

Trabajo Práctico



Materia: Paradigmas de programación 3

Profesora: Alejandra

Alumnos: Gómez, Agustina

Sacca, Axel

Año: 2020

Contenido

[Enunciado 3](#_Toc54386224)

[Desarrollo 3](#_Toc54386225)

[Explicación de las clases 3](#_Toc54386226)

[Detalle de algunas funciones de las clases 5](#_Toc54386227)

# Enunciado

Hacer un sistema que permita la actualización de las cuentas de distintos clientes de la sucursal de un banco realizando altas, bajas, modificaciones y listados. Se manejarán dos archivos binarios: **Cuentas.dat** y **Clientes.dat** con la siguiente información.



# Desarrollo

## Explicación de las clases

Cuentas:

1. Atributos:

* Número de cuenta: nrodeCuenta int
* Tipo (de ahorro: CA o corriente: CC): char tipodeCuenta[2]
* Dni del cliente: int dniC;
* Saldo: doublé Saldo;
* Interés (común para todas las cuentas): static float interés
* Borrado: int borrado (nos indica si el dato tiene que ser borrado o no)

1. M**étodos:**

Constructor con valores predefinidos**:**

* Cuenta(int=0,int=0,string="", double= 0.0);

Destructor:

* ~Cuenta();

Modificación del valor del dni:

* void setDni(int);

Modificación del valor del número de cuenta:

* void setNro(int);

Modificación del valor del tipo de cuenta:

* void setTipo(string);

Modificación del valor del saldo de la cuenta:

* void setSaldo(double);

Modificación de la variable Borrado:

* void setBorrado(int);

Hacemos un depósito en la cuenta:

* void Deposito(double);

Extraemos dinero de la cuenta:

* void Extracion(double);

Buscamos si la cuenta ya existe:

* void buscar(fstream &);

Obtenemos el valor del dni del cliente:

* int getDni();

Obtenemos el número de cuenta:

* int getNro();

Obtenemos el saldo de la cuenta:

* double getSaldo();

Obtenemos el tipo de cuenta:

* string getTipo();

Clientes:

1. Atributos

* Dni: int dni;
* Nombre: char nombre[25]
* Apellido: char apellido[25]
* Dirección: char dirección[25]
* Teléfono: int teléfono;
* Borrado: int borrado(nos indica si el cliente debe ser borrado)

1. Métodos:

Construcctor de cliente con valores predefinidos:

* Cliente(string="", string="", int=0, int=0,string="");

Destructor:

* ~Cliente();

Modifica el valor del dni:

* void setDni(int);

Modifica el valor del nombre respetando el largo de 24 caracteres:

* void setNombre(string);

Modifica el valor del apellido respetando el largo de 24 caracteres:

* void setApellido(string);

Modifica el número de teléfono del cliente:

* void setTelefono(int);

Modifica el valor de la dirección respetando el largo de 29 caracteres:

* void setDireccion(string);

Modifica el valor de la variable borrado:

* void setBorrado(int);

Busca por dni si el cliente ya existe:

* void buscar(fstream &);

Retorna el valor de la variable borrado:

* int getBorrado();

Retorna el valor de la variable nombre:

* string getNombre();

Retorna el valor de la variable apellido:

* string getApellido();

Retorna el valor de la variable direccion:

* string getDireccion();

Retorna el valor de la variable teléfono:

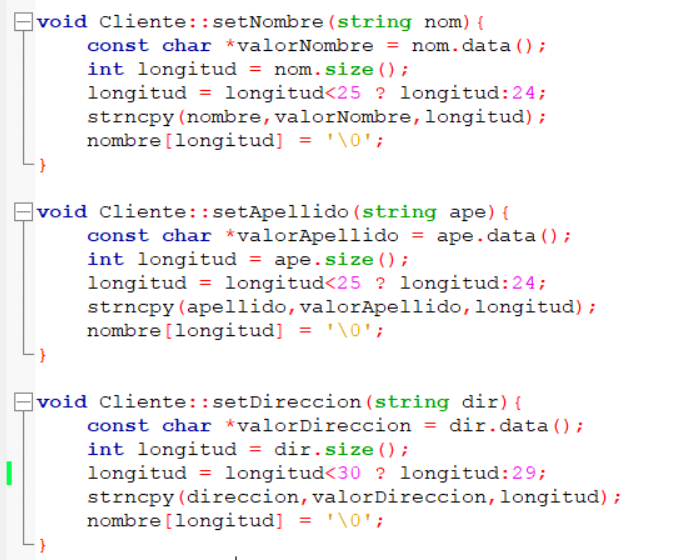
* int getTelefono();

Retorna el valor de la variable Dni:

* int getDni();

## Detalle de algunas funciones de las clases

* Funciones Set:



En la primera parte de las funciones podemos ver que se crea una variable de tipo puntero char cuyo contenido será el dato contenido en la variable string, hacemos esto debido a que la función strncpy solicita que ambas variables sean de tipo puntero char o un vector de chars. Luego podemos ver que se calcula el largo de la variable string debido a que si es más largo que el largo estipulado para cada vector menos 1, ya que en la última posición del vector se guardara ‘\0’ que significa que la cadena termino. Si la cadena de tipo string es más larga se cortara para que no pase el límite.

* Funciones Buscar:

