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

IIC2233 - Programación Avanzada
2^o semestre 2016

Actividad 12

Manejo de strings, bytes y bytearrays

Introducción

¡La Progra Perla ha llegado a la isla!. De la mano de su temido capitán, los piratas hacen arribo a la isla DCcé, donde te capturan y esclavizan para cumplir sus cometidos. El amo y señor de los mares, el sublime capitán Mabraquis, tiene en sus manos los archivos del tesoro, los que planea descifrar y hacerse con toda la riqueza que se encuentra oculta bajo estas lejanas tierras. Como puede suponer, el capitán tiene muchas otras cosas que hacer[1] y le encarga esta gran tarea.

Instrucciones

Junto con el enunciado, podrá encontrar dos archivos: `archivo1` y `archivo2`, ambos estrechamente ligados. En `archivo1` se encuentra la información necesaria para la desencriptación `archivo2`. Para llevarlos a su forma original, el capitán le entrega la siguiente información:

Parte I: Descifrando la clave

En `archivo1` se encuentra la clave que se utilizará en la segunda parte. La estructura de `archivo1` es la siguiente:

$$Chunk_1 - Chunk_2 - Chunk_3 - .... - Chunk_n$$

Donde el tamaño de cada chunk (en bytes) se corresponde con su número de Fibonacci, es decir, el primer chunk tiene tamaño 1 al igual que el segundo, y el tamaño del n-ésimo chunk se consigue de la forma:

$$Tamano\_Chunk_N = Tamano\_Chunk_{N-1} + Tamano\_Chunk_{N-2}$$

Además, debido a las turbulencias en el barco, cada chunk del archivo se ha invertido, por lo tanto, quedan escritos como:

$$knuhC_1 - knuhC_2 - knuhC_3 - .... - knuhC_n$$

Entre los ininteligibles gritos del buen capitán Mabraquis acerca de este archivo usted logra entender que después de reordenar los chuncks, debe guardar el archivo con extensión `.pdf` sin importar el nombre de este.

[1] como rankear en el rocket league, es trabajo de capitán

Parte II: En búsqueda del abundante tesoro

- Recuperando los archivos:

Para continuar, deberá haber realizado la parte I correctamente y haber abierto el archivo `.pdf` resultante. A continuación, se referirá como *resultado* a la suma de cada code point de la clave obtenida en la primera parte (incluyendo espacios) en su forma de unicode. Al final del enunciado se indica la función de python que permite calcular este número. Luego, sobre la suma aplicamos la operación `mod256`, así tendremos el resultado final. Donde *mod256* hace referencia a la aritmética modular[2]. A modo de ejemplo, $[5]_{mod3} = 2$, ya que al dividir 5 entre 3, el resto de la división es 2.

Cada byte en `archivo2` ha sido modificado con este resultado. Así, cada byte original se obtiene haciendo:

$$byte\_original_i = [byte\_archivo_i + resultado]_{mod256}$$

- Separando los archivos:

Finalmente, para separar los ~~tesoros~~ archivos ocultos en `archivo2` deberá considerar la siguiente estructura:

$$C_1F_1 - C_1F_2 - C_2F_1 - C_2F_2 - C_3F_1 - C_3F_2 - ...C_iF_1 - C_iF_2...C_{n-1}F_x - C_nF_x$$
$$C_n = Chunk_n$$
$$F_n = File_n$$

Como podrá notar, los archivos están entrelazados. El tamaño (en bytes) de cada chunk se corresponde según su número abundante. Es decir, el tamaño del i-ésimo chunk (para el archivo 1 o 2) es igual al i-ésimo número abundante. Además, podrá ver que los últimos chunks corresponden a un mismo archivo, esto es debido a que los archivos **no** necesariamente tienen el mismo largo[3]. Pero no desespere, el ingenioso capitán Mabraquis le susurra al oído el tamaño de uno de los archivos: **1456619 Bytes (.mp3)**

Números Abundantes

En matemática, un **número abundante** o un **número excesivo** x, es aquel que la suma de todos sus divisores positivos, incluyendo el propio x, es mayor al doble de x. Digamos que la función que calcula la suma de los divisores de x es γ(x) entonces se debe cumplir que:

$$\gamma(x) > 2x$$

Algunos ejemplos de numeros abundantes:

- 12, 18, 20, 24, 30, 36, 40...

