Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 1er. Semestre 2025

PRÁCTICA NO. 1

201907179 – Wilson Manuel Santos Ajcot

Resumen

El ensayo describe un sistema bancario desarrollado en Python mediante la implementación de conceptos de Programación Orientada a Objetos (POO). El sistema gestiona cuentas bancarias de dos tipos: Cuenta de Ahorro y Cuenta Monetaria, utilizando los pilares fundamentales de la POO como herencia, polimorfismo, encapsulamiento y abstracción. La Cuenta de Ahorro permite el cálculo y aplicación de intereses, mientras que la Cuenta Monetaria permite operar con un límite de crédito. Este sistema responde a la necesidad de gestionar diferentes tipos de cuentas de manera eficiente y sencilla, utilizando una interfaz de consola interactiva. La propuesta tiene un impacto técnico en la automatización de operaciones bancarias, socialmente facilita la gestión de finanzas personales y económicamente optimiza los procesos de gestión de cuentas en una entidad bancaria. Este ensayo destaca cómo la correcta aplicación de la POO contribuye a la creación de soluciones más estructuradas y eficientes en el ámbito del desarrollo de software.

Palabras clave

Programación Orientada a Objetos (POO)
Paradigma de programación que organiza el software en "objetos", los cuales tienen propiedades y comportamientos, facilitando la modularidad y reutilización del código.
Sistema Bancario: Sistema que gestiona cuentas, transacciones y servicios financieros como depósitos, retiros y cálculo de intereses.
Cuenta de Ahorro: Tipo de cuenta bancaria donde se depositan fondos con el objetivo de ahorrar, y que generalmente genera intereses con el tiempo.
Cuenta Monetaria: Cuenta bancaria similar a la de ahorro, pero sin la posibilidad de generar intereses, y utilizada principalmente para transacciones flexibles.
Python: Lenguaje de programación de alto nivel, interpretado y fácil de aprender, utilizado en desarrollo web, análisis de datos, automatización y más.

Abstract

The essay describes a banking system developed in Python using the implementation of Object-Oriented Programming (OOP) concepts. The system manages two types of bank accounts: Savings Account and Monetary Account, utilizing key OOP principles such as inheritance, polymorphism, encapsulation, and abstraction. The Savings Account allows for the calculation and application of interest, while the Monetary Account allows operations with a credit limit. This system meets the need for efficient and simple management of different account types through an interactive console interface. The proposal has a technical impact on automating banking operations, socially facilitating personal finance management, and economically optimizing account management processes in a banking entity. This essay highlights how the proper application of OOP contributes to the creation of more structured and efficient software solutions.

Keywords

Object-Oriented Programming (OOP)

Programming paradigm that organizes software into "objects," which have properties and behaviors, enhancing code modularity and reuse.

Banking System: *System that manages accounts, transactions, and financial services like deposits, withdrawals, and interest calculations.*

Savings Account: *Type of bank account where funds are deposited with the purpose of saving, usually generating interest over time.*

Monetary Account: *Bank account similar to a savings account but without earning interest, primarily used for flexible transactions.*

Python: *High-level, interpreted programming language that is easy to learn, used in web development, data analysis, automation, and more.*

Introducción

El objetivo de esta práctica es desarrollar un sistema bancario utilizando la Programación Orientada a Objetos (POO) con el lenguaje de programación Python. El sistema tiene como objetivo la gestión de cuentas bancarias, específicamente las de tipo Cuenta de Ahorro y Cuenta Monetaria. Las funcionalidades principales del sistema incluyen la capacidad de realizar depósitos, retiros, y calcular intereses.

Utilizando los principios fundamentales de la POO, como la herencia y el polimorfismo, se busca crear un diseño modular y flexible que facilite la adición de nuevos tipos de cuentas y funcionalidades en el futuro. Esta práctica permitirá entender cómo las técnicas de POO pueden ser aprovechadas para desarrollar sistemas complejos y escalables.

Desarrollo del tema

a. Introducción a la Programación Orientada a Objetos

La Programación Orientada a Objetos (POO) es un paradigma de programación que permite modelar sistemas de manera más modular, reutilizable y fácil de mantener. En el contexto de este sistema bancario, la POO nos permite estructurar las cuentas y sus operaciones de forma que podamos extender el sistema en el futuro sin dificultad. Además, proporciona abstracción, encapsulamiento, herencia y polimorfismo.

En este proyecto, creamos una clase base llamada Cuenta, que encapsula los atributos y métodos comunes a todas las cuentas. Las clases CuentaAhorro y CuentaMonetaria son derivadas de la clase Cuenta y heredan sus propiedades, pero también tienen sus propios comportamientos específicos, como la forma en que se calculan los intereses en cada tipo de cuenta.

Para ilustrar mejor cómo se organiza el sistema y cómo se manejan estas relaciones entre clases, se presenta un diagrama de clases UML, el cual muestra la estructura de las clases y cómo se heredan las funcionalidades.

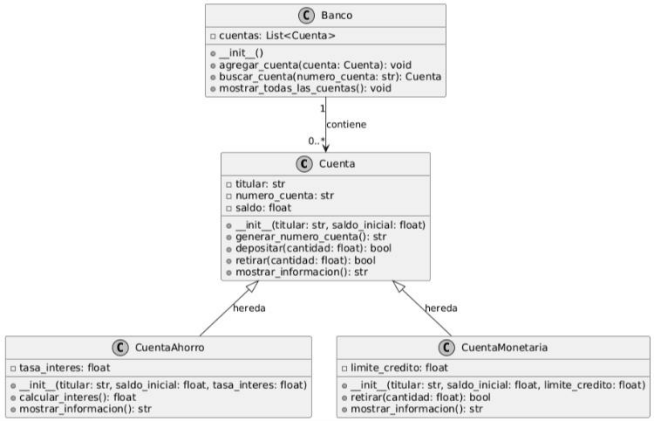

Figura 1: Diagrama de clases del sistema bancario.

