|  |  |
| --- | --- |
|  | **FACULTAD DE CIENCIAS NATURALES Y EXACTAS**  **DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN**  **PRÁCTICA LABORAL INVESTIGATIVA I**  **PRIMER AÑO LICENCIATURA EN CIENCIA DE LA COMPUTACIÓN**  **Título del trabajo**  Sistema Gestor de Cuentas Bancarias  **Autor**: Alejandro Fonseca Cuza  **Tutor**: MSc. Magali Matilla Belett  **Diciembre de 2022** |

**RESUMEN**

En el presente trabajo se realizó el diseño e implantación de un Sistema Gestor de Cuentas Bancarias. Se implementaron las principales operaciones, como son: inserción, actualizaciones, cierres, registros de transacciones, y consultas sobre las cuentas y operaciones realizadas. El programa consta de una interfaz gráfica creada con el IDE QTCreator y programada con el lenguaje de programación C++.El programa da los resultado a las problemáticas que presenta un banco para administrar las cuentas bancarias, con seguridad, rapidez.

**ABSTRACT**

.

In the present work, the design and implementation of a Bank Account Management System was carried out. The main operations were implemented, such as: insertion, updates, closing accounts, transaction records, and inquiries about the accounts and operations carried out. The program consists of a graphical interface created with the IDE QTCreator and programmed with the C++ programming language. The program gives the results to the problems that a bank presents to manage bank accounts, safely, quickly.

Contenido

[1. Introducción 5](#_Toc122366186)

[1.1 Enunciado del problema a resolver. 5](#_Toc122366187)

[1.2 Análisis de la solución del problema. 6](#_Toc122366188)

[1.3 Objetivos del trabajo. 6](#_Toc122366189)

[2. Desarrollo 7](#_Toc122366190)

[2.1 Modelos matemáticos de la solución. 7](#_Toc122366191)

[2.2 Diagrama UML de la(s) clase(s). 8](#_Toc122366192)

[2.3 Descripción de los algoritmos elaborados. 9](#_Toc122366193)

[2.4 Implementación del sistema 11](#_Toc122366194)

[2.4.1 Requerimientos de Hardware y Software usados 11](#_Toc122366195)

[2.4.1 Interfaces del sistema 11](#_Toc122366196)

[3. Conclusiones 14](#_Toc122366197)

[4. Recomendaciones 15](#_Toc122366198)

[5. Bibliografía 17](#_Toc122366199)

# Introducción

La automatización de la banca cubana ha sido una prioridad desde el año 1995. Se ha trabajado en diferentes acciones con significativos aportes al  sistema bancarios,  como fueron en un principio el ensamble de computadoras, la creación de sistemas automatizados para la gestión contable bancaria, la introducción de los cajeros, entre otros. Hoy contar con aplicaciones que controlen todas las operaciones de las cuentas bancarias es la actividad fundamental que se ha automatizado en todos los sistemas bancarios.

## 1.1 Enunciado del problema a resolver.

Se desea mantener un registro de cuentas bancarias. De cada cuenta se tiene como atributos: número de cuenta, CI del cliente, nombre del cliente, fecha de creación , saldo actual y el interés anual que se aplica a la cuenta (porcentaje). Además, para cada cuenta se mantienen las transacciones que se han realizado en el mes. Se definen en los siguientes métodos:

* **Creación de una cuenta**, con saldo e interés inicial.
* **Cierre de una cuenta**, dado su código y CI del cliente.
* Dado el número de cuenta, **devolver su saldo actual**.
* Dado el número de cuenta, **actualizará el saldo de la cuenta aplicándole el interés diario** (interés anual dividido entre 365 aplicado al saldo actual).
* **Depositar una cantidad de dinero** en una cuenta dado su número, actualizando el saldo.
* **Extraer una cantidad de la cuenta** (si hay saldo), actualizando el saldo.
* **Mostrar todos los datos** de una cuenta.
* **Mostrar todas las transacciones realizadas** por la agencia bancaria en el último mes, clasificadas por depósitos y extracciones.
* El número de cuenta se asignará de forma correlativa a partir de 100001, asignando el siguiente número al último asignado.

## 1.2 Análisis de la solución del problema.

Teniendo en cuenta el problema a resolver podemos llegar a la pregunta ¿Cómo puede un sistema informático resolver la problemática que presentan los bancos a la hora de administrar sus cuentas?

Se diseñaron 3 clases: **Cuentas, Operaciones y Banco**.

* **Clase Cuentas:**

Atributos: (nombre, carnet de identidad, número de cuenta, saldo, interés anual,fecha ).

Métodos presenta el constructor y destructor de la clase.

