2022 Fall Data Structure - HW5

Handwriting

1. (20%) Imagine we have the list shown in Figure 5-26 implemented as a linked list. As discussed in "List Search," in Section 5.2, the search needs to be able to pass back both the location of the predecessor (`pPre`) and the location of the current (`pLoc`) node based on search criteria.

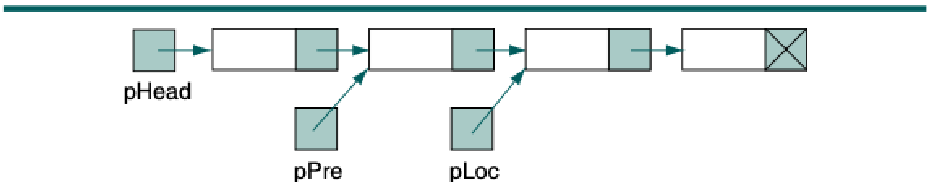


FIGURE 5-26 Linked List Implementation for Exercise 2

The following code to set `pPre` and `pLoc` contains a common error. What is it and how should it be corrected?

```
pLoc = pLoc->link
pPre = pPre->link
```

(*Hint:* What are the contents of these pointers at the beginning of the search?)

2.(20%) What would happen if we applied the following statements to the two lists in Exercise 7?

```
1 set temp to listl
2 loop (temp link not null)
   1 set temp to temp link
3 end loop
4 set temp link to list2
```

- 將Figure 5-29的兩個list進行題目敘述之操作後，畫出最終兩個list的各個node跟link的連結情況

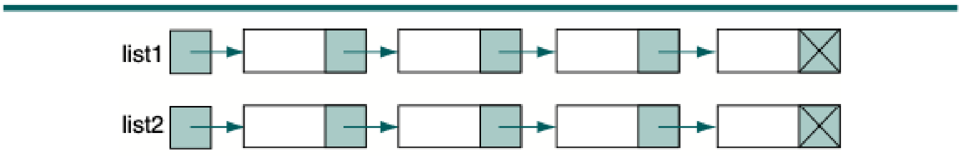
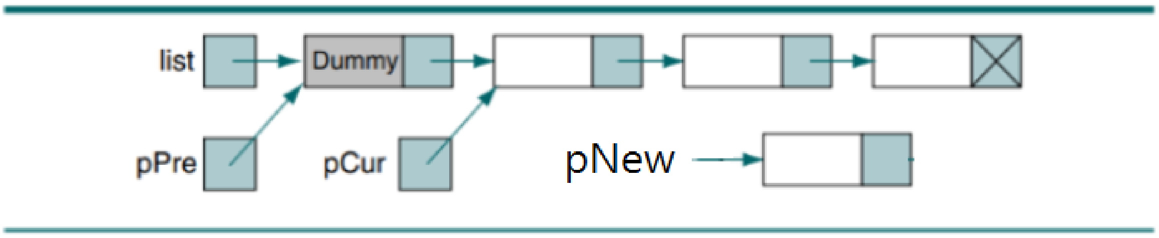


FIGURE 5-29 Linked Lists for Exercise 7

3. (60%)
According to the example of the linked list with a dummy node.

- pCur is a pointer which points to any node on the list except the dummy node.
- pPre is a pointer which points to the previous node of the “pCur node”.
- pNew is a pointer which points to the new node.

Imagine we implement a list using a dummy node at the beginning of the list. The dummy node does not carry any data. It is not the first data node, it is an empty node.



