Taller de introducción a la programación: Trabajo Practico 0

M. Sc. Saúl Calderón Ramírez
Instituto Tecnológico de Costa Rica,
Escuela de Computación.

15 de marzo de 2022

El presente trabajo práctico pretende introducir al estudiante en el proceso de análisis, diseño e implementación de un sistema de software, usando recursividad numérica y conceptos elementales de manipulación de archivos.

- **Modo de trabajo**: Parejas.
- **Fecha de entrega:** Martes 19 de abril del 2022.
- **Tipo de entrega:** Entrega del archivo de documentación externa y el código fuente en .py, por medio de la plataforma TEC-digital.

Parte I
Descripción del problema

En la presente Tarea Programada, usted desarrollará un programa para un sistema de cajero automático. Este sistema contará con un conjunto de usuarios definidos previamente mediante el sistema bancario, el cual usted también implementará. Por cada usuario se almacenará en el sistema de cajeros (**no puede usar funciones de la clase str**):

- El nombre de usuario, el cual debe cumplir la expresión regular $[(a-z)|(A-Z)]^{+}[0-9][0-9][0-9][0-9][!|\$|\#|\&|?]$, por ejemplo la clave saul1918! es válida según esta expresión regular. Se debe verificar que el nombre de usuario no haya sido tomado (**utilice la codificación unicode, y recursividad simple para implementarlo**).
- La clave secreta, la cual debe ser únicamente de cuatro digitos y será inicializada aleatoriamente.
- El monto disponible en la cuenta.
- El historial de transacciones (retiros o depósitos), hechos a la cuenta del cliente.

Para el cajero se deben almacenar los siguientes atributos (**no puede usar funciones de la clase str**):

- El identificador del cajero debe cumplir la expresión regular $[(A-Z)(A-Z)(A-Z)][0-9]^{+}$, y el identificador también debe ser único (**utilice la codificación unicode, y recursividad simple para implementarlo**).
- La cantidad de billetes disponible por cada denominación.

En un cajero específico, podrán realizarse retiros o depósitos de dinero, ambos con billetes de denominaciones de 1, 2, 5, 10, 20, 50 y 100 pesos. Cuando se realice un retiro de la cuenta de un cliente específico, el cajero debe:

1. Definir la mínima cantidad de billetes a entregar, especificando la cantidad de billetes por denominación.

2. Verificar si existen los billetes requeridos de cada denominación en el cajero, si no existe la cantidad de billetes debe notificar al usuario y abortar el débito.

3. Si existen los billetes requeridos por cada denominación en el cajero, realizar el débito y notificar al usuario la transacción exitosa. Si la transacción no se puede realizar, no se ingresa en el historial.

4. En la información del usuario, debe ingresarse la entrada de un débito (actualizando el historial), además de actualizar el saldo.

Para el caso de un depósito al cajero, se sucederán las siguientes acciones:

1. El usuario especificará la cantidad de billetes a depositar por denominación.

2. Actualizar el monto en la cuenta del usuario, además de la cantidad de billetes disponible en el cajero.

3. Notificar al usuario la transacción exitosa.

El cajero además permitirá desplegar el monto disponible y el historial de transacciones en pantalla.

Parte II
Requerimientos del programa

El programa debe ser desarrollado en Python, basado en la descripción del problema anterior, debe cumplir los siguientes requerimientos funcionales:

1. Permitir a un ente interno del banco (banquero) crear cajeros y usuarios, mediante un sistema en el que se ingresen esos datos por consola.

   a) Para la creación del usuario, el banquero ingresará el nombre del usuario, y el sistema le generará aleatoriamente un pin de 4 dígitos, el cual mostrará en pantalla. Además guardará en un archivo de extensión *.txt* con el nombre del usuario y tal pin generado.

   b) Para la creación de cajeros, el banquero ingresará el identificador y el monto de billetes de cada denominación que contendrá. Además el banquero podrá agregar billetes en cada denominación a un cajero existente.

2. Para un usuario del banco, se permitirá el uso del sistema de cajeros, el cual permitirá al usuario:

   a) Escoger el cajero donde desea hacer la transacción. Una vez escogido el cajero, el usuario podrá:

      1) Realizar un retiro de una cantidad especifica de pesos, para lo cual se debe seguir el procedimiento descrito en la sección anterior.
      2) Consultar el monto disponible en su cuenta, el cual debe desplegar en pantalla.
      3) Consultar el historial de transacciones, el cual debe desplegar en pantalla, en forma tabulada. Se debe desplegar el monto (si fue depósito o retiro, con +/-), la fecha y hora, además del cajero donde se hizo la transacción.

Los datos del cajero y el usuario deberán ser almacenados en uno o varios archivo de formato *csv, coma separated values*, el cual separa los valores entre columnas con un punto y coma (;), y las filas con un cambio de línea. En el diseño debe decidir como representar los datos estipulados en un archivo.

Finalmente, además del código fuente del programa, es requisito entregar la documentación pertinente del programa. El formato del mismo será previamente especificado, posiblemente usando la plataforma LaTeX (mediante el programa Lyx).

No se permite tomar código de la red para implementar las funcionalidades principales o llamar a librerías externas para leer los archivos *.csv*. Debe usar la funcionalidad nativa para leer y escribir archivos `http://www.pythonforbeginners.com/files/reading-and-writing-files-in-python`.

1. Implementación

Para construír este y los demas proyectos, es necesario utilizar el siguiente conjunto de herramientas:

1. Star UML, para la diagramación de los casos de uso del sistema.
2. Lyx junto con LaTeX para la elaborar la documentación externa.

Para la implementación sólo se puede usar recursividad, y no estructras de control iterativas.

2. Evaluación

El siguiente corresponde al esquema de evaluación:

1. Implementación (70 %)
    a) Correcto funcionamiento en las pruebas funcionales (30 %). Se estipularán las pruebas a realizar para la tarea programada en la plataforma.
    b) Uso correcto de las prácticas recomendadas de programación en el curso: documentación interna, variables y métodos signiticativos, etc. (30 %).
    c) Modularización apropiada de los procedimientos (10 %).
2. Documentación externa (30 %)
    a) Seguimiento de la estructura propuesta (15 %).
    b) Correcta redacción, uso de figuras y citas bibliográficas (15 %)