21MDS37-DATA STRUCTURES AND ALGORITHMS-II

Project Title: Stock Analysis

Team Members:

71762232026 (Kavya V.V),

71762232059 (Vedhapriyaa . S),

71762232039 (Prithika . J)

Abstract:

The Stock Analysis System is a comprehensive application built using the tkinter library, designed to offer users an intuitive and user-friendly investment advisory interface. This system leverages a binary search tree to systematically organize and manage inputted data pertaining to various companies, encompassing critical financial details such as the company's name, market capitalization, revenue, PE ratio, and dividend yield. The core functionality of the program lies in its utilization of a proprietary scoring algorithm during the insertion process, strategically arranging companies within the binary search tree based on a nuanced evaluation of multiple financial metrics.The graphical user interface (GUI), facilitated through the Investment App class, provides users with a seamless platform to input intricate details about companies. Users can dynamically add these companies to the binary search tree, where the scoring algorithm works to rank them based on a holistic assessment of their financial health. A key feature of the application is its ability to furnish real-time updates as users input new companies, ensuring that the binary search tree consistently reflects the most up-to-date information.The heart of the application is the sophisticated scoring criterion, which takes into account crucial factors such as market capitalization, revenue, PE ratio, and dividend yield. This multidimensional approach enables users to identify and prioritize companies with the highest investment potential, offering a nuanced and insightful perspective on potential investment opportunities.In essence, the Stock Analysis System serves as an invaluable investment advisor, empowering users to assess and compare companies systematically. By providing a tool that evaluates companies based on specified financial indicators, this application caters to both novice and experienced investors, offering a holistic solution for making informed investment decisions in the dynamic stock market landscape.

Keywords: Python Tkinter , Investment Advisory ,Company Information, Application, Financial Metrics , Scoring Algorithm , User Input , Real-time Updates

Data Structure used: Binary search tree

STOCK ANALYSIS

Problem Statement:

Design a Stock Analysis System for user-friendly investment advice. Utilize a binary search tree to organize company data, employing a scoring algorithm based on market cap, revenue, PE ratio, and dividend yield. Enable users to input multiple companies dynamically, updating the BST in real-time. The system aims to assist users in assessing and comparing companies for potential investments based on specified financial indicators.

Aim:

To create a Stock Analysis System with a tkinter GUI to facilitate user-friendly investment advisory, utilizing a binary search tree and scoring algorithm for dynamic company evaluation. The aim is to assist users in making informed investment decisions based on real-time analysis of specified financial indicators.

Installation:

Dependencies:

- Python 3.x
- Tkinter

Usage:

Launch Application:

Run the Python script in a terminal or command prompt.

A graphical user interface (GUI) window titled "Company Investment Advisor" will appear.

Add Company:

Enter company details in the provided field.

Click the "Add Company" button.

Real-time updates will be displayed in the text area, confirming the successful addition of the company.

Add Multiple Companies:

Repeat the above steps to input and add additional companies.

The binary search tree dynamically updates to reflect the new companies.

Find Best Company to Invest:

Click the "Find Best Company to Invest" button.

The system evaluates the companies based on the scoring algorithm and displays the best investment option in the text area.

Technology Stack:

Programming Language:Python 3.x

GUI Toolkit: Tkinter

Data structure : Binary search tree

Additional Libraries (Standard Libraries):

- ttk - Part of tkinter for enhanced GUI widgets.
- tk - Core module for creating the GUI.

