University of Waterloo - MTE 121 Digital Computation
Assignment 8

Question 1

Problem Description

The bank BigMoney offers a variety of financial products, including loans and checking and savings accounts. You are BigMoney's biggest client, and they would like to store your financial product data. In particular, you currently have the following bank products:

1. A loan with a balance of $20,000 and an interest rate of 6%.
2. A checking account with a balance of $10,000.
3. A savings account with a balance of $20,000, an interest rate of 5%, and a withdrawal charge of $100.

What you need to do

1. Create a BankProduct class with the current balance and account name as private variables, a constructor that sets the initial balance and account name, gettors to return the current balance and the account name, and a settor to reset the account balance.
2. Create a BankLoan class that inherits from the BankProduct class, includes the loan interest rate as a private variable, has an appropriate constructor, and has the following additional functions:
   (a) A function which returns the annual interest on the loan, which is the balance of the loan times the interest rate.
   (b) A function which allows you to make a payment on the loan, which takes a float as input representing the payment and reduces the balance of the loan by the payment amount.
3. Create a BankAccount class that inherits from the BankProduct class, includes a withdrawal charge as a private variable, has an appropriate constructor, and has the following additional functions:
   (a) A function which takes as input a float representing a withdrawal amount, and which reduces the balance of the account by the withdrawal amount plus the withdrawal charge.

(b) A function which takes as input a deposit amount, and which increases the balance of the account by the deposit amount.

4. Create a CheckingAccount class that inherits from the BankAccount class, and has an appropriate constructor in which the withdrawal charge is set to zero.

5. Create a SavingsAccount class that inherits from the BankAccount class, includes the savings interest rate as a private variable, has an appropriate constructor, and has the following additional function:

   (a) A function which increases the account balance by the annual interest on the account, which is equal to the account balance times the interest rate.

6. Write the following additional functions:

   (a) a print function which prints the current balances of the loan, checking account, and savings account to the console

   (b) a payInterest function which reduces the balance in the checking account by the amount of the annual interest payment on the loan.

   (c) a savingsInterest function which increases the balance in the savings account by the amount of the annual interest on the account.

   (d) a depositSalary function which takes a float as input representing the annual salary, and increases the balance in the checking account by this salary.

   (e) a payLoan function which takes a float as input representing the amount to pay towards the loan balance, which decreases both the checking account balance and the loan balance by this amount.

   (f) an addSavings function which takes a float as input representing the amount to add to savings, which decreases the checking account balance by this amount and increases the savings account balance by this amount.

   Functions (a)-(f) should take BankLoan, CheckingAccount, and SavingsAccount object pointers, or some subset of them, as additional inputs. Do NOT pass any of these objects into functions without using pointers!

7. Write a main function which

   - creates pointers to BankLoan, CheckingAccount, and SavingsAccount objects

- uses the pointers to create new objects for each of your bank products (loan, checking account, and savings account)
- updates your savings account balance based on the annual interest it earned
- pays the annual interest on your loan
- deposits your annual salary of $30,000
- makes a payment of $10,000 directly on the loan balance
- transfers $5,000 to your savings account
- outputs the bank product balances to the console

What you need to submit

The code and your output from the console.

Question 2

Problem Description

The company SafeStreets uses a robot to take measurements along the 401 highway of concentrations of dangerous pollutants in the air, including carbon dioxide (CO2), nitrogen oxides (NOx), and sulfur oxides (SOx). SafeStreets wants to (a) store all the measurement data in a database using a linked list data structure, (b) add data as new measurements are taken, (c) remove data from old measurements, and (d) sort the data to determine the locations with the greatest concentrations of each pollutant.

A single measurement by the robot includes the following data:

- The location of the measurement along the highway, measured in kilometers.
- The time at which the measurement was taken, measured in minutes.
- The concentration of CO2, measured in parts per million (ppm).
- The concentration of NOx, measured in ppm.
- The concentration of SOx, measured in ppm.

In this problem you will implement a linked list data structure to add, remove, and sort the measurement data obtained by SafeStreets' robot. The main difference between a linked list and a queue is that queues only allow data to be added at the end and removed at the beginning, whereas lists can add or remove data at any location in the list.

Data for an unknown number of measurements are included in the files "data1.txt" and "data2.txt," where each line includes the location, time, concentrations of CO2, NOx, and SOx, and the index where the measurement should be added to the list. In particular, the first two lines of "data1.txt" are
27.2 12.4 420 1.2 0.2 0
11.3 62.3 410 1.3 0.1 1

What you need to do

1. Create a Measurement class such that each Measurement object stores the data for a single measurement as private variables, and has appropriate gettors, settors, and constructors. The Measurement class should also have a print() function that prints all of its data to the console.
2. Create a Node class that stores pointers to a Measurement object (the data) and the next Node object as private variables, and has appropriate gettors, settors, and constructors.

3. Create a LinkedList class that stores a pointer to the header Node object as a private variable, and has an appropriate constructor.

4. Add the following functions to the LinkedList class:

   (a) A function insert which takes as input data for a single measurement (location, time, and concentrations of CO2, NOx, and SOx) and an index, creates a new Measurement object to store the data, creates a new Node object to point to the Measurement object, and then inserts the new Node at the position index in the list. If the index is larger than the size of the list, the new Node should be inserted at the end of the list.

   (b) A function remove which takes as input an index, removes the Node at that index from the list, and deletes that Node. If the index is larger than the size of the list, no Node should be removed.

   (c) A function print which prints to console the Measurement data for each Node in the list.

   (d) A function swap which takes as input one index, and swaps the positions of the two Node pointers at the positions index and index+1 in the list. No Node objects should be moved in memory, and no new Node objects should be created. If the index is larger than the size of the list, no swapping should occur.

   (e) A function sort which takes as input a string that is either location, time, CO2, NOx, or SOx, and sorts the list in decreasing order (largest at the head, and smallest at the end) based on the choice indicated in the string. No Node objects should be moved in memory, and no new Node objects should be created. This function should call your swap function.

   (f) A function addData that takes as input a file stream representing an input file, reads the measurement data from that file, and inserts a new Node into the list for each line of measurement data in the file at the indicated index.

   (g) A function cutoffIndex that takes as input a float representing a cutoff time in minutes, and returns the index of the first position in the list whose corresponding Measurement data has a time that is earlier than the cutoff time.

   (h) A function removeData that takes as input a float representing a cutoff time in minutes, and removes all Nodes from the list whose corresponding Measurement data has a time that is earlier than the cutoff time. This function should call your sort function.

(i) A function printHead that takes as input an integer representing the first several elements of the list, and prints the locations of that number of elements from the head of the list.

5. Write a main function which

   (a) creates a LinkedList object

   (b) reads the measurement data into the list from "data1.txt"

   (c) removes all data collected prior to 30 minutes

   (d) reads the measurement data into the list from "data2.txt"

   (e) removes all data collected prior to 60 minutes

   (f) prints to console the top 5 locations with the greatest concentrations of (a) CO2, (b) NOx, and (c) SOx.

   (g) sorts the list according to location.

   The list should also be output to the console after steps (b), (c), (d), (e), and (g).

What you need to submit

The code and your output from the console.