Proyecto: Sistema Bancario con Gestión de Matrices y Arrays

Introducción:

En el ámbito de la programación estructurada, particularmente en el lenguaje C++, uno de los desafíos más comunes es la organización y manipulación eficiente de datos sin recurrir a tecnologías complejas, como bases de datos externas o estructuras dinámicas avanzadas. Ante esta problemática, las estructuras básicas como los arrays unidimensionales y las matrices bidimensionales representan una alternativa eficaz para gestionar información de forma ordenada, rápida y con bajo consumo de recursos.

El presente proyecto tiene como finalidad diseñar un sistema bancario funcional que permita la gestión de múltiples cuentas, el almacenamiento de saldos y el registro de transacciones entre usuarios, utilizando exclusivamente arrays y matrices en lenguaje C++. A través de este enfoque, se busca demostrar cómo dichas estructuras, combinadas con algoritmos de control y validación mediante bucles, condicionales y operadores lógicos, pueden garantizar la integridad de los datos y la correcta ejecución de las operaciones financieras simuladas.

El sistema propuesto responde a la necesidad de construir soluciones informáticas sencillas pero efectivas, especialmente útiles en contextos educativos o de prototipado. En estos escenarios, la claridad del código, la eficiencia operativa y la comprensión de los fundamentos de la programación estructurada son prioridades esenciales. Por lo tanto, este proyecto no solo contribuye al desarrollo técnico de soluciones orientadas a la gestión de datos financieros, sino que también fortalece las competencias básicas en programación al integrar teoría y práctica en una aplicación concreta.

Pregunta problema:

- ¿Cómo pueden los arrays y las matrices unidimensionales optimizar el almacenamiento de datos en C++?

Planteamiento del problema:

- En el contexto de la programación en C++, uno de los retos más comunes es cómo organizar y manipular estos datos de forma eficaz sin recurrir a estructuras complejas o bases de datos externas. Los arrays y matrices unidimensionales y bidimensionales representan herramientas fundamentales dentro de este lenguaje, que permiten estructurar y procesar información de manera rápida y ordenada. Sin embargo, su uso eficiente requiere una adecuada planificación y comprensión de las operaciones lógicas que se pueden implementar con bucles, condicionales y operadores.

En este proyecto se busca abordar cómo estas estructuras básicas de datos pueden emplearse para construir un sistema bancario funcional, capaz de gestionar múltiples cuentas, almacenar saldos, registrar transacciones entre usuarios y garantizar la integridad de la información sin redundancias ni pérdidas. La finalidad es demostrar que, a pesar de su simplicidad, los arrays y matrices pueden ser claves para optimizar el rendimiento y la claridad de programas orientados a la gestión de datos en contextos financieros.

Objetivo General

- Diseñar un sistema que gestione múltiples cuentas bancarias una entidad financiera mediante el uso de arrays y matrices para simular operaciones financieras.

Objetivos Especifico

- Implementar un sistema de almacenamiento de cuentas usando arrays.
- Diseñar un modelo de transacciones Inter cuentas usando matrices bidimensionales.
- Garantizar la integridad operativa mediante validaciones con operadores y bucles

Marco Teórico

Las estructuras de datos permiten organizar y manipular información en la memoria de manera eficiente. En el lenguaje C++, las estructuras más básicas y fundamentales son los *arrays* y las *matrices*, las cuales ofrecen un acceso rápido a los elementos por medio de índices. Estos se dividen en dos tipos, el primero Arrays unidimensionales, que ayudan a estructurar y almacenar datos del mismo tipo en posiciones contiguas de memoria. Son útiles para representar listas de saldos, identificadores de cuentas o nombres de clientes. El segundo tipo son las Matrices bidimensionales: Se utilizan para representar relaciones entre conjuntos de datos, como, por ejemplo, una tabla de transacciones entre cuentas bancarias (matriz de montos transferidos de cuenta A a B). Estos tipos de almacenamiento de datos nos brindan ciertas ventajas como la rapidez de acceso, estructura clara y mínima carga de procesamiento. Pero también nos limitan al tener un tamaño fijo, menor flexibilidad frente a estructuras dinámicas como listas enlazadas o vectores.

Para operar con arrays y matrices en C++, se requiere el uso adecuado de estructuras de control como bucles (for – while) y condicionales (if – switch), así como operadores lógicos y aritméticos. Que permitan recorrer arrays y matrices, verificar condiciones (por ejemplo, saldo suficiente antes de transferir), y garantizar la lógica operativa del sistema. Siendo esenciales los operadores lógicos para garantizar la integridad de los datos evitar transferencias inválidas, prevenir pérdidas de datos o duplicaciones, y asegurar que las operaciones respeten las reglas del sistema. El desarrollo de un sistema de simulación bancario básico en C++ permite aplicar conceptos fundamentales de programación orientada a la solución de problemas reales como la gestión de cuentas a través de Arrays que pueden usarse para almacenar datos como números de cuenta, nombres de clientes y saldos. Un registro de transacciones para a través de una matriz representar un historial de transferencias entre cuentas, simulando operaciones bancarias básicas sin necesidad de bases de datos.

El uso de arrays y matrices en C++ representa una solución adecuada para construir sistemas bancarios básicos con bajo consumo de recursos y buena organización de datos. Estos elementos, combinados con algoritmos de control y validación, permiten simular operaciones financieras reales y desarrollar competencias fundamentales en programación estructurada. A través de este proyecto, se demuestra que estructuras simples, cuando se aplican con lógica adecuada, pueden resolver problemas relevantes.

Diseño metodológico

Sistema

- Un array unidimensional para almacenar las cuentas bancarias y sus respectivos saldos.
- Una matriz bidimensional para registrar las transacciones entre cuentas (cuenta origen y cuenta destino).
- Reglas operativas para asegurar la integridad de la información (por ejemplo, evitar saldos negativos o duplicaciones).
  El diseño incluye el uso de estructuras de control como bucles (for, while) y condicionales (if, switch) para manejar la lógica del sistema.

Implementación

Las funcionalidades implementadas incluyen:

- Creación de cuentas.
- Gestión de saldos.
- Transferencias intercuentas.
- Validaciones lógicas antes de cada operación.
  El sistema utiliza la consola como interfaz principal, lo que permite una

Validación del sistema

Se llevan a cabo pruebas funcionales con diferentes escenarios para evaluar el comportamiento del sistema. Estas pruebas consideran:

- Transacciones válidas y exitosas.
- Manejo de errores por saldo insuficiente.
- Control de cuentas inexistentes o duplicadas.
- Visualización de resultados en pantalla.
  Los resultados se contrastan con los objetivos específicos del proyecto para verificar la funcionalidad del sistema y su coherencia lógica.