Pontificia Universidad Católica de Chile
Escuela de ingeniería
Departamento de Ciencia de la Computación
Segundo Semestre del 2014

IIC1103 Introducción a la Programación

Tarea 2

Objetivo General

Esta tarea tiene como objetivo que te enfrentes a un desafío de programación que te permita practicar los conocimientos adquiridos de Python, centrados principalmente en los tópicos de definición y uso de *funciones,* y cadenas de texto (*strings*). Para el logro de este objetivo deberás hacer una implementación en Python del sistema de mensajería llamado ***WhatsIntro.***

Introducción

Una de las principales funciones de la criptografía es lograr una comunicación segura entre dos puntos. En términos simples, esto significa establecer un canal para enviar mensajes encriptados de tal forma que si un tercero intercepta un mensaje no sea capaz de descifrarlo para obtener el contenido original. El término es muy antiguo y ha sido utilizado desde antes de la era digital - por ejemplo, Julio César usaba mensajes encriptados para comunicarse con sus generales.

El protocolo de encriptación *CipherSaber* ofrece una comunicación segura, basada en que los dos puntos que se comunican conocen una clave que no conoce nadie más. Una versión simplificada de este protocolo funciona de la siguiente manera:

1. Los puntos que se van a comunicar se ponen de acuerdo en una clave de largo máximo 246 caracteres. El tamaño del mensaje a ser enviado no puede exceder los 256 caracteres.
2. Luego se transforma el mensaje a enviar en una cadena de ceros y unos siguiendo estos pasos:
    a. Cada caracter del mensaje es transformado en un número entero positivo según la codificación ASCII [1]. Por ejemplo, los números de esta codificación para la cadena *hola* son 104, 111, 108 y 97.
    b. Luego, el número ASCII es transformado a la correspondiente secuencia binaria de 8 dígitos. Si ésta tiene menos de 8 dígitos, entonces se agregarán ceros a la izquierda hasta obtener una cadena de ocho dígitos.
    c. Finalmente todas las representaciones binarias son unidas en una sola cadena de texto.

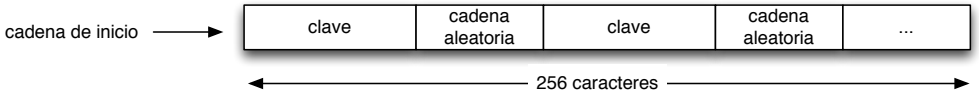
    La siguiente tabla muestra la codificación en ceros y unos del mensaje “hola”

[1] http://msdn.microsoft.com/en-us/library/60ecse8t(v=vs.80).aspx

Caracter	Código ASCII	Codificación Binaria (ceros y unos)
h	104	`01101000`
o	111	`01101111`
l	118	`01101100`
a	97	`01100001`
hola	104 111 118 97	`01101000011011110110110001100001`

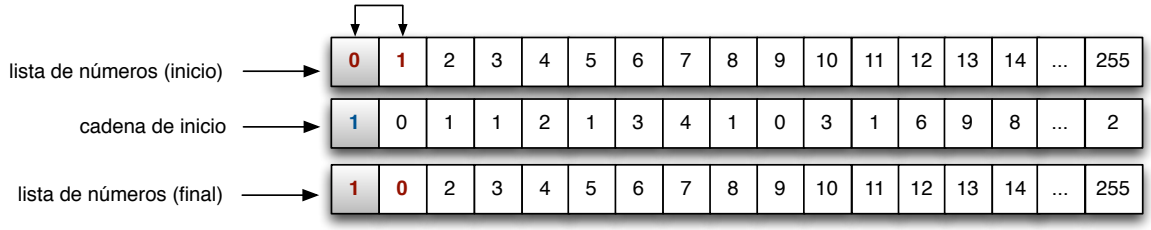
3. Luego, se crea la *cadena aleatoria* que consiste de una secuencia de diez dígitos elegidos al azar entre 0 y 9. Para descifrar el mensaje esta cadena aleatoria es obtenida de los primeros 10 elementos del mensaje a descifrar.

