COMP603/ENSE600

Program Design & Construction / Software Construction – Assessment

2023 Semester 1 – PDC Project 1 Report – Group 124

Contributors: Hayden Richard-Marsters SID: 21152003

Aim: Create a comprehensive system that provides a subset of tools for managing a user's assets. It allows control over the user's assets variables, such as market values and total amount in holdings. The program also enables data persistence by incorporating a saving system to a text file, which can be read and written to in the program. The program aims not to be limited by asset values, as it incorporates user analysis by including portfolio metrics such as sorting algorithms to return a list of assets in order of highest to lowest returns, as well as a weighting method to show the portfolio's asset composition. The program is designed to handle a range of base common asset foundations, including stock, real estate, and ETF types. The user can interact and maintain control over how the portfolio is treated through a command-line interface. Overall, the program's aim is to create a centralized asset management system that simplifies the process of managing various assets and provides the user with clear analytics.

Setup: The program consists of ten classes. Firstly, the Asset class, which is the foundation of the program. Through inheritance, Asset includes sub-set classes Stock, Real Estate, and ETF, which all inherit Asset classes' instance variables, constructors, and methods to round the object classes. A Portfolio object class houses the assets and asset information, containing several public methods for managing and analysing the portfolio. The portfolio is instantiated into various data structures, including lists, maps, and sets, for varying functionalities of the program.

Functionality: Functional classes are incorporated into the program as modular classes to incorporate reusability and maintainability, such as portfolio metric calculations. These methods return a string of asset weight composition. The program also incorporates a file updater class that includes methods that read and write to files, allowing the user to pre-load an existing file and use the asset management system or create a new and load their portfolio into a text file themselves, or choose to read and load an imported text file into the system. A CUI class defines a command-line user interface for an asset management system, creating a looping menu until the user inputs "99," allowing the user to follow prompts to their desired function. Lastly, the program incorporates a transaction class that returns a list of transaction objects. This is done by taking a portfolio as a parameter and iterating through the portfolio's asset list, assigning the subsequent values to a transaction object and adding it to a transaction list, which is then returned.