Assignment 5

DDL: 23:59, Dec 22, 2023

Note: Late homework assignments will not be accepted, unless you have a valid written excuse (medical, etc.). You must do this assignment alone. No team work or "talking with your friends" will be accepted. No copying from the Internet. Cheating means zero. Here are a few extra instructions: Give meaningful names to your variables so we can easily know what each variable is used for in your program. Put comments in your code (in English!) to explain WHAT your code is doing and also to explain HOW your program is doing it. Make sure all your code is properly indented (formatted). Your code should be beautiful to read.

Part 1: Programming

Write programs in C++ to complete the following tasks:

Task 1

Write a C++ program to implement a class called Circle that has private member variables for radius. Include member functions to calculate the circle's area and circumference.

```
int main() {
  // Create a circle object
  double radius;
  std::cout << "Input the radius of the circle: ";
  std::cin >> radius;
  Circle circle(radius);

  // Calculate and display the area
  double area = circle.calculateArea();
  std::cout << "Area: " << area << std::endl;

  // Calculate and display the circumference
  double circumference = circle.calculateCircumference();
  std::cout << "Circumference: " << circumference << std::endl;

  return 0;
}
```

The sample output:

```
Input the radius of the circle: 5.2
Area: 84.9486
Circumference: 32.6725
Input the radius of the circle: 4
Area: 50.2654
Circumference: 25.1327
```

Task 2

Write a C++ program to implement a class called BankAccount that has private member variables for account number and balance. Include member functions to deposit and withdraw money from the account.

```
int main() {
  // Create a bank account object
  string sacno = "SB-123";
  double Opening_balance, deposit_amt, withdrawal_amt;
  Opening_balance = 1000;
  cout << "A/c. No." << sacno << " Balance: " << Opening_balance << endl;

  BankAccount account(sacno, 1000.0);

  // Deposit money into the account
  deposit_amt = 1500;
  cout << "Deposit Amount: " << deposit_amt << endl;
  account.deposit(deposit_amt);

  // Withdraw money from the account
  withdrawal_amt = 750;
  cout << "Withdrawl Amount: " << withdrawal_amt << endl;
  account.withdraw(withdrawal_amt);

  // Attempt to withdraw more money than the balance
  withdrawal_amt = 1800;
  cout << "Attempt to withdrawl Amount: " << withdrawal_amt << endl;
  account.withdraw(withdrawal_amt);

  return 0;
}
```

The sample output:

```
A/c. No.SB-123 Balance: 1000
Deposit Amount: 1500
Deposit successful. Current balance: 2500
Withdrawl Amount: 750
Withdrawal successful. Current balance: 1750
Attempt to withdrawl Amount: 1800
Insufficient balance. Cannot withdraw.
```

Task 3

The goal of this task is to simulate an investment fund scenario using C++ classes and objects. Students will implement a program that models the behavior of an investment fund, allowing users to buy and sell stocks, record investments from individual investors, and display relevant information about the fund's portfolio and investor contributions.

1. Global Constants and Stock Prices

- Define global constants for the number of stocks (`NUM_STOCKS`) and trading days (`TRADING_DAYS`).
- Create a 2D array `stockPrices` representing the close prices of 5 stocks over 22 trading days.

2. Human, Investor, and Fund Classes

- Implement a `Human` class with basic attributes such as name and age.
- Create an `Investor` class that inherits from `Human`. Investors can invest money into a fund.
- Develop a `Fund` class with methods for buying and selling stocks, recording investments, and displaying fund information.

3. Fund Operations

- Allow the `Fund` class to buy and sell stocks on specific days, adjusting stock weights and cash positions accordingly.
- Implement methods to record investments from investors, display fundraising information, and showcase the fund's portfolio.

4. Investor Operations

- Enable the `Investor` class to invest a specified amount of money into a fund, updating the investor's capital.
- Display information about each investor, including their investments and remaining capital.

5. Simulation and Display

- Create a simulation in the main function where investors invest in the fund, stocks are bought and sold, and the fund's portfolio is displayed.
- Display relevant information at each step, such as investment details, fund portfolio, and the total value of the fund on the last trading day.