4. Se crea la *cadena de inicio* uniendo la clave con la cadena aleatoria (obtenida del paso 3). Esto se repite hasta obtener una secuencia de 256 caracteres. Ten en cuenta que no siempre tendrás que repetir todo para obtener exactamente la cantidad de caracteres requeridos, para esto solo tendrás que copiar la parte necesaria.

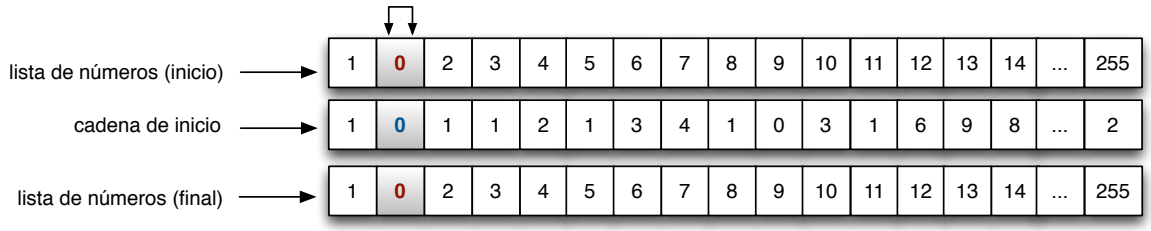


5. Crea una lista de los números 0 hasta 255. Luego cambia el orden de todos los números de esta lista de la siguiente forma: el número de la posición *i* se intercambia con el número de la posición *j*, siendo *j* el valor de la suma de *i* y el valor obtenido de la cadena de inicio en el índice *i*. Por ejemplo:

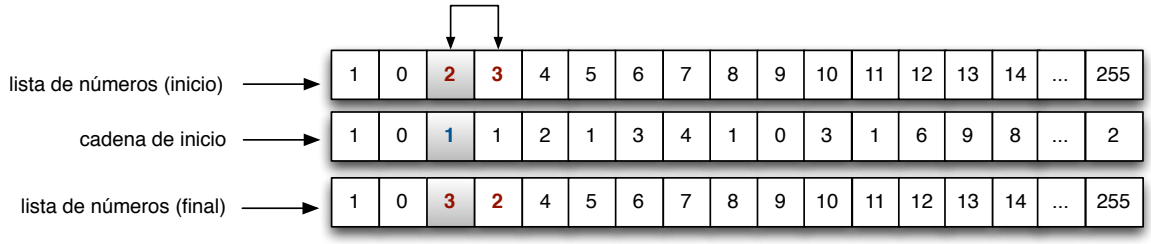
   a. El número de la posición "0" ($i = 0$) será intercambiado con el número de la posición $j = 0$ + cadena de inicio[0] = 0+1 = 1.



   b. Luego, el número de la posición 1 ($i = 1$), será intercambiado con el de la posición $j = 1$+ cadena de inicio[1] = 1+0 = 1 (que es lo mismo que no sea intercambiado).



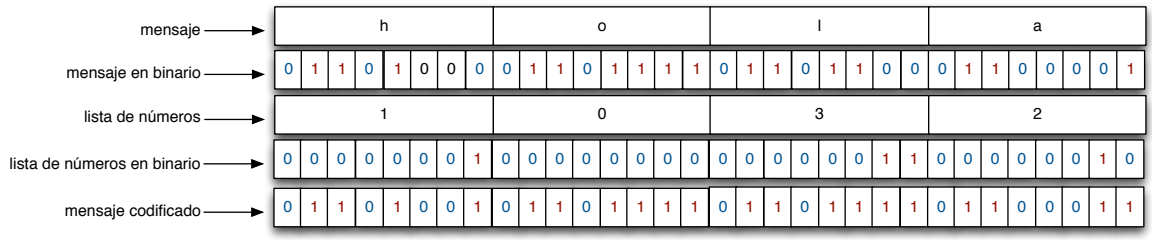
   c. De la misma forma, el número de la posición 2 ($i = 2$), será intercambiado con el de la posición j = 2+cadena de inicio[2] = 2+1 = 3. Esto se repite para todos los elementos de la lista.