Please answer the following question:

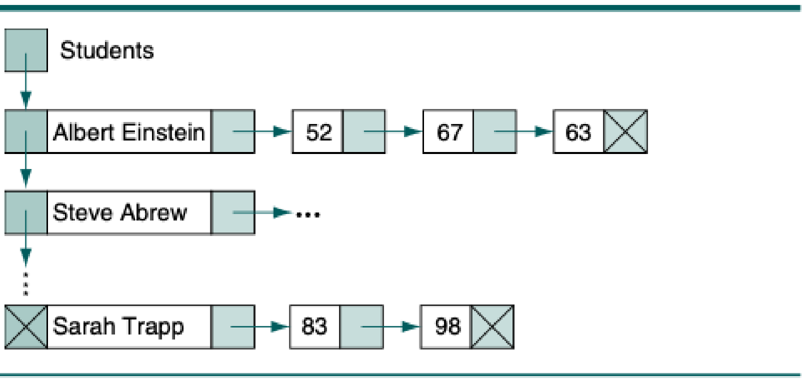
(a) Write the pseudocode to “delete” the “pCur node”.
(b) Write the pseudocode to “add” the node behind the “pPre node”.
(c) Does the dummy node simplify “delete” and “add” operations on the list? How?

請按照下列規定回答:
(a), (b)小題均假設非empty list。
(a)小題需要寫三行程式碼，前兩行是指標的變動，第三行是釋放記憶體，記得pCur要指到被刪除node的下個node。
(b)小題需要寫三行程式碼，都是指標的變動，一樣記得pCur要指到”pNew node”上。

Programming

4.(25%) Write a program to read a list of students from a file and create a list. The program should use a linked list for implementation. Each node in the linked list should have the student's name, a pointer to the next student, and a pointer to a linked list of scores. There may be up to four scores for each student. The structure is shown in Figure 5-33.

FIGURE 5-33



The program should initialize the student list by reading the students' names from the text file and creating null score lists. It should then loop through the list, prompting the user to enter the scores for each student. The scores' prompt should include the name of the student.

After all scores have been entered, the program should print the scores for each student along with the score total and the average score. The average should include only those scores present.

The data for each student are shown in Table 5-3.

TABLE 5-3

Student name	Score 1	Score 2	Score 3	Score 4
Albert Einstein	52	67	63	
Steve Abrew	90	86	90	93
David Nagasake	100	85	93	89
Mike Black	81	87	81	85
Andrew Dijkstra	90	82	95	87
Joanne Nguyen	84	80	95	91
Chris Walljasper	86	100	96	89
Fred Albert	70	68		
Dennis Dudley	74	79	77	81
Leo Rice	95			
Fred Flintstone	73	81	78	74
Frances Dupre	82	76	79	
Dave Light	89	76	91	83
Hua Tran	91	81	87	94
Sarah Trapp	83	98	94	93

- 請使用Linked List依照Figure 5-33實作，可以根據個人需求在node中儲存額外資訊，若無使用Linked List結構則不予計分
- 不允許使用C++ STL Container
- 學生姓名不會超過128個字元
- Input: 讀取同目錄下[input.txt]
- Output: 將結果在Terminal上印出
- 詳細輸入/輸出格式規範如下

輸入格式	輸出格式
第一行為**n**，表示有**n**名學生，接下來**n**行為學生姓名，學生姓名結束後則有**2n**行表示學生個別成績。每名學生成績包含兩行資料，第一行為成績個數**k**，第二行為**k**個成績，中間由空白區隔。	印出每個學生的姓名及平均分數到小數點後第二位數，由空白區隔。
範例輸入 **#1** 2 Albert Einstein Steve Abrew 3 52 67 63 4 90 86 90 93	範例輸出 **#1** Albert Einstein 60.67 Steve Abrew 89.75
範例輸入 **#2** 3 Fred Albert Dennis Dudley Leo Rice 2 70 68 4 74 79 77 81 1 95	範例輸出 **#2** Fred Albert 69.00 Dennis Dudley 77.75 Leo Rice 95.00

5.(25%) Write a program that adds and subtracts polynomials. Each polynomial should be represented as a list with linked list implementation. The first node in the list represents the first term in the polynomial, the second node represents the second term, and so forth.