* **Clase Operaciones:** instanciará todas las operaciones realizadas en las cuentas bancarias, llevando un registro según su tipo (1. Depósito ó 2. Extracción).Este registro se guardará en un QList para luego ser mostrado en el programa.
* **Clase Banco:** se encargará de administrar las instancias de la clase Cuentas, teniendo las funciones de: crearlas, borrarlas,depositar o extraer dinero, mostrar sus datos, actualizarlas a diario según su interés anual y mostrar las últimas operaciones hechas con las cuentas.

## 1.3 Objetivos del trabajo.

*Diseñar e implementar un sistema informático para las operaciones básicas de las cuentas bancarias en el BPA, garantizando mayor rapidez en las consultas realizadas por los usuarios del sistema.*

Para alcanzar el objetivo general, nos trazamos las tareas de investigación siguientes:

* Análisis y diseño del problema
* Implementación del sistema
* Prueba y análisis de los resultados del sistema

# Desarrollo

A continuación se detallan aspectos de diseño e implementación, así como el análisis de resultados del sistema, que simula las operaciones básicas realizadas en un banco a las cuentas de los clientes.

## 2.1 Modelos matemáticos de la solución.

**Modelo para el Deposito de Dinero:**

saldo\_resultante=saldo\_base+saldo\_depositado;

**Modelo para la Extracción de Dinero:**

Primero se debe comprobar que el saldo que quieren extraer no es mayor que el saldo que tiene la cuenta.

If(saldo\_base>=saldo\_extraido){

saldo\_resultante=saldo\_base-saldo\_extraido;

}

**Modelo para Actualizar Saldo (interés diario):**

Actualizar saldo, en este caso se suma el interés diario al saldo de la cuenta

Paso 1. Se divide entre 365 el interes\_anual para obtener el interes\_diario

interes\_diario=interes\_anual/365;

Paso 2. Se multiplica el saldo de la cuenta por el interes\_diario para saber que parte conforma el ese interes\_diario(%) de el saldo

parte=saldo\*interes\_diario;

Paso 3. Como último se le suma esa parte al saldo para tener el saldo actualizado;

saldo=saldo+parte;

## 2.2 Diagrama UML de la(s) clase(s).

Los diagramas UML, de las clases implementadas se muestran a continuación.

|  |
| --- |
| **Cuentas** |
| -numero\_de\_cuenta:long  -nombre\_del\_cliente:QString  -carnet\_de\_identidad:QString  -fecha\_de\_creacion:QDate  -interes\_anual:int  -saldo:float |
| +<<Constructor>>(...)  +<<Destructor>> |

|  |
| --- |
| **Operaciones** |
| -numero\_de\_cuenta:long  - identificador (1.Deposito - 2.Extraccion):int  -nombre\_del\_cliente:QString  -fecha\_de\_la\_operacion:QDate  - saldo\_operacion: float;  -saldo\_anterior: float;  -saldo\_resultante: float; |
| +<<Constructor>>(...)  +<<Destructor>> |

|  |
| --- |
| **Banco** |
| -Listado\_de\_cuentas:QList<Cuentas>  -Listado\_de\_operaciones:QList<Operaciones> |
| + <<Constructor>> ()  +<<Destructor>>  +crear\_cuenta(Cuentas):bool  +eliminar\_cuenta(int):void  +saldo\_actual(int,QString):void  +actualizar\_saldo(int):void  +depositar\_saldo(int,float,QDate,int):bool  +extraer\_saldo(int,float,QDate,int):int  +mostrar\_datos(int,QString,QString):void  +mostrar\_transacciones(QDate&):QList<Operaciones>  +buscar\_en\_cuenta(QString ,QString.int):bool |

## 2.3 Descripción de los algoritmos elaborados.

En el epígrafe se muestran los algoritmos más significativos del programa.

* **Algoritmo de búsqueda de cuenta dentro del listado de cuentas.**

bool buscar\_cuenta(cadena código, cadena carnet,entero &index){

Para (desde i=0 hasta listado\_de\_cuenta.tamaño paso 1)

Si(listado\_de\_cuenta[i].get\_carnet()==carnet y listado\_de\_cuenta[i].get\_codigo()==código)

index=i

retorna verdadero

fin\_si.

fin\_Para.

retorna falso

Fin\_buscar\_cuenta.

