron contacto estrecho. Por ejemplo,

```
imprimeContactosEstrechosEntre(
 usuario("jperez", {"mercado1":13,"cafeteria3":15,"tienda2":11}),
 usuario("rmunoz", {"feria1":10,"tienda2":11,"mercado1":13}))
```

   Imprime

```
jperez coincidio con rmunoz en mercado1 a las 13 horas
jperez coincidio con rmunoz en tienda2 a las 11 horas
```

   `Para esta función debe incluir tests además de la receta de diseño.`

4. **`imprimeContactosEstrechosDe(id_sospechoso):`** este procedimiento recibe un string con un identificador de usuario (ej. '`jperez`') e imprime en pantalla líneas que muestran con quién, dónde y cuándo tuvo contacto estrecho con otras personas de acuerdo a lo almacenado en la variable global **`lista_usuarios.`** Esta función debe usar las funciones **`contactoEstrecho()`** e **`imprimeContactosEstrechosEntre()`** definidas anteriormente. En el ejemplo,

   `imprimeContactosEstrechosDe('jperez')` imprime en pantalla:

```
        Contacto estrecho rmunoz
        jperez coincidio con rmunoz en mercado1 a las 13 horas
        jperez coincidio con rmunoz en tienda2 a las 11 horas

        Contacto estrecho aruiz
        jperez coincidio con aruiz en cafeteria3 a las 15 horas
```

Importante:

- Para hacer esta tarea puede usar todos los módulos vistos en clases hasta la semana 11 (inclusive).
- El trabajo es **individual**, preocúpese de leer las indicaciones del curso con respecto a *política de copias y formas permitidas de colaboración*. Se verificarán de forma automatizada que no exista plagio en las tareas.
- Debe enviar su tarea en la fecha de entrega indicada en u-cursos, puede hacer sus consultas de enunciado en las sesiones de consulta con los profesores, auxiliares, o en el foro del curso/Discord. En la sesión de consultas con los profes se explicará el enunciado. Se aplicará un descuento de 1 punto por día hábil de retraso en entregar

SECCIÓN CULTURA GENERAL (nada que implementar o retornar, sólo información general):

Respecto a la privacidad de los ciudadanos al usar el algoritmo indicado más arriba:

- Este algoritmo de trazabilidad no es el recomendado pues, en entornos autoritarios, puede permitir violar la privacidad de los ciudadanos al posibilitar el monitoreo y seguimiento digital respecto a dónde y con quiénes se juntan los ciudadanos, posiblemente afectando a disidentes o activitas políticos.
- Para preservar la privacidad de los ciudadanos, actualmente el proceso de trazabilidad de contactos realizado por Google y Apple (denominado “Notificación de Exposición” en este caso) se basa en un algoritmo denominado DP3T, diseñado con privacidad en mente por un consorcio académico y científico europeo a inicios del 2020. En este algoritmo, los gobiernos NO pueden monitorear a sus ciudadanos y la idea es que cada ciudadano puede evaluar por sí solo si estuvo en contacto cercano con alguien posiblemente infectado.
- ¿Cómo funciona? El algoritmo requiere que cada persona tenga un teléfono inteligente. Cada teléfono puede detectar que otro teléfono está cerca pues pueden comunicarse usando un tipo de comunicación para distancia corta (pocos metros) llamado Bluetooth Low Energy (BLE). Si implementa DP3T, cada teléfono hace lo siguiente:
    - Al "encontrarse" cerca de otro teléfono, ambos teléfonos intercambian mensajes donde se "presentan" enviando sus respectivos identificadores. Por ejemplo, si el teléfono de Alejandro está cerca de Bárbara, el teléfono de Alejandro le envía su identificador al de Bárbara, y el de Bárbara le envía su identificador al de Alejandro. Pero esto viola la privacidad de los dueños de los teléfonos? En realidad, no, pues los identificadores van cambiando periódicamente! Cada teléfono genera un identificador propio al azar (llamado "token" o "mensaje") cada 5 min por ejemplo, pero los va recordando.
    - Cada teléfono no solo envía mensajes, sino que almacena los identificadores únicos de quienes "ve pasar". Esto produce en mi teléfono un contenido similar al del archivo “registros.txt” pero sin lugares y sólo con los identificadores de teléfonos cercanos a mi teléfono.
- Si Bárbara se enferma, por ejemplo, le enviará a un servidor centralizado la lista de todos los identificadores (mensajes) únicos que ella envió. El servidor luego los publica. El teléfono de Alejandro podrá luego bajar la lista de todos los identificadores de todos los contagiados (incluidos los de Bárbara) y evaluar si tuvo contacto estrecho con alguno de ellos. Y por ejemplo, fue así con Bárbara, el teléfono de Alejandro alerta a su dueño sobre esto, solicitando que se haga un examen y se auto-aisle.
- Noten que el algoritmo no le entrega al gobierno ningún tipo de información respecto a quién se junta con quién, preservando la privacidad de los ciudadanos.
- Para más información, se recomienda ver
    - Historieta de cómo funciona DP3T https://github.com/DP-3T/documents/blob/master/public_engagement/cartoon/es/combined_panels.pdf

- Video explicativo de cómo funciona DP3T (en la versión usada por Apple-Google), https://vimeo.com/461442424, o https://youtu.be/xZrjIxwSd2I
- Aplicación Radar COVID (España), que implementa el sistema de Apple-Google, https://www.xataka.com/aplicaciones/probamos-radar-covid-asi-funciona-aplicacion-rastreo-contactos-que-usaremos-espana