Materia: Programación Orientada a Objetos

Profa. Karen Azurim Gamboa

Proyecto Final

Banco POO

Daniela Vignau A01021698

Rubén Sánchez A01021759

Carlos García A01025948

Randy Hazael Jiménez A01251333

Gerardo Anglada A01021917

Campus Santa Fe

10 de mayo 2018

MANUAL DE USUARIO

Introducción

Este programa fue hecho en lenguaje Java, por lo tanto puede ejecutarse en cualquier sistema operativo que tenga instalada la Máquina Virtual Java, disponible en la dirección web http://www.java.com o http://www.java.com/es en español. Para tener una mejor experiencia del programa, se recomienda tener el *command prompt* (Windows)/terminal(macOS) en pantalla completa. Se debe de compilar y ejecutar dentro del *command prompt*/terminal del sistema donde se vaya a probar para poder comenzar a utilizar el programa.

El programa es un sistema de banco, donde de acuerdo a tu usuario (Administrador o Cliente) se te permiten hacer distintas operaciones. Por ejemplo el administrador tiene la posibilidad de crear un cliente donde se le piden distintos datos personales, posteriormente se le da la posibilidad de otorgarle una tarjeta de crédito. El usuario administrador también puede modificar los datos personales, borrar a un cliente e imprimir los datos de los clientes. Por otra parte, los usuarios cliente pueden hacer distintas operaciones como realizar un retiro, una compra y pagar su tarjeta. Todos los datos son guardados en un archivo CSV para que la siguiente vez que se ejecute el programa, los datos introducidos la vez anterior sigan existiendo.

Requisitos del programa

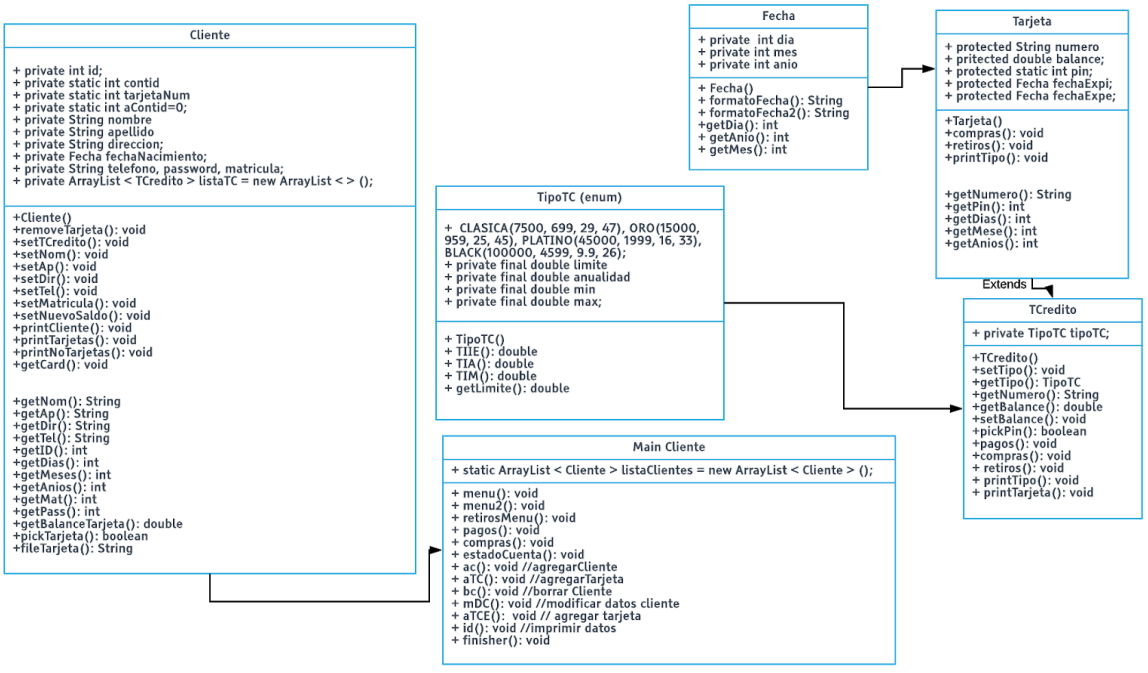
Para el correcto funcionamiento del programa, es necesario tener cubiertos los siguientes requisitos:

1. Tener instalado y actualizado:
    - *Java Development Kit*, JDK por sus siglas, para poder compilar el programa. Es necesario tener el compilador funcionando, pues es el encargado de convertir el código fuente (.java) en bytecode (.class), el cuál será interpretado por la Máquina Virtual de Java, JVM.
    - *Java Runtime Environment*, JRE, que permite la implementación de la JVM que se dedica a ejecutar los programas. Normalmente este se instala de manera simultánea al instalar el JDK.