* **Algoritmo de crear cuenta.**

bool crear\_cuenta(Cuenta new\_cuenta){

Para (desde i=0 hasta listado\_de\_cuenta.tamaño paso 1)

Si(listado\_de\_cuenta[i].get\_carnet()==new\_cuenta.get\_carnet())

listado\_de\_cuenta[i]=new\_cuenta.

retorna verdadero

Fin\_Si

Fin\_Para

listado\_de\_cuenta.push\_back(new\_cuenta)

retorna falso

Fin\_crear\_cuenta

* **Algoritmo de borrar cuenta .**

voideliminar\_cuenta(entero index)

listado\_de\_cuentas.erase(listado\_de\_cuenta.begin()+index)

Fin\_eliminar\_cuenta

* **Algoritmo de actualizar cuenta .**

void actualizar\_cuenta(entero index)

decimalinteres\_d<-listado\_de\_cuenta[index].obtener\_saldo()/365

decimal parte <- listado\_de\_cuenta[index].obtener\_saldo()\*interes\_d

decimal total<- listado\_de\_cuenta[index].obtener\_saldo()+parte

listado\_de\_cuenta[index].establecer\_saldo(total)

Fin\_actualizar\_cuenta

## 2.4 Implementación del sistema

### 2.4.1 Requerimientos de Hardware y Software usados

* IDE usado: QTCreator v4.8
* Lenguaje de programación utilizado++
* Sistema Operativo: Windows
* Requerimiento mínimo: Ram:2 GB, Espacio de Disco:1GB

QTCreator es un IDE multiplataforma programado en C++, JavaScript y QML creado por Trolltech el cual es parte de SDK para el desarrollo de aplicaciones con Interfaces Gráficas de Usuario (GUI) con la bibliotecas de QT.

### 2.4.1 Interfaces del sistema

Pasos para la instalación del programa. Este proceso solo se hace la primera vez que ejecute el programa.

Paso 1.Dar Clic en Siguiente en el asistente de instalación (ver figura 1).

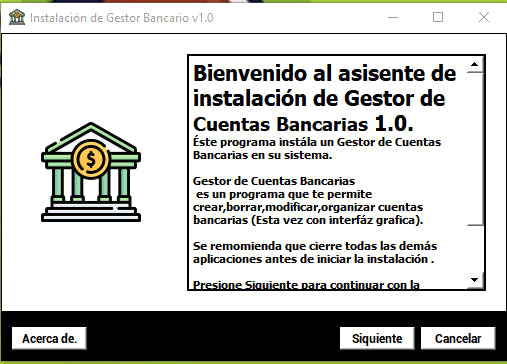


Figura 1. Interfaz de instalación

Paso 2. Leer y aceptar Términos y condiciones, luego presionar el botón Instalar (ver figura 2).

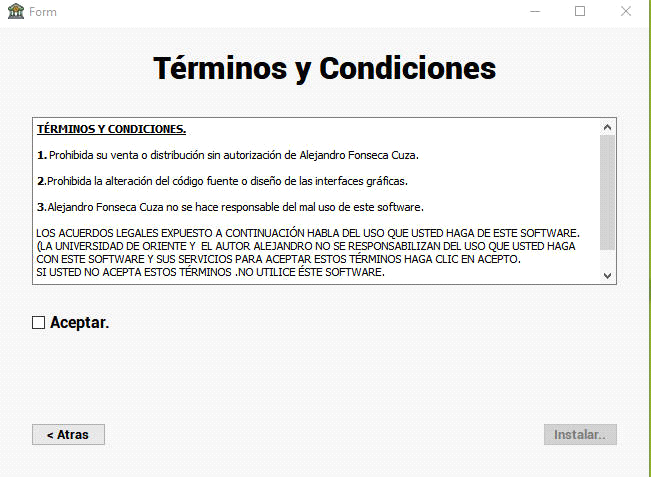


Figura 2. Interfaz de aceptación

Paso 3.La figura 3, muestra la interfaz con las operaciones básicas relacionadas con las cuentas bancarias. (Ver figura 3).



Figura 3, Menú Principal del sistema

A continuación se mostraran instantáneas de pantalla de interfaces en las diferentes operaciones del programa.

* Interfaz Crear Cuenta. (Ver figura 4).



Figura 4, Crear Cuenta

* Interfaz Cerrar cuenta, actualizar cuenta, saldo actual y mostrar saldo (Ver figura 5).



Figura 5.

* Las interfaz de Depositar y Extraer saldo, (Ver figura 6).



Figura 6, Depositar y Extraer Saldo

* Las interfaz de Transacciones (Ver figura 7).