Este diagrama muestra cómo la clase Cuenta se extiende por las clases CuentaAhorro y CuentaMonetaria, y cómo todas las cuentas pueden realizar operaciones comunes como depósitos y retiros, además de incluir sus propios métodos para calcular los intereses.

b. Operaciones Básicas del Sistema Bancario

El sistema que hemos diseñado permite realizar tres operaciones principales: depósitos, retiros y cálculo de intereses.

- Depósitos: Los usuarios pueden agregar dinero a sus cuentas. Esta operación aumenta el saldo de la cuenta.
- Retiros: Los usuarios pueden retirar dinero de sus cuentas, siempre que el saldo sea suficiente para cubrir la transacción.
- Cálculo de Intereses: Dependiendo del tipo de cuenta, los intereses pueden ser calculados de forma diferente. Las cuentas de ahorro tienen un tipo de interés fijo, mientras que las cuentas monetarias

tienen un interés basado en el saldo y el tiempo que ha permanecido en la cuenta.

Estas operaciones se implementan como métodos dentro de las clases correspondientes. Para representar cómo interactúan estas operaciones con los objetos de cuenta, se puede agregar un diagrama de flujo.

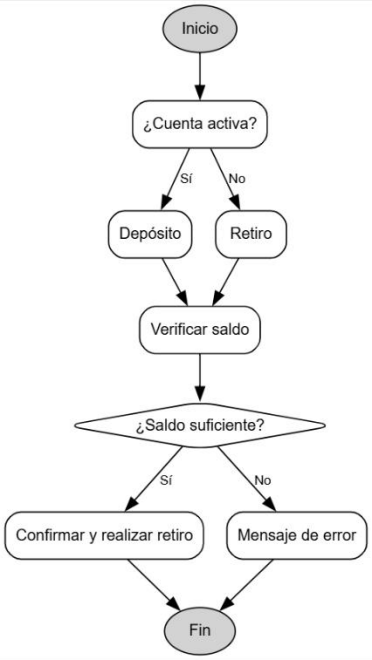

Figura 2: Diagrama de flujo de las operaciones bancarias

Fuente: elaboración propia.

Este diagrama puede mostrar cómo un usuario realiza un depósito o un retiro, y cómo el sistema ajusta el saldo y verifica si las condiciones necesarias son cumplidas.

c. Escalabilidad y Flexibilidad del Sistema

Una de las ventajas de usar la POO es la facilidad con la que podemos extender el sistema. Si en el futuro se necesitan agregar más tipos de cuentas, como una Cuenta Corriente o una Cuenta de Inversión, solo basta con crear nuevas clases que hereden de la clase Cuenta y agregar sus comportamientos específicos.

Además, la estructura modular permite añadir nuevas funcionalidades, como transferencias entre cuentas o la integración de un sistema de gestión de usuarios, sin afectar la base del sistema.

Para ilustrar esto, podemos agregar una tabla comparativa de las posibles extensiones futuras y sus implicaciones, en términos de lo que se necesitaría modificar o agregar.

Tabla I:

Comparativa de las futuras extensiones del sistema.

Extensión	Modificación Necesaria	Beneficios
Cuenta Corriente	Crear nueva clase que herede de Cuenta	Implementar cheques y cargos por sobregiro.
Cuenta de Inversión	Crear nueva clase que herede de Cuenta	Calcular intereses más complejos y transferencias a fondos.
Gestión de Usuarios	Agregar clases para gestionar clientes y sus cuentas	Permitir interacción con diferentes clientes y sus saldos.

Fuente: elaboración propia.

Conclusiones

El uso de la Programación Orientada a Objetos ha permitido diseñar un sistema bancario modular, flexible y fácil de extender. Gracias a la herencia y el polimorfismo, el sistema es capaz de gestionar diferentes tipos de cuentas y operaciones bancarias con una estructura clara y bien organizada.

Este enfoque no solo facilita el mantenimiento del sistema, sino que también permite agregar nuevas funcionalidades en el futuro sin tener que rehacer el código base. Como conclusión, la POO ofrece

ventajas importantes en el desarrollo de sistemas complejos, como la escalabilidad, la reutilización de código y la fácil integración de nuevas características.

Anexos

Figura 3. Diagrama de Componentes del Sistema Bancario

Fuente: elaboración propia.

Referencias bibliográficas

Beazley, D. M. (2009). *Python Essential Reference*. 4th Edition. Addison-Wesley.

Docs.python.org. (2022). *Python 3.10.4 Documentation*. [online] Available at: https://docs.python.org/3/ [Accessed 19 February 2025].

Hunt, A., & Thomas, D. (2009). *Pragmatic Programmer: From Journeyman to Master*. Addison-Wesley Professional.

Python Software Foundation." (2023). Python Documentation. [online] Available at: https://www.python.org/doc/ [Accessed 15 February 2025].

W3Schools." (2023). *Python OOP Tutorial*. [online] Available at: https://www.w3schools.com/python/python_classes.asp [Accessed 17 February 2025].