CARACTERÍSTICAS PROPUESTAS

- El usuario cliente no podrá realizar las mismas actividades que el usuario administrador, por lo tanto se pedirá un usuario y contraseña para poder diferenciar entre uno u otro.
- El usuario cliente podrá realizar retiros, compras y pagar su tarjeta de crédito.
- El usuario administrador podrá dar de alta/baja a un cliente, modificar los datos personales del mismo, agregar una tarjeta al cliente e imprimir los datos de un cliente en específico.
- En los usuarios clientes, cada tarjeta contará con un NIP que será un requisito para poder realizar las distintas acciones.
- La información de cada cliente y sus tarjetas estarán estará guardada en un archivo de texto. El código leerá el archivo de texto antes de hacer cualquier otra cosa y, al final, agarrará todos los cambios y los pondrá de regreso en el archivo de texto.

DIAGRAMA DE DISEÑO DE CLASES



Programa

El programa está compuesto por seis distintas clases, una de ellas siendo el Main, que conforman el funcionamiento del programa como se requiera.

1. Main: es donde se va a ejecutar el programa como tal. Incluye 12 métodos, los cuales serán explicados con mayor detalle más adelante.
2. Cliente: es donde se crea un cliente, contiene métodos para que se le asigne una tarjeta, una dirección, teléfono y otras cosas.
3. Tarjeta: Clase padre para la tarjeta de crédito. Tiene un número de tarjeta, pin, fecha de expedición, entre otras cosas.
4. TCredito: se asigna una tarjeta de crédito, se le asigna un saldo, un número de tarjeta. Contiene los métodos para realizar compras, retiros y pagos.
5. TipoTC (enum): como las tarjetas serán siempre las mismas se usa una clase enum para calcular el TIM, TIIE y TIA dependiendo del tipo de tarjeta de crédito que sea. Hay 4 tipos de tarjeta.
6. Fecha: Tiene como propósito darle formato a la fecha.

Menú Inicial

La interfaz del programa le presenta al usuario (cliente o admin) un menú inicial, tal como se muestra en la Figura 1. Se le da la posibilidad de seleccionar distintas opciones, éstas las presentamos en la Tabla 1. Las opciones deben ser seleccionadas a partir de la numeración, tiene un try/catch que sirve para evitar que el programa se cierre en caso de que el usuario meta un caracter y no un número como se pide.

Figura 1. Menú Inicial

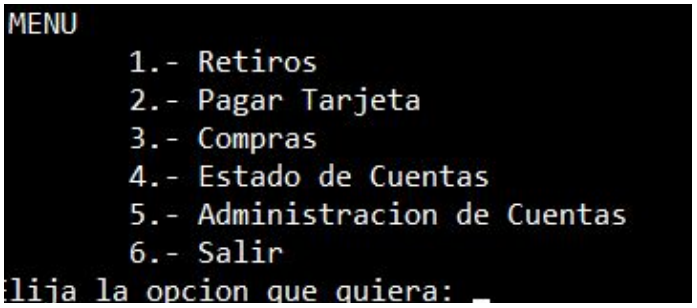


Tabla 1. Funciones del menú

OPCIÓN		FUNCIONALIDAD
1	Retiros	Le permite al usuario cliente realizar un retiro desde la tarjeta que él desee.
2	Pagar Tarjeta	Le permite al usuario cliente pagar su tarjeta.
3	Compras	Le permite al usuario cliente realizar compras.
4	Estado de Cuentas	Le permite al usuario que haya ingresado su ID ver el estado actual de todas sus cuentas.
5	Administración de Cuentas	Le permite únicamente al administrador manipular las cuentas del banco. Se abre otro menú que despliega más opciones.
6	Salir	Para terminar el programa

1. Retiros

Imprime una instrucción al usuario pidiéndole que ingrese su ID asignado, el número de tarjeta y el NIP para la misma. Por último se solicita la cantidad a retirar. La cantidad a retirar será reducida inmediatamente del saldo de la tarjeta. En caso de no haber ID que coincida con algún cliente registrado, se regresará al usuario al menú principal.