Tomando por ejemplo el 24: sus divisores son el 1, 2, 3, 4, 6, 8, 12 y 24 la suma de estos es 60 y como 60 es mayor a $2 \cdot 24$ (=48) entonces 24 es un número abundante.
Puede leer más al respecto haciendo click aquí.

[2]Si le quedan dudas, puede visitar esta página
[3]Ver tips

Reportando al capitán

Para dar cuenta de su trabajo deberá imprimir en consola el avance de su desencripción en forma de tabla, la que deberá contener: **cantidad y porcentaje de bytes procesados y sin procesar, y tiempo de la iteración actual**. Es decir, para cada iteración (sea de lectura o escritura de archivo) se debe imprimir en dicha tabla el estado del proceso. Además, los porcentajes deben mostrarse en el formato VWX.YZ donde se debe mostrar siempre los dígitos WXYZ (correspondientes a decenas, unidades, decima y centésima respectivamente) y se debe mostrar el dígito V solo cuando corresponda. Por ejemplo:

Los siguientes porcentajes siguen el formato especificado:

- 04.20 %
- 45.03 %
- 100.00 %

Los siguientes porcentajes **NO** siguen el formato especificado:

- 4.20 %
- 45.032 %
- 100.0 %

Es decir debe verse más o menos así:

TOTAL	PROCESADO	SINPROCESAR	PERCENT.	DELTATIME
785796	0	785796	00.00%	0.00018311
785796	1024	784772	00.13%	0.00036073
785796	2048	783748	00.26%	0.00047708
785796	3072	782724	00.39%	0.00059080
785796	4096	781700	00.52%	0.00081825

Finalmente, a modo de resumen para el excelso capitán Mabraquis, debe imprimir en una tabla similar la información sobre la cantidad total de bytes procesados y número total de iteraciones al finalizar cada parte.

A modo de ejemplo será algo más o menos así:

PARTE	PROCESADOS	ITERACIONES
I	123456	56
II	654321	43

Usted es libre de añadir cualquier otro aspecto a mostrar que usted considere pertinente a cualquiera de las dos tablas.

Bonus

Luego de verlo trabajar arduamente para recuperar los archivos originales, el magnánimo capitán Mabraquis le encarga crear un programa que, dada una clave como string y dos archivos, permita combinarlos de la misma manera en la que estaban combinados los archivos en `archivo2`, es decir, realizar el proceso inverso al que acaba de realizar para la parte II.

Requerimientos

- Con la información entregada por el capitán: leer, reestructurar y entregar los archivos originales.
- La tabla encargada de mostrar la información durante la escritura debe encontrarse sólidamente estructurada mediante manejo de strings.

Notas

- Procure ser cuidadoso siguiendo las estructuras señaladas y en su desencriptación: cualquier error, por pequeño que sea, no permitirá recuperar los archivos originales.
- El archivo con la clave tiene formato File.pdf
- El formato de los arvhicos resultantes son: `File1.gif` y `File2.mp3` (cada uno con los tamaños respectivos entregados anteriormente).
- El nombre de los archivos finales queda a su criterio, pero respetar el tipo de archivo (`.gif` y `.mp3`).

To - DO

- (1.00 pts) Escribir el proceso en consola.
- (2.00 pts) Decifrar el primer archivo y conseguir la clave.
- (3.00 pts) Decifrar el segundo archivo y conseguir los dos archivos resultantes.
- (1.00 pts) Combinar dos archivos de forma inversa a la segunda parte, a partir de una clave. **(BONUS)**

Tips

- Puede usar *os.path.getsize(filepath)* para obtener el tamaño de un archivo
- El built-in ord(c) entrega el entero que representa el code point unicode para un caracter c.
- Le reiteramos: es **muy importante** resolver la primera parte antes que la segunda, ya que no son independientes.

Entrega

- **Lugar:** GIT - Carpeta: Actividades/AC12
- **Hora:** 16:55