A Centralized Bank-System written in C

Sérgio Costa

August 2023

Contents

1 Introduction

This document is a guideline to solve a problem which was created to revisit my skills in the **C** programming language in the first place.

The skills and nuances that I want to focus on are discussed in the Aim section.

The problem itself is a simple Bank-System where a person can have multiple bank accounts.

An account keeps track of the current balance which is the result of withdraw and currency deposits.

2 Aim

My aim with this project is to develop and reinforce skills of structuring data, and manage it, programming, and design software itself.

For the purpose, the all project shall be brake down at tiny and simple counterparts.

Structures

In this category I want to aim the simple manage of a `structure` itself, with atomic variables only.

This `structure` shall have some properties which will serve orders criteria. Moreover it shall be fitted in a element abstract data structure.

The `ID` is created.

Structures of Structures

In here I want to do everything that a `structure` does, with the extent of an indirection to another one. Here i want to work in specific with the inheritance of order criteria as mentioned up above.

The `Person` is born... As long as the `BankAccount`.

Date & Time

In order to have a library that handles quick & easy the shores of date and time.

Enum

To provide and manage simple (and at scale) translations of the `Date-Time` library despite of the interest to handle `enum`'s in **C** and explore its potentials.

3 Strategy

My strategy aims the elaboration of some data structure modules that can handle any type of data and work with it.

Due to the scale of this project I will first deal with the elements as pointers to an existing data. Further I will try to accommodate the data itself via byte management.

At design level I will try to control the data of my modules via encapsulation. I will declare new types along side pointers to `structure`'s.

Redundancy of functions may occur in the API extent to ensure simplicity of usage and disambiguation.

I will deal with dynamic memory as much as possible. That way I know that the program is scalable, and robust in a new entities creation level.

The `Bank` will store `Peron`'s and `BankAccount`'s in `HashTable`'s and the link/relationship between an owner and it's respective accounts will be provided by an `HashTable` from the `ID` of the former to an `Array` of `ID`'s of the second.

The interface between human and machine will be via command line such that the view will be controlled by a component through the extents of the model (`Bank`).

4 Highlights

- Reusable components.
- Import & Export data:
  - Text;
  - Binary.

5 Disclosure

A software it is a mere logic relationship between simple and atomic actions (procedures & functions).

A component (a module) may be permissive to the extent that can compromise the system itself, nonetheless, the safety of its use shall be granted by a combination of it's atomic actions. That way ensuring the correctness of the system.

Through this approach results an abroad use of components, a more clean code, and a much more rapid integration of components that have already been made.

6 Who am I?

My name is Sérgio (Miguel Passinhas) Costa, I am from Portugal and I am currently a student of *Engenharia Informática*, informatics engineering, at Minho university.

I'm looking to integrate theoretical knowledge of the field alongside the practical one. In this way I am building a repertoire of my insights in this area, mine of expertise, as long as building some portfolio content to get you to know me beside you are a colleague or a fellow recruiter... boy or girl.