2. Pagar Tarjeta

Le pide al usuario el ID, el número de tarjeta y el NIP de la misma para que finalmente el usuario ingrese la cantidad de dinero que pagará de la tarjeta. Si el monto ingresado es mayor que lo que se ha gastado, no se lleva a cabo, si es igual o menor, se hace la resta y esto permite que la tarjeta puede seguir siendo utilizada si ya había llegado a su límite de crédito.

3. Compras

Le pide al usuario el ID, el número de tarjeta y el NIP de la misma para que finalmente el usuario meta la cantidad que se debe de cobrar de la tarjeta. El monto ingresado será reducido inmediatamente del saldo de la tarjeta.

4. Estado de Cuentas:

Le pide al usuario el ID, el número de tarjeta y el NIP de la misma para posteriormente imprimir el estado actual de la cuenta correspondiente.

5. Administración de Cuentas

Permite al usuario agregar/borrar un cliente, modificar los datos personales, agregar una tarjeta de crédito al cliente deseado. Dado a que son tantas las opciones, se le presenta un nuevo menú: Menú de Administración de Cuentas.

Menú de Administración de Cuentas

Cuando el usuario administrador, opta por seleccionar la opción de “Administración de Cuentas” en el Menú Inicial, se le presenta un nuevo menú (ver Figura 2) donde se muestran las distintas opciones que el administrador puede realizar; las opciones se presentan en la Tabla 2 y deben de ser seleccionadas a partir de las letras enlistadas; cuenta con un try/catch para evitar que el programa se cierre en caso de que el usuario meta un caracter que no está dentro de las opciones que se le presentan

Figura 2. Menú de Administración de Cuentas

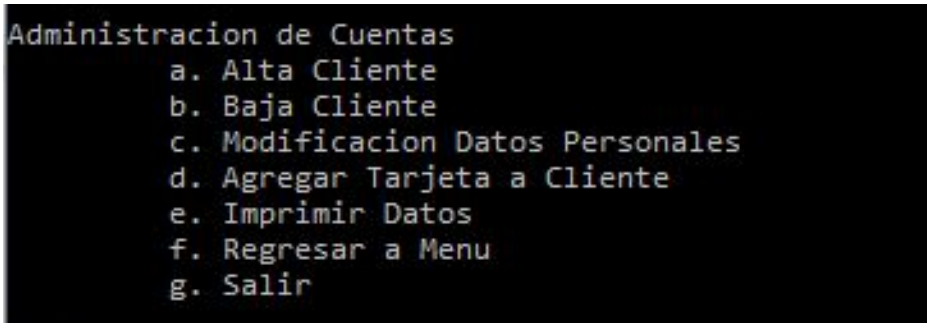


Tabla 2. Funciones del menú de Administración de Cuentas

OPCIÓN		FUNCIONALIDAD
a	Alta Cliente	El usuario tiene la posibilidad de dar de alta a un nuevo cliente.

b	Baja Cliente	El usuario tiene la posibilidad de dar de baja a un cliente.
c	Modificación de Datos Personales	El usuario tiene la posibilidad de modificar los datos personales del cliente.
d	Cambiar Tipo TC	El usuario puede cambiar el tipo de tarjeta de crédito de un cliente.
e	Agregar Tarjeta a Cliente	El usuario tiene la posibilidad de agregar una tarjeta a un cliente ya existente.
f	Imprimir Datos	Imprime los todos los datos del cliente.
g	Regresar al menú	Regresa al menú inicial
h	Salir	Para terminar el programa

a. Alta Cliente

Le pide al usuario distintos datos personales para poder crear un cliente. Posteriormente se le pregunta si quiere agregar una tarjeta de crédito, si es así se ejecuta el método “aTC()”.

b. Baja Cliente

A partir del ID del cliente, se le permite al usuario dar de baja a un cliente. Y se elimina todo rastro de él.

c. Modificar los Datos Personales

Permite al usuario cambiar los datos personales del cliente seleccionado. Se le presenta un menú sobre las distintas cosas que puede modificar y partir de la selección se hace el cambio pertinente.

d. Cambiar tipo TC

Permite al usuario cambiar el tipo de tarjeta de crédito, borrando la tarjeta anterior que se tenía y creando una nueva.

e. Agregar Tarjeta a Cliente

Le presenta al usuario las distintas tarjetas de crédito que tiene el banco, el usuario selecciona una, se genera un número aleatorio de la tarjeta de crédito, pide la fecha de expedición, el balance y finalmente a partir del tipo de tarjeta que seleccionó el usuario en un inicio, se le asigna la tarjeta de crédito al usuario seleccionado.

f. Imprimir Datos

Le pregunta al usuario el ID de la persona de la que quiere imprimir sus datos e imprime los datos que tiene el banco sobre su cliente.

g. Regresar al Menú

Regresa al usuario al menú inicial.

