Test Cases

int defineSector(String sectorName)

Input validation
- Null or empty sectorName
- sectorName already exists in our database.

Boundary cases
- No such boundary case.

Control flow
- Call with the sectorName containing a simple string.
- Call with special characters and Numbers in the form of string only.

Data flow
- Call before any other method in this program.

int defineStock(String companyName, String stockSymbol, String sector)

Input validation
- Null or empty companyName.
- Null or empty stockSymbol
- Null or empty sector.
- stockSymbol already exists in our database.
- sector does not exist in our database.

Boundary cases
- No such boundary case.

Control flow
- Call first time for this companyName.
- Call this for the same company but different stockSymbol and sector.
- Call with the same companyName and sector but different stocks.
- Call twice for different companyName and stockSymbol but same sector.

Data flow
- Call after defineSector.

- Call before or after defineProfile.

boolean setStockPrice(String stockSymbol, double perSharePrice)

Input validation
- Null or empty stockSymbol
- Set persharePrice as 0.
- Set perSharePrice as negative.
- stockSymbol does not exist in our database.

Boundary cases
- No such boundary case.

Control flow
- Call with the newly created stock.
- Call to set the price lower than the current price.
- Call to set the price higher than the current price.

Data flow
- Call after defineStock.
- Call before or after createAccount.

int defineProfile(String profileName, Map<String, Integer> sectorHoldings)

Input validation
- Null or empty profileName.
- Null or empty sectorHoldings.
- sectorHoldings does not contain the cash sector.
- sectorHoldings contain sectors that do not exist in the database.
- profileName already exists in our database.

Boundary cases
- Call with 100% Cash sector.
- Call with more than one sector other than cash.

Control flow
- Call with the 0% cash sector.
- Call with 50% cash and 50% any other sector.

- Call with sectorHoldings which contain cash and many other sectors.

Data flow
- Call after defineSector.
- Call before or after defineStock.

int addAdvisor (String advisorName)
int addClient (String clientName)

Input validation
- Null or empty advisorName or clientName.
- advisorName or clientName already exists in our database.

Boundary cases
- No such boundary case.

Control flow
- Call with the advisorName or clientName containing a simple string.
- Call with special characters and Numbers in the form of string only.

Data flow
- Call before createAccount.
- Call before or after defineSector, defineProfile, defineStock, or SetStockPrice.

int createAccount(int clientId, int financialAdvisor, String accountName, String profileType, boolean reinvest)

Input validation
- clientId, financialAdvisor, accountName, and profileType any one of the data did not exist in our database.
- clientId or financialAdvisor is set as a negative integer.
- clientId set as advisorId or financialAdvisor set as investorId
- Null or empty accountName or profileType.
- accountName for the same client already exists in our database.

Boundary cases
- Create an account for a client who already has many accounts.

Control flow

- Create an account for a new client who did not have any previous account.
- Create an account for a client who already had an account but this time with a different financialAdvisor.
- Create an account for a client who already had an account but this time with a different profileType.
- Create an account with the account name that had already been used by another client.
- Create an account with the same financialAdvisor that they had in their previous account.
- Create an account with the same profileType that they had in their previous account.
- Create an account by setting reinvest as true and then also check with false.

Data flow

- Call after defineSector, defineProfile.
- Call after addClient and addAdvisor.
- Call before or after defineStock, or SetStockPrice.

boolean tradeShares(int account, String stockSymbol, int sharesExchanged)

Input validation

- accountId not exist in our database.
- accountId is set as a negative integer.
- accountId set as advisorId.
- Null or Empty stock symbol.
- Set sharesExchanged as zero.
- stockSymbol does not exist in our database.

Boundary cases

- Sell all the stocks held by the account.
- Buy stocks with the entire amount present in cash.

Control flow

- Perform trade with the newly created account.
- Perform “Buy Trade” when there is no cash in the account
- Perform “Sell Trade” when there are no stocks in the account.
- Perform “Cash In Trade” when there is already some cash present in the account.
- Perform “Cash Out Trade” when there is zero cash balance in the account.
- Perform trade when there is a both cash balance and stocks.

Data flow

- Call after createAccount.

boolean changeAdvisor(int accountId, int newAdvisorId)

Input validation

- accountId or newAdvisorId does not exist in our database.
- accountId or newAdvisorId is set as a negative integer.
- accountId set as advisorId.
- newAdvisorId set as investorId.

Boundary cases

- No such case is present.

Control flow

- Call with newly created account.
- Call twice this method on the same account.

Data flow

- Call after createAccount.

double accountValue(int accountId)

Input validation

- accountId not exist in our database.
- accountId is set as a negative integer.
- accountId set as advisorId.

Boundary cases

- Call when the account only contains cash
- Call when the account contains cash as well as invest in many of the stocks.

Control flow

- Call with newly created account.
- Call with an account containing no cash, and no stocks.
- Call with an account that only holds cash.
- Call with an account that only holds stocks in one particular sector.
- Call with an account that holds stocks in different sectors.

Data flow

- Call after createAccount.

double advisorPortfolioValue(int advisorId)

Input validation

- advisorId not exist in our database.
- advisorId is set as a negative integer.
- advisorId set as investorId.

Boundary cases

- Call when the advisor is assigned to only one account.
- Call when the advisor is assigned to multiple accounts.