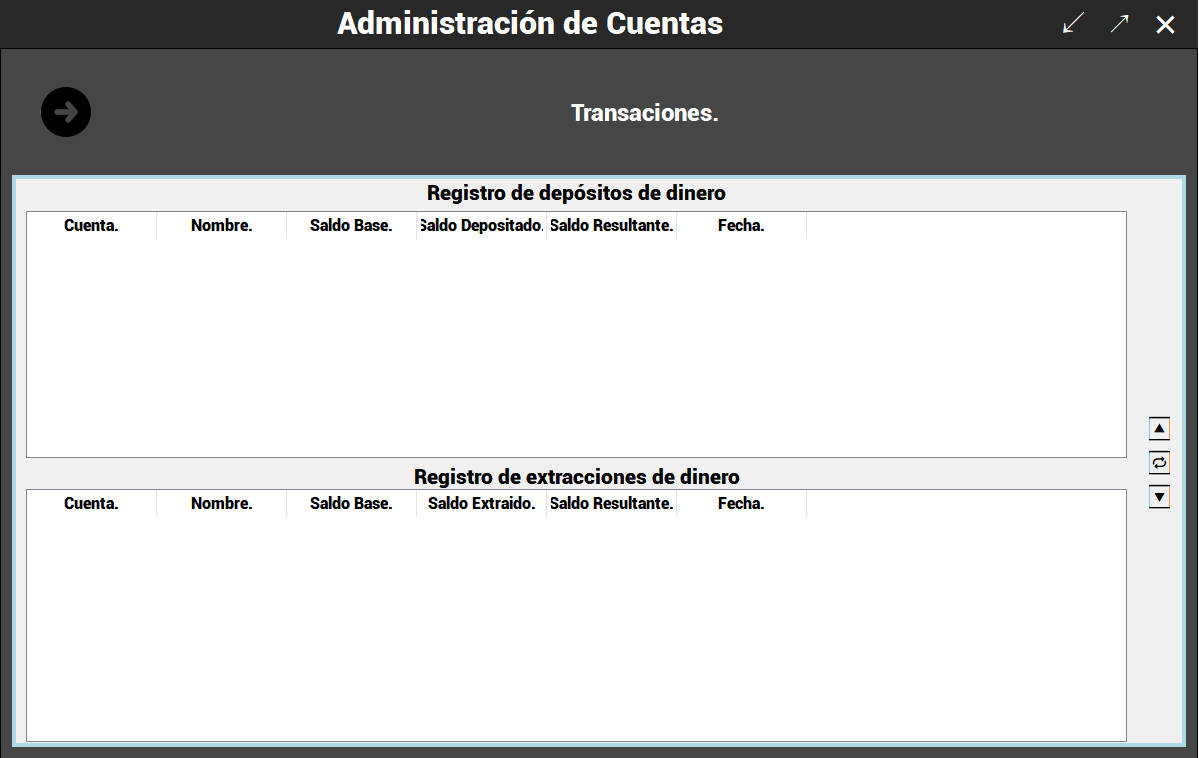


Figura 7, Transacciones

# Conclusiones

En el trabajo se diseñó e implementó un sistema informático para las operaciones básicas de las cuentas bancarias para los clientes de un BPA, garantizando mayor rapidez en las consultas realizadas por los usuarios del sistema.

En la actualidad los bancos poseen sistemas informáticos con sus operaciones automatizadas, constituyendo estas implementadas las más elementales. Considero que se cumplió el objetivo de la práctica laboral investigativa I, aplicando conocimientos de programación y simulando una interacción con los clientes con interfaces visual, adquiriendo mayor experiencia en la implementación de aplicaciones, para lo que se usó el lenguaje C++ con el IDE QTCreator.

# Recomendaciones

Se recomienda para este tipo de sistema, que el trabajo debe encaminarse, considerando:

* El uso de una base de datos relacional, por la posibilidad de su manipulación y la existencia del lenguaje de consulta SQL.
* Aumentar las operaciones que se realizan en este tipo de sistema cercano a la vida real.
* Implementar una ruta cuando se instale el programa.

# 5. Bibliografía

1. ¿Cómo programar C++? 9na Edición - 2014 de Paul Deitel y Harvey Deitel

2. C++ How to Program? 10th Edition - 2017 de Paul Deitel y Harvey Deitel

3. The C++ Programming Language 4th Edition de Bjarne Stroustrup

4. El Lenguaje de Programación 2da Edición de KernighanAmp Ritchie

5. The C++ Standard Library C++ 11 2nd Edition - 2012 de Nicolai M. Josuttis

6. Manual Básico de Programación en C++ de U.C.M

7. Manual Teórico - Práctico de C++ de Alan D. Osorio Rojas

8. Aprenda C++ como si estuviera en primero de Javier García de Jalón

9. Hands-On GUI Programming with C++ and Qt5 de Lee Zhi Eng

10. C++ Soportado con Qt de Efraín Oviedo Regino