Conclusión del Proyecto

Podemos decir que el proyecto fue concluido satisfactoriamente. Aparte de un pequeño detalle, todo lo demás cumple con las propuestas y el programa está muy completo.

- El usuario cliente no puede realizar las mismas actividades que el usuario administrador. Se pide un usuario y contraseña para poder diferenciar entre uno u otro.
- El usuario cliente si puede realizar retiros, compras y pagar su tarjeta de crédito.
- El usuario administrador si puede dar de alta/baja a un cliente, modificar los datos personales del mismo, agregar una tarjeta al cliente e imprimir los datos de un cliente en específico.
- Para los clientes, cada tarjeta si tiene un NIP que es un requisito para poder realizar las distintas acciones.
- La información de cada cliente y sus tarjetas sí son leídas desde un archivo de texto. El único detalle es que, a la hora de regresar los datos al archivo de texto, sólo se registra una tarjeta por cliente..

REFLEXIONES

Daniela: Me pareció que en un inicio no escogimos una buena manera para trabajar el código, porque cada quién lo empezó a trabajar por separado creando así, un verdadero problema juntar todos los códigos. Una vez que nos pusimos mejor de acuerdo sobre cómo íbamos a trabajar el código, todo fluyó mucho más fácil, pues ya sólo existía un código “master” que nos rotábamos por día. Otra cosa que considero nos faltó, fue sentarnos un día todos pare decidir qué características queríamos implementar. Siento que éste programa fue una buena manera de poner a prueba nuestros conocimientos.

Rubén: No fue fácil realizar este proyecto. Conforme íbamos agregando más código al programa, nos fuímos dando cuenta de que algunas cosas eran más difíciles de programar de lo que pensábamos y ciertos problemas requirieron un gran uso de nuestro razonamiento para ser resueltos. Otro problema fue que, a la hora de querer meter nuevas cosas al código, nos dábamos cuenta de que teníamos que tener en consideración el resto del código y nos aparecieron muchos errores porque partes nuevas no eran compatibles con las viejas. En conclusión, para mejorar nuestro rendimiento y, en consecuencia, y el producto final en proyectos posteriores, debemos de tener más reuniones de equipo para aclarar qué queremos tener al final. Estuvo bien que nos hayamos dividido por día la elaboración del proyecto, pero, para que esto funcione, debemos que evitar que los errores permanezcan sin ser corregidos. Antes de que un miembro del equipo le pase el código al siguiente, este debe de concentrarse en corregir todos los errores existentes para que no sean un problema después. Así es como podemos seguir mejorando.

Carlos: Los problemas que surgieron en la elaboración del proyecto fueron gracias a que conforme pasaba el tiempo nos íbamos dando cuenta que le podíamos implementar más funcionalidades a este y por lo tanto intentamos extender nuestro proyecto en vez de terminar la base de este. Anterior a la entrega nos juntamos para concatenar los códigos y terminar primordialmente la funcionalidad básica y de

segunda mano las funcionales que consideramos adicionales. Pese a estas circunstancias considero que la elaboración del proyecto fue fundamental para poner en práctica los conocimientos adquiridos en el curso, aunque considero que lecciones o prácticas en trabajo colaborativo hubieran sido de gran ayuda para esta entrega.

Randy: La parte en la que, a mi parecer, tuvimos más dificultades fue en el implementar una manera de definir el tipo de tarjeta que el cliente fuese a solicitar ya que el código lo empezamos pensando en la tarjeta como un solo tipo de objeto con saldo fijo. De igual manera, una parte del trabajo que no fue de nuestros mayores fuertes fue la organización; a veces ocurrían confusiones acerca de qué hacía una parte del código que alguien había puesto. Por otro lado, la parte que hicimos mejor fue la funcionalidad en general de las interfaces, las metodologías de los movimientos bancarios y la escritura de datos en archivos.

Gerardo: Lo que se nos complicó acercándonos a la fecha de entrega del proyecto fue el hecho de que seguíamos agregando más funcionalidades, sin previamente tener completas las anteriores. Esto nos llevó a tener un código con más de 1300 líneas, con una funcionalidad del 80%, forzándonos a eliminar funciones con el propósito de mejorar el código fundamental para el funcionamiento correcto de lo requerido. Esto ocurrió por la falta de conocimientos en interfaces como github o TeleType, que nos permiten colaborar en tiempo real la edición del código.