PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN

**IIC2233 - Programación Avanzada**
**$1^{er}$ semestre 2016**

# Actividad 13

## Manejo de Strings y ByteArrays

### Instrucciones

**TOP SECRET**: El destino del país y probablemente del mundo se encuentra en sus manos. La *NATSA* tiene el importante proyecto *ETs* para detectar vida alienígena. Después de 33 años apuntando sus radares hacia la galaxia IIC22-33, al fin se ha detectado lo que podría ser un mensaje alienígena, el cual fue almacenado en el archivo **"TOP SECRET.iic2233"**. Lamentablemente nuestros especialistas se encuentran trabajando arduamente con el Dr. Karim y el Dr. Christian, por lo que requerimos de usted para descifrar el enigma.

#### Traduciendo el mensaje

Los técnicos de la *NATSA* antes de abandonar el proyecto consiguieron extraer la siguiente *metadata* del archivo:

- Existen dos archivos ocultos. Se presume un archivo de audio **.WAV** y un archivo de imagen **.GIF**.
- El archivo se encuentra estructurado de la siguiente manera:

  [*Header*, .WAV - *Chunk* 1, .GIF - *Chunk* 1, ... , .WAV - *Chunk* 4, .GIF - *Chunk* 4]

  donde *header* corresponde a la metadata del archivo de audio.
- Cómo extraer el archivo de audio:
  - El *header* es de tamaño 44.
  - Cada *chunk* del archivo de audio es de tamaño 74300.
  - Lamentablemente los trozos del archivo de audio presentan anomalías. Se cree que al momento de construir el mensaje, el archivo de audio, dividido en cuatro partes, fue invertido, al igual que cada una de sus partes, es decir:

    $$\text{Inicialmente: } [header, chunk1, chunk2, chunk3, chunk4]$$
    $$\text{Finalmente: } [header, 4knuhc, 3knuhc, 2knuhc, 1knuhc]$$

    El *header* se mantuvo intacto durante la creación del mensaje.
- Cómo extraer el archivo de imagen:

- Cada *chunk* del archivo de imagen es de tamaño distinto:

$$\textit{Chunk}\ 1 = 114587;\ \textit{Chunk}\ 2 = 452976;\ \textit{Chunk}\ 3 = 54325;\ \textit{Chunk}\ 4 = 163936$$

- Se cree que los trozos del archivo de imagen no presentan ninguna anomalía ni modificación producto de la construcción del mensaje.

**Escribiendo la información**

Una vez obtenidos los bytes/bytearrays que componen al archivo de imagen y al archivo de audio corregido, es necesario que sean escritos mediante **buffering**. Esto quiere decir que la escritura se realice por medio de transportadores de información de tamaño fijo, evitando escribir la información por completo de una sola vez. Este método de lectura/escritura es bastante útil cuando se trata con información de gran tamaño. Su aplicación en Python deberá simular esta técnica escribiendo en los archivos mediante bytes/bytearrays de tamaño:

- **512** en el caso del archivo de audio.
- **1024** en el caso del archivo de imagen.

Es necesario que en cada iteración de escritura se haga saber al usuario el estado del proceso. Para esto se le pide que imprima una tabla de seguimiento similar a la siguiente:

| TOTAL | PROCESADO | SIN PROCESAR | DELTATIME |
|---|---|---|---|
| 785810 | 778240 | 7570 | 0.029144 |

- **TOTAL** corresponde a la cantidad de bytes que deben ser escritos.
- **PROCESADO** corresponde a la cantidad de bytes que se han conseguido escribir hasta ahora.
- **SIN PROCESAR** corresponde a la cantidad de bytes que aún no han sido escritos.
- **DELTATIME** corresponde a la diferencia de tiempo entre que comenzó la escritura y el tiempo actual.

Usted es libre de agregar cualquier otro parámetro de interés a la tabla si lo considera pertinente.

## Requerimientos

- Utilizando la información entregada por nuestros científicos, rescatar, reestructurar y reconstruir la información correspondiente al archivo de audio y al archivo de imagen por separado.
- La escritura de la información de los archivos rescatados debe ser realizada mediante **buffering**.
- La tabla encargada de mostrar la información durante la escritura debe encontrarse sólidamente estructurada mediante **manejo de strings**.

## Bonus

Como científico de la *NATSA* se espera de usted una trabajo de excelencia. Para conseguir esto es necesario que la escritura de los archivos sea realizada de manera asíncrona, es decir, mediante el uso de **threads**, puesto que es un problema que puede ser perfectamente paralelizable dentro de lo que Python permite. La paralelización generará conflictos al momento de la impresión de la tabla de seguimiento que deberán ser solucionados como usted estime conveniente. Documente los puntos claves de la solución propuesta explicando brevemente el por qué se decidió por ella.

## Notas

- Está permitido utilizar **.reverse()** para invertir una lista pero **NO para invertir bytearrays**.

## To - DO

- (6.00 pts) Principal
  - (1.00 pts) Correcta modelación del problema.
  - (1.25 pts) Información del archivo de audio extraída y reconstruida.
  - (1.25 pts) Información del archivo de imagen extraída y reconstruida.
  - (1.50 pts) Escritura mediante **buffering**.
  - (1.00 pts) Tabla de seguimiento.
- (2.00 pts) Bonus
  - (1.00 pts) Escritura asíncrona.
  - (1.00 pts) Tabla de seguimiento funcional ante la paralelización del problema.

## Tips

- Las librerías **datetime** y **time** pueden ser de utilidad para calcular el tiempo que tarda en realizarse el proceso de escritura. Es libre de usar cualquiera de ellas.