

**Ingeniería de Software II**

Trabajo Practico N°4

Carrera: Licenciatura en sistema de infromacion.

Catedra: Ingenieria de Software 2.

Docentes:Colla, Pedro.

Sanchez, Hernan.

Alumna: Landini, Magalí.

Año 2023

**TP4 RRR**

**Ingeniería Reversa, Re-factoría y Re-Ingeniería**

1. La instalación dispone de un programa llamado *getJason.pyc* legado de un sistema ya obsoleto y del cual no existen fuentes. El mismo permite recuperar el API token para acceder a microservicios bancarios del *Banco XXX* contenido en un archivo JSON. Se desea reusar el programa asegurando que puede recuperarse cualquier clave existente en el archivo JSON el que ahora contiene claves múltiples indicándola como argumento (siendo el default “token1”).

a. Instale los elementos contenidos en RRR\_TP4.tar.gz, en particular el programa getJason.pyc junto con el archivo JSON de prueba provisto y

la documentación existente. Utilice Python 2.7. Documente lo que

encuentre.

b. Ejecútelo, verifique la correspondencia entre lo que indica la

documentación y el comportamiento real. Realice brevemente pasos 1-

6 de la metodología.

c. Instale el paquete Python *uncompyle6*.

d. Ejecute *uncompyle6* y obtenga el código fuente getJason.py. Ejecútelo para verificar se comporta igual que *getJason.pyc*.

e. Identifique las razones para las diferencias encontradas en el punto

anterior entre el código obtenido y la documentación del mismo.

f. Modifique el programa getJason.py para que actúe como indica la

documentación y satisfaga los requerimientos de reuso. Valide y verifique su nuevo comportamiento.

g. Genere la nueva versión de getJason.pyc (python –m compileall

getJason.py) Verifique y valide. Provea una clase que dado un número entero cualquiera retorne el factorial del mismo, debe asegurarse que todas las clases que lo invoquen utilicen la misma instancia de clase.

1. Continuando con el programa getJason.pyc (y su versión fuente getJason.py) del ejercicio anterior se ha decidido que el programa tiene un código deficiente para lo crítico que resulta para la instalación.

a. Manteniendo las modificaciones realizadas en el trabajo práctico

anterior analice una intervención que permita aplicar programación orientada a objetos de forma que transforme al programa.

b. La transformación consistirá en reorganizar el código para utilizar un

patrón de diseño “Singleton”.

c. La clase resultante del punto anterior debe poder ser ejecutada desde línea de comandos, al efecto deberán mejorarse las condiciones de

chequeo de los parámetros externos de ejecución de forma de hacer a la ejecución más robusta. El objetivo primario es que el programa nunca termine con un error de sistema y siempre lo haga con un error del programa controlado.

d. Para hacer la convergencia entre el programa original y su versión re factorizada utilice una estrategia “*Branching by abstraction*”.

e. Aproveche a agregar comentarios al programa de su funcionamiento y una carátula donde se especifique es propiedad de la compañía

(“copyright IS2 © 2022,2023 todos los derechos reservados).

f. Haga una corrida de revisión con el analizador estático de código pylint y corrija las observaciones hasta que el mismo otorgue un puntaje de 8 o superior.

1. Continuando con el programa getJason.py previamente utilizado se usará ahora en el contexto de un cambio organizacional más profundo, donde resulta indicado aplicar una intervención de tipo re-ingeniería.

a. En el proceso de pagos anterior un empleado decide sobre que banco liberar un pago y elige un token respectivo para poderlo realizar.

b. En realidad el proceso de decisión se puede automatizar, bastará que exista saldo en la respectiva cuenta y que los pagos se hagan en forma

balanceada.

c. Partiendo del objeto singleton que dado un nombre de token (banco) da la clave integrarlo en un nuevo componente que ante una solicitud de

pago seleccione automáticamente la cuenta desde la que se hará el

mismo. La información de relación entre banco (token) y clave está en

sitedata.json

d. Se realizará una clase utilizando el patrón “cadena de comando” que

controle las dos cuentas (las correspondientes a “token1” y “token2”, ambas inicialmente con un saldo de $1000.- la correspondiente a “token1” y $2000.- la correspondiente a “token2”.