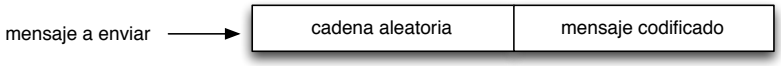
d. El último número de la lista (posición 255) es un caso especial ya que eventualmente será intercambiado con el siguiente, y este no existe. Para este caso considera que el siguiente elemento al último de la lista es el de la posición 0, el subsiguiente el de la posición 1, y así sucesivamente.

6. Finalmente se codifica el mensaje siguiendo los siguientes pasos:
   a. Cada caracter del mensaje es transformado a un secuencia binaria de ocho dígitos de acuerdo al paso 2. Luego se crea una cadena concatenando todas estas secuencias binarias.
   b. Cada elemento de la lista de números obtenida en el paso 5 es transformado a una secuencia binaria de ocho dígitos de acuerdo al paso 2. Luego, se genera una cadena concatenando todas estas secuencias.
   c. Con las secuencias binarias de los pasos 6a y 6b, se genera finalmente el mensaje codificado. Este mensaje consiste en una cadena de dígitos binarios que cumplen lo siguiente: el dígito en la posición i del mensaje codificado es un 0 si el dígito en la posición i de la cadena obtenida en el paso 6a es igual al dígito en la posición i de la cadena obtenida en el paso 6b. De lo contrario, el dígito en la posición i del mensaje codificado será un 1..

   Por ejemplo, si el mensaje a enviar es “hola” y la lista de números modificada empieza en [1,0,3,2], el mensaje codificado resultante es tal como se muestra en la siguiente figura.



7. Finalmente, el mensaje a enviar es la unión de la cadena aleatoria del paso 3 y el mensaje codificado del paso 6.



Habrás notado que el mensaje codificado no tiene ningún significado para alguien que no conoce la clave, por lo que si alguien intercepta lo que se envió, no puede obtener mucha información sobre el mensaje original.

Problema

En esta tarea deberás escribir el programa ***WhatsIntro***, que permite participar en un *servicio seguro de mensajería*, al estilo del popular ***WhatsApp***. En particular tu programa deberá permitir:

- Enviar mensajes (encriptados) a uno o más compañeros.
- Recibir y desencriptar los mensajes que te han enviado tus compañeros.
- Mostrar los mensajes en pantalla.
- Filtrar los mensajes que se muestran en la pantalla.

Librerías

Las librerías con las que cuentas para llevar a cabo tu tarea son las siguientes:

`1. whatsintro_msg`

Esta librería servirá para interactuar con el servidor que manejará el servicio de mensajería. Provee las siguientes funciones:

- `conectar(usuario, clave)`: Esta función debe ser llamada antes que cualquier otra función de la librería `mensajeria` para que la librería y el servidor sepan quien eres. Recibe como parámetros tu nombre de usuario y tu clave. Retorna `True` si se puede conectar correctamente con el servidor, y `False` en caso contrario.
- `mensaje_recibido(k)`: Retorna el k-ésimo mensaje que te han enviado tus compañeros. Cada mensaje esta compuesto por tres partes separadas por el caracter "`\n`" . La primer parte contiene la cuenta de quien lo envía, la segunda parte es la fecha y hora en que fue enviado, y la tercer parte es el mensaje encriptado. Si aún no has recibido el k-ésimo mensaje, la función retornara un mensaje sin las tres partes, esto es "`\n\n\n`"
- `mensaje_enviado(k)`: Retorna el k-ésimo mensaje que has enviado. Cada mensaje esta compuesto de la misma forma que el punto anterior.
- `enviar_mensaje(mensaje_encriptado)`: Envía el mensaje encriptado a aquella persona que ha sido señalada en el texto del mensaje. Para esto, cada uno de ustedes tendrá una cuenta del tipo @*cuentauc*. Por ejemplo, si Pablo Perez tiene la cuenta UC *pperez*@uc.cl, él tendrá la cuenta de mensajería @*pperez*. El servidor recibirá tu mensaje e intentará descifrar y enviarlo al destinatario, por lo cual el mensaje debe estar correctamente encriptado y debe tener un único destinatario válido. Esta función retorna dos valores, el primero indica si el mensaje pudo ser enviado (`True` o `False`) y el segundo es un mensaje que resume el resultado de la operación.
- `cantidad_recibidos()`: Retorna la cantidad de mensajes recibidos.
- `cantidad_enviados()`: Retorna la cantidad de mensajes enviados.