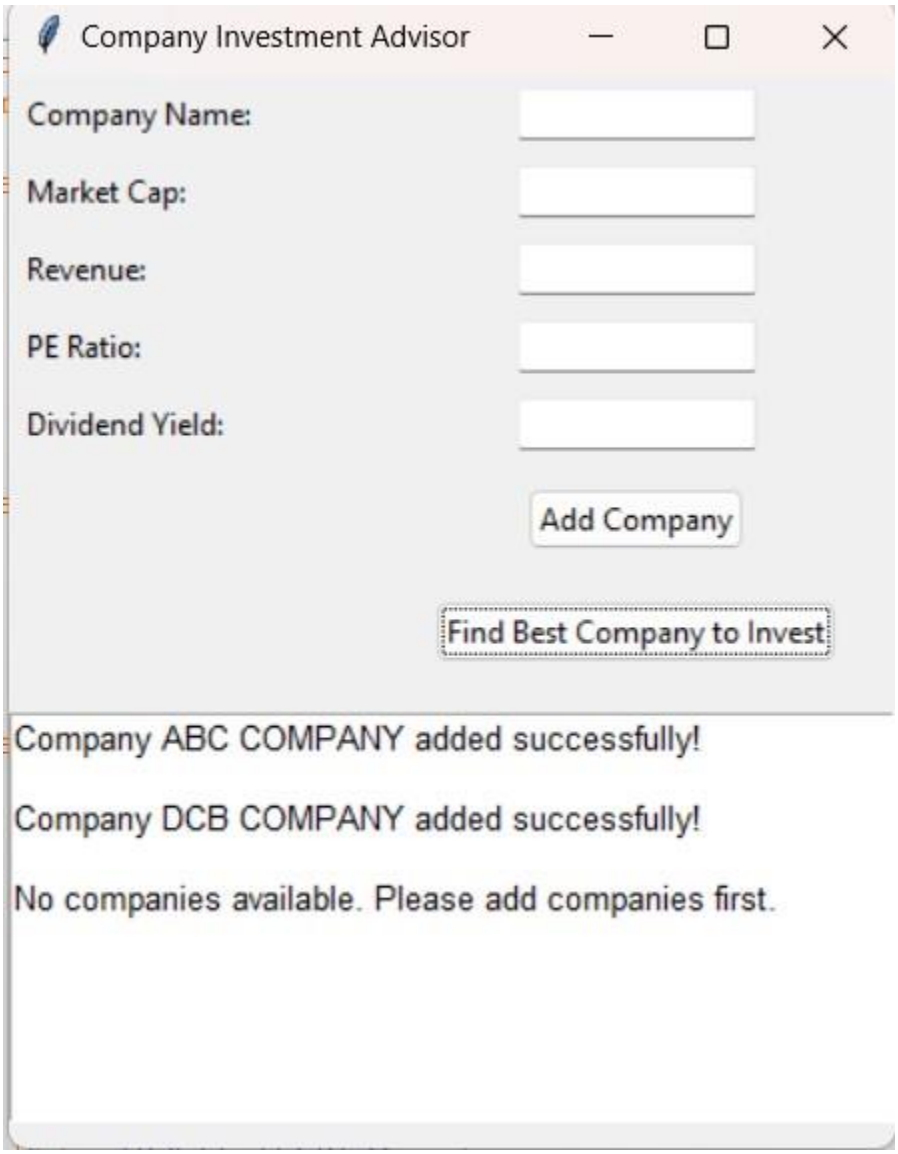
PSEUDOCODE**:**

```
Class CompanyInfo:
    def __init__(self, name, market_cap, revenue, pe_ratio, dividend_yield):
        self.name = name
        self.market_cap = market_cap
        self.revenue = revenue
        self.pe_ratio = pe_ratio
        self.dividend_yield = dividend_yield
Class TreeNode:
    def __init__(self, company):
        self.company = company
        self.left = None
        self.right = None
```

```
Class BinarySearchTree:
    def __init__(self):
        self.root = None
    def insert(self, company):
        if not self.root:
            self.root = TreeNode(company)
        else:
            self._insert_recursive(self.root, company)
    def _insert_recursive(self, node, company):
        if (node.company.market_cap + node.company.revenue -
                node.company.pe_ratio + node.company.dividend_yield) < \
                (company.market_cap + company.revenue -
                 company.pe_ratio + company.dividend_yield):
            if node.right:
                self._insert_recursive(node.right, company)
            else:
                node.right = TreeNode(company)
        else:
            if node.left:
                self._insert_recursive(node.left, company)
            else:
                node.left = TreeNode(company)
```

```
Class InvestmentApp:
    def __init__(self, root):
        self.root = root
        self.root.title("Company Investment Advisor")
        self.company_binary_tree = BinarySearchTree()
        ttk.Label(root, text="Company Name:").grid(row=0, column=0, padx=5, pady=5,
sticky=tk.W)
        self.name_entry = ttk.Entry(root, width=15)
        self.name_entry.grid(row=0, column=1, padx=5, pady=5
    def add_company(self):
        name = self.name_entry.get()
        market_cap = float(self.market_cap_entry.get())
        revenue = float(self.revenue_entry.get())
        pe_ratio = float(self.pe_ratio_entry.get())
        dividend_yield = float(self.dividend_yield_entry.get())
        company = CompanyInfo(name, market_cap, revenue, pe_ratio, dividend_yield)
        self.company_binary_tree.insert(company)
        self.clear_entries()
        self.result_text.insert(tk.END, f"Company {name} added successfully!\n\n")
if __name__ == "__main__":
    root = tk.Tk()
    app = InvestmentApp(root)
    root.mainloop()
```

Output:


Company Investment Advisor
Company Name:
Market Cap:
Revenue:
PE Ratio:
Dividend Yield:
Add Company
Find Best Company to Invest
Company ABC COMPANY added successfully!
Company DCB COMPANY added successfully!
No companies available. Please add companies first.