CSCE 310 Data Structures & Algorithms

Fall 2004
Steve Goddard

Homework 1, September 2
(Total of 100 points)

Chapters 1 and (portions of) 2

Due: 9:00pm Thursday, September 16, 2004

Exercises from the Text (50 points):

Section 1.1 #4, #5
Section 1.2 #4
Section 1.3 #1
Section 1.4 #10
Section 2.2 #1, #2
Section 2.3 #6

Programming Exercises (50 points):

Given below is a generic List ADT. From this ADT, write three implementations in C++, one using a stack, one using a priority queue, and one using a linked list.

bool isEmpty(FloatList list) – returns true if the list is empty, otherwise false.

FloatList constructList(float newElement, FloatList oldList) – constructs a new list with copies of all of the elements from oldList and newElement
Precondition: oldList is a list with 0 or more elements
Postcondition: the list returned by constructList contains newElement

float firstElement(floatList list) – returns the first element in the list, but does not remove it
Precondition: isEmpty(list) == false
Postcondition: none

FloatList restOfList(FloatList oldList) – returns a copy of list that contains every element except firstElement(oldList)
Precondition: oldList is a list with 1 or more elements
Postcondition: newList = restOfList(oldList)
newList contains every element in oldList except the first element

Note: This is an abstract definition of the operations that the List ADT should perform. It is your job to create classes for this ADT which implement at least these features.

What the program does

We are interested in generating forecasts of sales data based on the three calculations below. Your task is to write a program called **forecast** that will take as its arguments the name of a data file and the name of a product. From these arguments, the program will extract the data needed to perform calculations for the specified product. You must use the generic ADT List defined above. For the data file, use the comma-delimited file similar to the one for Homework 0.

The output of the program, if the product is listed in the data file, should be exactly as below (truncate values to integers for the output, but not during the calculations, as this would decrease the accuracy of the results):

```
Product: Wheel
Simple average: 2491
Slope: 2019
Linear regression: 2148
```

If the product is not found in the file, the output should be exactly as below:

```
Product productName not found in fileName
```

Where *productName* is the name of the product and `fileName` is the name of the input file. For example, if the product Wheel is not found in the file Forecast, the output should be exactly as below:

```
Product Wheel not found in Forecast
```

Desired forecasts

I. **Simple average** – use the average number of historical products needed in the file given to estimate the next period's value.

$$Forecasted\ value = \frac{\sum Historical\ values}{Number\ of\ \ periods}$$

II. **Slope**– use the slope of the line between the first and last points to estimate the future value

$$Slope = \frac{First\ \ historical\ \ value - Last\ \ historical\ \ value}{First\ \ period\ \ number - Last\ \ period\ \ number}$$

$$Intercept = First\ \ historical\ \ value - (Slope)(First\ \ period\ \ number)$$

$$Forecasted\ \ value = (Slope)(Number\ \ of\ \ \ periods + 1) + Intercept$$

III. **Linear regression** – use a linear regression to find a best-fitting line, and then evaluate the point on the line that represents the next period to find its forecasted value.

$$SumX = \sum Period\ \ number$$

$$SumY = \sum Historical\ \ values$$

$$SumXY = \sum ((Period\ \ number)(Historical\ \ value))$$

$$SumXSquared = \sum ((Period\ \ number)(Period\ \ number))$$

$$Slope = \frac{(Number\ \ of\ \ \ periods)(SumXY) - (SumX)(SumY)}{(Number\ \ of\ \ \ periods)(SumXSquared) - (SumX)(SumX)}$$

$$Intercept = \frac{SumY - (Slope)(SumX)}{Number\ \ of\ \ \ periods}$$

$$Forecasted\ \ value = (Slope)(Number\ \ of\ \ \ periods + 1) + Intercept$$

Your program should be able to be compiled 3 different ways, one for each of the three ADT Lists. I should be able to type “make stack”, “make queue”, and “make list” to make the program **forecast** coinciding with each of the three ADTs. See the TA for help with creating makefiles.

Analysis

Examine the output files for each of the three implementations. Prepare a report in which the following are considered.

- Do the results match your expectations
- Are any of the results different from the rest? If so, why?
- Compute the worst-case asymptotic complexity of the algorithm for each computation, ignoring the ADT implementations.
- Compute the worst-case asymptotic complexity of the algorithm for each computation, considering the ADT implementations.
- Give a critical review of the generic List ADT specification, including changes that might seem appropriate.

This analysis report should also (as in HW0) describe how your program works, what design choices (if any) were made and why, and any known defects. You may also discuss enhancements you think should be made, but were beyond the scope of the assignment (i.e., inadequacy of the specifications).

This problem will be scored at follows:

Program correctness (as defined)	35%	(17.5 pts)	
Quality of design/readability	20%	(10 pts)	
In-line documentation/coding standard	15%	(7.5 pts)	
Design and analysis document	25%	(12.5 pts)	This is extremely important considering the objective of this assignment
Thoroughness of test cases	5%	(2.5 pts)	