- `validar_encriptacion(a,b)`: Esta es una función de apoyo para validar tu encriptación. Recibe el mensaje original `a` y el mensaje encriptado `b`. La función retorna dos valores, el primero indica si la encriptación es correcta o no (`True` o `False`), el segundo es un mensaje que resume el resultado de la operación.

Habrás notado que en todo lo anterior hay algo extraño, y esto es que los mensajes que te envían tus compañeros son encriptados con sus claves, pero a ti te llegan encriptados con tu clave. ¿Cómo puede ser esto? Lo que ocurre es que el servidor al que se conecta a librería `whatsintro_msg` conoce todas las claves. Luego cuando envías un mensaje encriptado con tu clave, el servidor lo descifrará con tu clave y lo encriptará con la clave del destinatario. Te darás cuenta que esto nos permite a nosotros leer los mensajes originales, por lo que no es una buena idea enviar mensajes de mal gusto.
Para descifrar un mensaje puedes suponer que el mensaje recibido del servidor viene encriptado con tu propia clave y conociendo la clave y el algoritmo de encriptación puedes diseñar un algoritmo para descifrar los mensajes.

2. whatsintro_gui

Esta librería te permitirá manejar el programa a través de una interfaz gráfica. Al iniciar tu programa se abrirá una interfaz gráfica que tiene los elementos necesarios para que funcione el sistema de mensajería (ver Figura 1).

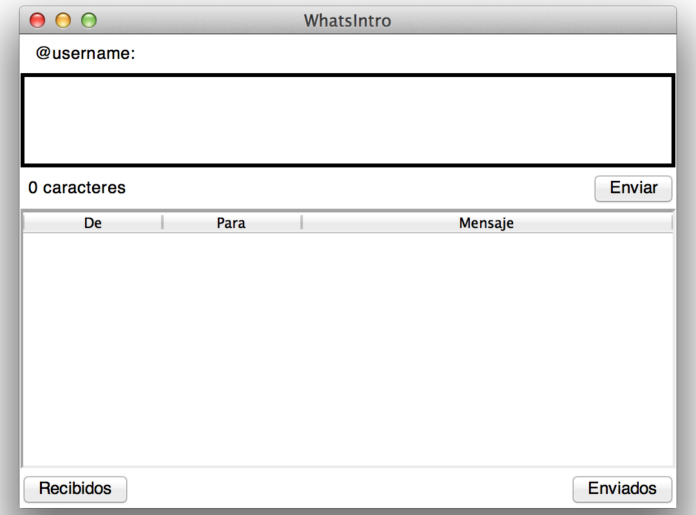


Figura 1. Ventana principal de la librería `whatsintro_gui`

Para que tu programa pueda ser manejado por una interfaz gráfica, la librería `whatsintro_gui` ofrece las siguientes instrucciones:

- `emisor(s)`: Cambia la cuenta del emisor en la ventana de la aplicación por la cadena de texto `s`.
- `mensaje_redactado()`: Retorna la cadena de texto que se encuentra en el cuadro de texto en el cual se escribe un mensaje.