e. Se realizarán pedidos de pago de $500.- (número de pedido, monto) los que alternativamente se rutearán a cada cuenta que tenga saldo suficiente., para lo cual se indicará el número de pedido, el token a utilizar y el monto del pago realizado como salida de la clase. Prever una función de “listado” que muestre todos los pagos realizados por orden cronológico utilizar un patrón iterator al efecto)

1. Para poder ejecutarlo se uso la version 2.7 ya que con la version mas reciente el programa no ejecutaba.

Paso 1: colecccion de datos

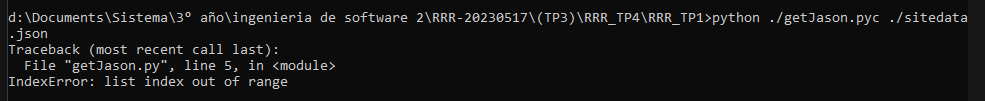
* Extractor de token para acceso API Servicios Banco XXX (versión 1.0)
* ¿Que hace el programa? Este programa permite extraer la clave de acceso API para utilizar los servicios delBanco XXX.
* Se posee de un archivo getJason.pyc que es el programa ejecutable.
* Para poder ejecutarlo se debe pasar el argumento sitedata.json
* A partir de standar output se obtiene el token
* Para obtener mensaje de ayuda se debe ejecutar ./getJason.pyc -h

Excepciones:Todas las condiciones de error del programa deben producir un mensaje de error bajo su control antes de

terminar.

* Al ejecutar el prgrama como indica la documentación el mismo falla tirando el error: IndexError: list index out of range
* Al ejecutar el -h no nos proporciona la ayuda.
* Se observan discrepancias entre lo que dice la documentacion y como funciona realmente el programa

Paso 2: Extraer infromacion

* La documentación disponible proporciona informacaòn imprecisa.
* Se dispone de los archivos necesarios para realizar la ejecución. Aun asi no funciona el programa.
* Falta mas documentación, falta los archivos fuentes para evaluar porque sucede ese error al ejecutarlo.
* No concuerda la forma documentada de como ejecutar el archivo pyc de como realmente debería ser, suponiendo que falta pasarle algún parámetro.
* Resultado de la ejecución: 
* AL pedir ayuda al programa “-h” el mismo no funcionaba generando error.

Paso 3: Extraer estructura.

* A partir del uncompyle6 se obtuvo el código fuente para validar el funcionamiento.
* Se observo que el programa esperaba una key como parámetro, la misma no estaba especificada en la documentación.
* La ayuda “-h” no estaba generada por lo mismo es que generaba error al ejecutarla.
* No tenia exepciones realizadas, algo que la documentacion nos indicaba.

Paso 4:registra funcionalidad

* El progrma funciona de la siguiente manera:

A partir de la modificaciones realizadas el programa sigue el siguiente funcionamiento:

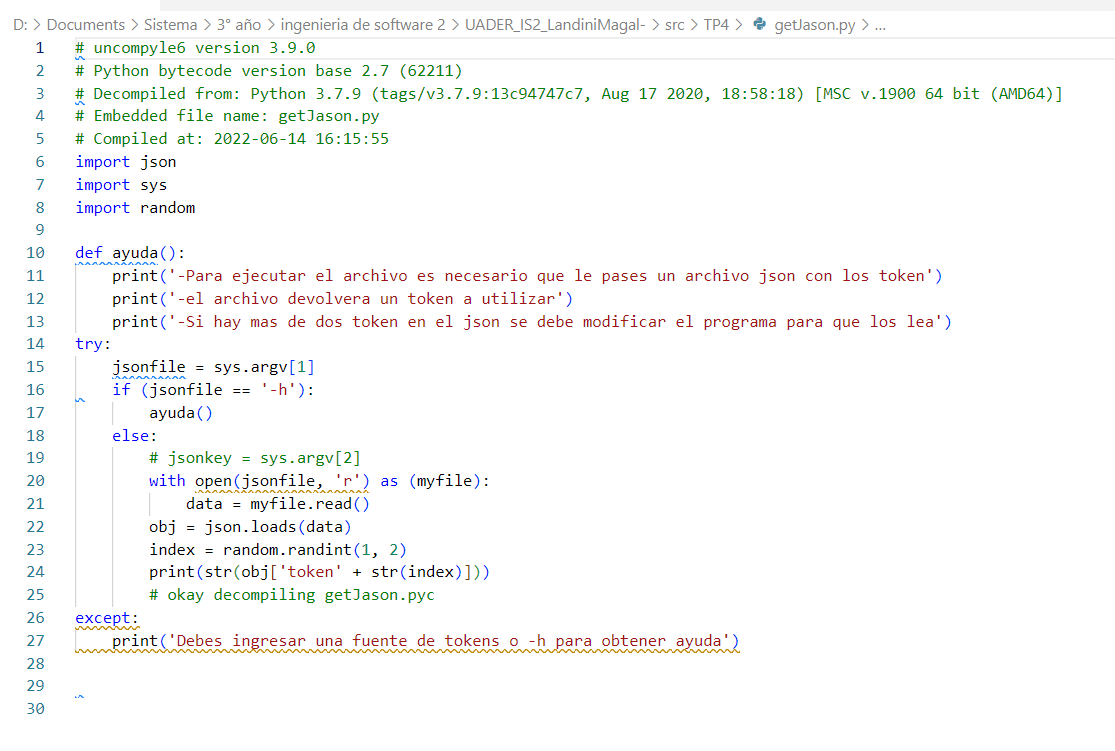
* + Si se le pasa como argumento -h: se ejcuta la ayuda al usuario
  + Si se le pasa el sitedata.json se ejecuta el programa devolviendo un token aleatorio contenido en el json.
  + En caso de que halla errores se se ejecuta una excepcion

Paso 5: registra el flujo de pasos

* Dependiendo el parámetro que se le pase al programa el mismo va a funcionar de cierta manera, o imprimirá la ayuda o ejecutara el resto del programa.
* La modificaciones que se le hizo fueron las necesarias para que este acorde a la documentación provista.

Paso 6: documentación

* Quedara el git el código fuente junto con el nuevo archivo ejecutable pyc.



Nuevo archivo pyc : <https://github.com/MagaliLandini/UADER_IS2_LandiniMagal-/blob/main/src/TP4/NewGetJason.pyc>

getJason.py: <https://github.com/MagaliLandini/UADER_IS2_LandiniMagal-/blob/main/src/TP4/getJason.py>

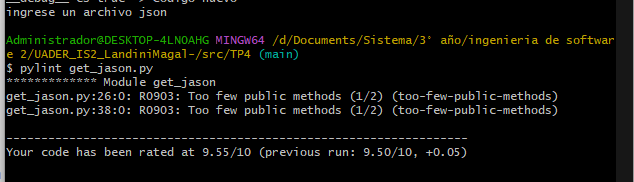
1:B) Al obtener el codigo fuente del archivo getJason.pyc se observo que el error se debía a que el archivo esperaba otro parámetro que es la key.

Debido a que la documentación no especificaba esto no se sabia que parámetro se le debía proporcionar.

La documentación no concordaba con el funcionamiento del programa y especificaba funciones que el mismo no poseia como la ayuda (-h)

1. Ejecucion de pylint

Una de las recomendaciones de pylint fue de cambiarle el nombre al archivo, por ese motivo paso a llamarse get\_jason.py



Archivo modificado de getJason.py ahora nombrado get\_jason.py: <https://github.com/MagaliLandini/UADER_IS2_LandiniMagal-/commit/1e263e43e20c12d8eb4cd4a432424ff83e7585eb>

1. Mejoras realizadas al archivo get\_jason.py: <https://github.com/MagaliLandini/UADER_IS2_LandiniMagal-/commit/24822cfbf1c8dc4e5acc7ef60a6362a620c59f5f>