Additional Notes

- Use appropriate member variables and encapsulation in the classes.
- Ensure error handling for insufficient funds or stocks during transactions.
- The provided code contains placeholders for certain functionalities; students should complete the logic based on the task requirements.

```
int main() {
    //  Create investors
    Investor investor_john("John Doe", 30, 15000.0);
    Investor investor_alex("Alex Wong", 27, 7000.0);

    // Create funds
    Fund fund_g("Global Growth Fund");
    Fund fund_r("River Flow Hedge Fund");
    cout << "Display initial investor information" << endl;
    investor_john.displayInvestorInfo();
    investor_alex.displayInvestorInfo();
    cout << endl;

    cout << "Simulate investments" << endl;
    investor_john.invest(12000.0, fund_g);
    investor_alex.invest(5000.0, fund_r);
    investor_john.invest(3000.0, fund_r);
    investor_alex.invest(5000.0, fund_g);
    cout << endl;

    cout << "Display updated investor information" << endl;
    investor_john.displayInvestorInfo();
    investor_alex.displayInvestorInfo();
    cout << endl;

    cout << "Simulate buying and selling stocks in the fund" << endl;
    fund_g.buyStocks(2, 10, 4000.0);  // Buy $4000 worth of Stock 3 on Day
10
    fund_g.sellStocks(2, 15, 3000.0);  // Sell $300 worth of Stock 5 on
Day 15

    fund_r.buyStocks(2, 12, 2000.0);  // Buy $2000 worth of Stock 3 on Day
12
    fund_g.sellStocks(2, 19, 1000.0);  // Sell $1800 worth of Stock 5 on
Day 19
    cout << endl;

    cout << "Display fund investments" << endl;
    fund_g.displayFundRaiseInfo();
    fund_r.displayFundRaiseInfo();
    cout << endl;

    cout << "Dislplay fund final value" << endl;
    double lastValue_fund_g = fund_g.getFundValueLastDay();
    double lastValue_fund_r = fund_r.getFundValueLastDay();
    cout << "The Global Growth Fund 's value at the last day is: " <<
lastValue_fund_g << endl;
    cout << "The River Flow Hedge Fund 's value at the last day is: " <<
lastValue_fund_r << endl;

  return 0;
}
```

The sample output is below:

```
Display initial investor information
Investor Name: John Doe Age: 30 Current cash remained: 15000
Investor Name: Alex Wong Age: 27 Current cash remained: 7000
```

```
Simulate investments
John Doe invested $12000 in Global Growth Fund
Alex Wong invested $5000 in River Flow Hedge Fund
John Doe invested $3000 in River Flow Hedge Fund
Capital is insufficient for the investment

Display updated investor information
Investor Name: John Doe Age: 30 invests Global Growth Fund for 12000 USD,
invests River Flow Hedge Fund for 3000 USD,  Current cash remained: 0
Investor Name: Alex Wong Age: 27 invests River Flow Hedge Fund for 5000
USD,  Current cash remained: 2000

Simulate buying and selling stocks in the fund
Global Growth Fund Bought 52 shares of Stock 3 on Day 10 for $4000
Global Growth Fund Sold 38 shares of Stock 3 on Day 15 for $3000
River Flow Hedge Fund Bought 25 shares of Stock 3 on Day 12 for $2000
Global Growth Fund Sold 12 shares of Stock 3 on Day 19 for $1000

Display fund investments
In Fund Global Growth Fund:
Investor John Doe invests 12000USD
In Fund River Flow Hedge Fund:
Investor Alex Wong invests 5000USD
Investor John Doe invests 3000USD

Dislplay fund final value
The Global Growth Fund 's value at the last day is: 12160.4
The River Flow Hedge Fund 's value at the last day is: 8005
```

Submission

The main_task*.cpp file is used to call the functions to check if success or not. Implement the `Assignment5.h` and `Assignment5.cpp` file. Compress these files into a zip file and rename it Assignment5_q0123456789.zip and upload to iSpace before deadline. (replace q0123456789 with your student ID number) Files list in the zip:

```
Assignment5.h
Assignment5.cpp
main_task1.cpp
main_task2.cpp
main_task3.cpp
```