Each node contains three fields. The first field is the term's coefficient. The second field is the term's power, and the third field is a pointer to the next term. For example, consider the polynomials shown in Figure 5-34. The first term in the first polynomial has a coefficient of 5 and an exponent of 4, which is then interpreted as $5x^4$.

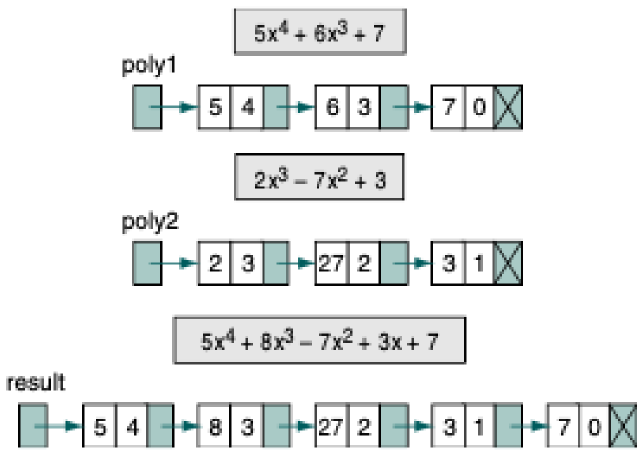


FIGURE 5-34 Example of Polynomials for Project 29

The rules for the addition of polynomials are as follows:

a. If the powers are equal, the coefficients are algebraically added.
b. If the powers are unequal, the term with the higher power is inserted in the new polynomial.
c. If the exponent is 0, it represents x^0, which is 1. The value of the term is therefore the value of the coefficient.
d. If the result of adding the coefficients results in 0, the term is dropped (0 times anything is 0).

A polynomial is represented by a series of lines, each of which has two integers. The first integer represents the coefficient; the second integer represents the exponent. Thus, the first polynomial in Figure 5-35 is

```
5   4
6   3
7   0
```

To add two polynomials, the program reads the coefficients and exponents for each polynomial and places them into a linked list. The input can be read from separate files or entered from the keyboard with appropriate user prompts. After the polynomials have been stored, they are added and the results are placed in a third linked list.

The polynomials are added using an operational merge process. An operational merge combines the two lists while performing one or more operations—in our case, addition. To add we take one term from each of the polynomials and compare the exponents. If the two exponents are equal, the coefficients are added to create a new coefficient. If the new coefficient is 0, the term is dropped; if it is not 0, it is appended to the linked list for the resulting polynomial. If one of the exponents is larger than the other, the corresponding term is immediately placed into the new linked list, and the term with the smaller exponent is held to be compared with the next term from the other list. If one list ends before the other, the rest of the longer list is simply appended to the list for the new polynomial.

Print the two input polynomials and their sum by traversing the linked lists and displaying them as sets of numbers. Be sure to label each polynomial.

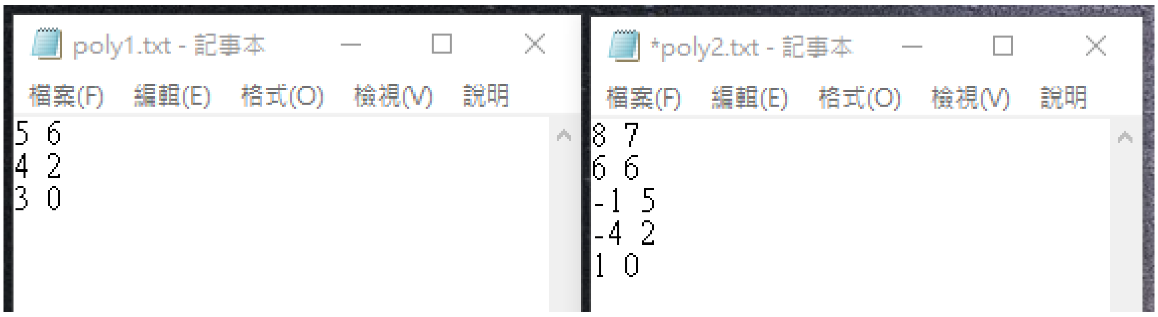
Test your program with the two polynomials shown in Table 5-4.