- `borrar_mensaje_redactado()`: Borra la cadena de texto que se encuentra en el cuadro de texto en el cual se escribe un mensaje.
- `agregar_mensaje_al_final(de, para, msg)`: Inserta el mensaje `msg` al final de todos los mensajes que se muestran.
- `agregar_mensaje_al_inicio(de, para, msg)`: Inserta el mensaje `msg` al principio de todos los mensajes que se muestran.
- `esperar_click()`: Al llamar a esta instrucción el programa se detiene hasta que el usuario haga click en alguno de los botones de la interfaz. Luego retorna el nombre en minúsculas del botón que se presionó. Por ejemplo, si el usuario presionó el botón *Enviar* entonces esta función retornará la cadena de texto "enviar". Los títulos de las columnas de la lista de mensajes *De*, *Para* y *Enviar* también son botones, pero estos no serán implementados en esta tarea.
- `borrar_lista_mensajes()`: Elimina los mensajes de la interfaz gráfica, pero no del servidor.
- `salir()`: Cierra el programa.

Además de estas librerías, existen funciones propias del lenguaje Python que podrían ser de utilidad:

- `bin(x)`: Convierte un número entero `x` a una secuencia de dígitos binarios[2].
- `ord(c)`: Dado un caracter `c`, retorna el número entero que representa a este caracter en el formato ASCII [3].
- `chr(x)`: Es la función inversa de `ord`. Recibe como parámetro un número entero y retorna el caracter que este número representa en la codificación ASCII [4].

Configuración

Para que tu proyecto funcione correctamente, debes tener en una misma carpeta tu proyecto junto con los archivos que se encuentran junto a este enunciado en la página del curso:

- `whatsintro_msg.py`
- `whatsintro_gui.py`

Además, el archivo de tu proyecto debe cumplir con la plantilla que se muestra en el siguiente fragmento de código y que también podrás encontrar en la página del curso.

[2] https://docs.python.org/3/library/functions.html#bin
[3] https://docs.python.org/3/library/functions.html?#ord
[4] https://docs.python.org/3/library/functions.html?#chr

```
import whatsintro_gui
import whatsintro_msg

def tarea(tablero):
    #Aqui empieza tu tarea


app = whatsintro_gui.Application("tarea")
app.title('WhatsIntro')
app.loadProgram(tarea)
app.start()
```

Guarda tu proyecto con el nombre `tarea2_nroalumno_seccion_n.py` donde debes reemplazar **`nroalumno`** por tu número de alumno y **`n`** por el número de tu sección. Por ejemplo, Juan Pérez de la sección 1 con número de alumno 12345, guardará su proyecto con el nombre `tarea2_12345_seccion_1.py`.

Funcionamiento

WhatsIntro debe funcionar de la siguiente manera:

1. Al iniciar el programa, este deberá intentar conectarse automáticamente con el servidor. Si ocurre un error el programa termina mostrando un mensaje al usuario indicando el error.
2. Si el usuario logra establecer una conexión con el servidor, debe aparecer como *emisor* el nombre del usuario que está conectado.
3. Luego, el programa debe permitir al usuario hacer lo siguiente:
    a. Enviar el mensaje redactado al presionar el botón "Enviar". Si el envío fue exitoso el mensaje desaparecerá del cuadro de texto para que el usuario ingrese uno nuevo si así lo requiere. Si hubo un error en el envío del mensaje, el mensaje redactado permanecerá en el cuadro de texto para que sea editado.
    b. Mostrar el listado de mensajes recibidos o enviados cuando el usuario presione el botón "Recibidos" o "Enviados" respectivamente.
    c. Salir del programa cuando el usuario lo requiera cerrando la ventana de la aplicación.

Indicaciones Generales

(1) Recuerda que la tarea es individual, y que las copias serán sancionadas con nota **1.1** final en la asignatura.

(2) La entrega de esta tarea es a través del SIDING en la sección cuestionario.

(3) La fecha última de entrega es el lunes 29 de septiembre hasta las 23:59 PM. No esperes hasta última hora, pues el buzón (cuestionario) se cierra automáticamente a esa hora. No se aceptarán entregas atrasadas ni entregadas por otros medios.

(4) Revisa bien lo que entregas, aunque esta vez podrás subir tu tarea ilimitadas veces, la versión final a corregir será la última que hayas subido.