Control flow

- Call with the newly created advisor.
- Call when the advisor is assigned to zero investors.
- Call when the advisor is assigned to one investor.
- Call when the advisor is assigned to multiple investors but has the same profiles.
- Call when the advisor is assigned to multiple investors and they have different profiles.
- Call when the advisor is assigned to the investors but investor none of the investors neither hold cash or stocks.
- Call when the advisor is assigned to the investors but investors only hold cash.
- Call when the advisor is assigned to the investors and investors hold both cash and stocks.
- Call when the advisor is assigned to the investor and all investors hold stocks in the same sectors.
- Call when the advisor is assigned to the investor and many of the investors hold stocks in different sectors.

Data flow

- Call after createAccount.

Map<Integer, Double> investorProfit(int clientId)

Input validation

- clientId not exist in our database.
- clientId is set as a negative integer.

- clientId set as advisorId.

Boundary cases
- Call when the client has zero accounts.
- Call when the client has multiple accounts.

Control flow
- Call the newly created client that had no accounts.
- Call clients with only one account and only contain cash.
- Call clients with only one account, have cash, and hold stock in one particular sector.
- Call clients with only one account, have cash, and hold stock in multiple sectors.
- Call clients who have multiple accounts, but only contain cash.
- Call clients who have multiple accounts, have cash, and hold stock in one particular sector.
- Call clients who have multiple accounts, and hold stock in multiple sectors.
- Call a client who has a profit.
- Call a client who had a loss
- Call a client who has multiple accounts and some of them are in profit while others are in loss.

Data flow
- Call after createAccount.

Map<String, Integer> profileSectorWeights(int accountId)

Input validation
- accountId not exist in our database.
- accountId is set as a negative integer.
- accountId set as advisorId.

Boundary cases
- Call the client who holds not a single stock only Cash.
- Call the client who holds cash and stock in different sectors.

Control flow
- Call the newly created account that has no cash and holds no stocks.
- Call the client who invests in only one particular sector.
- Call the client who invests in multiple sectors.
- Call the client who had a partial holding of stock in sectors.

Data flow

- Call after createAccount.

Set<Integer> divergentAccounts(int tolerance)

Input validation

- Call with positive tolerance
- Call with negative tolerance
- Call with zero tolerance

Boundary cases

- Call with a minimum of 1% tolerance.
- Call with a maximum of 100% tolerance.

Control flow

- Call with no accounts in the database.
- Call with only one account in the database.
- Call with accounts strictly follow the assigned profiles.
- Call an account that has a divergent in its profile but has a lower tolerance than the given tolerance.
- Call an account that has a divergent in its profile but has a higher tolerance than the given tolerance.
- Call with mixed accounts, in which some strictly follow assigned profiles, but many profiles are divergent less than or more than a given tolerance.

Data flow

- Call after createAccount.

int disburseDividend(String stockSymbol, double dividendPerShare)

Input validation

- Null or empty stock symbol.
- Set dividendPerShare as zero.
- Set dividendPerShare as negative.
- stockSymbol does not exist in our database.

Boundary cases

- Call with dividendPerShare as the amount is multiple of its current price.

- Call with dividendPerShare as the amount is not a multiple of its current price.

Control flow

- Call with the newly created stock.
- Call when there is no account in the database.
- Call when there are accounts but none of them holds this stock.
- Call when accounts hold stock but all accounts set reinvest as false.
- Call when some accounts set reinvest as true, and the dividendPerShare is a multiple of the current stock price.
- Call when some accounts set reinvest as true, and the dividendPerShare is a not multiple of the current stock price.
- Call when the firm does not hold any stocks of this stock symbol and the firm has to buy new stocks so that they can give the fraction of stocks to different accounts.
- Call when the firm already holds some fraction of stock and the new requirement is less than the current holding.
- Call when the firm already holds some fraction of stock and the new requirement is even more than the current holding.

Data flow

- Call after defineSector and defineStock.
- Call before or after createAccount.

Map<String, Boolean> stockRecommendations(int accountId, int maxRecommendations, int numComparators)

Input validation

- accountId not exist in our database.
- accountId is set as a negative integer.
- accountId set as advisorId.
- Set maxRecommendations as zero.
- Set maxRecommendations as a negative value.
- Set numComparators as zero.
- Set numComparators as a negative value.

Boundary cases

- Call with maxRecommendations and numComparators as 1.
- Call with numComparators as a count of accounts present in the database minus 1.

Control flow

- Call with no account in the database.
- Call with the newly created account.
- Call when accounts contain only cash, no stocks.
- Call when accounts invest in the same stocks but different in quantity.
- Call when accounts hold stocks in different sectors.
- Call when the given account holds all the stocks that are present in the database.
- Call when the given account holds only a few of the stock and the comparison accounts hold many different stocks than the given account.
- Call when half of the comparison accounts hold Stock_A and the remaining do not hold that stock. (50-50)
- Call with maxRecommendations is 1 and more than half of the comparison accounts didn't hold stock_A but given account hold.
- Call with maxRecommendations is 1 and more than half of the comparison accounts hold stock_A but the given account hold does not hold that stock.
- Call with maxRecommendations is 1 and more than half of the comparison accounts didn't hold stock_A but the given account held and at the same time more than half of the comparison accounts hold stock_B but the given account didn't hold that stock.
- Call with numComparators is 10 and maxRecommendations is 1 and 7 of 10 accounts hold stock_A and at the same time 9 of 10 accounts hold stock_B but the given account didn't hold either of the stocks.
- Call with maxRecommendations more than 1.

Data flow

- Call after createAccount.