Polynomial 1		Polynomial 2	
Coefficient	Exponent	Coefficient	Exponent
7	9	–7	9
2	6	2	8
3	5	–5	7
4	4	2	4
2	3	2	3
6	2	9	2
6	0	–7	1

TABLE 5-4 Text Data for Project 29

- 請使用Linked List依照Figure 5-34實作，可以根據個人需求在node中儲存額外資訊，若無使用Linked List結構則不予計分
- 可以使用STL list，不可使用其餘STL Container
- Input：讀取同目錄下的[poly1.txt]與[poly2.txt]，格式如題目中敘述，不必考慮Float的情況
- Output：在Terminal中Print出Input的兩個Polynomial及運算後的結果
- 輸出格式請參照Sample output，不要再加其他東西，若有則照格式錯誤扣分
- 運算出的結果需存入新的Link List中，不可在運算時直接Print出

Input :


poly1.txt - 記事本
檔案(F) 編輯(E) 格式(O) 檢視(V) 說明
*poly2.txt - 記事本
檔案(F) 編輯(E) 格式(O) 檢視(V) 說明

```
5 6
4 2
3 0
```

```
8 7
6 6
-1 5
-4 2
1 0
```

Output :

```
Poly1: 5x^6 + 4x^2 +3
Poly2: 8x^7 + 6x^6 - x^5 - 4x^2 + 1
result: 8x^7 + 11x^6 - x^5 + 4
```

6.(25%) In older personal computers, the largest integer is 32,767 and the largest long integer is 2,147,483,647. Some applications, such as cryptography and security algorithms, may require an unbounded integer. One way to store and manipulate integers of unlimited size is by using a linked list. Each digit is stored in a node of the list. For example, Figure 5-35 shows how we could store a five-digit number in a list.

Although the list in Figure 5-35 is represented as moving from right to left, there is no physical direction in a list. We represent it in this way to clarify the problem.

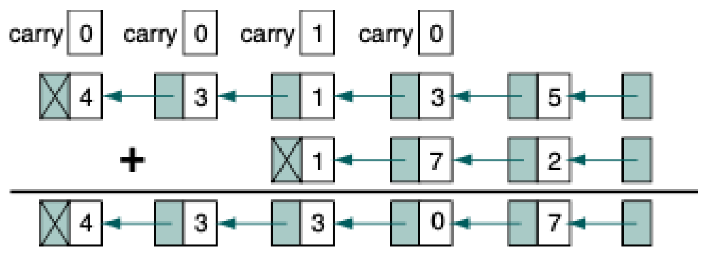


FIGURE 5-35 Integer Stored in a List for Project 30

To add two numbers, we simply add the corresponding digit in the same location in their respective lists with the carry from the previous addition. With each addition, if the sum is greater than 10, we need to subtract 10 and set the carry to 1. Otherwise, the carry is set to 0.

Write an algorithm to add two integer lists. Design your solution so that the same logic adds the first numbers (units position) as well as the rest of the number. In other words, do not have special one-time logic for adding the units position.

- 讀取同目錄下[input.txt]，其有兩個超過"int"的「正整數」，之間以一個space隔開
- 如FIGURE 5-35，應使用list結構暫存這兩個數字並計算其和
- 可以使用STL list，不可使用其餘STL Container
- 將兩數字的和Print至Terminal上(不需要有其他輸出)

Sample Input (in input.txt):

```
12345678987654321 98765432123456789
```

Sample Output:

```
111111111111111110
```

7.(25%) Write a program to process stock data. The stock data should be read from a text file containing the following data: stock code, stock name, amount invested (*xxx.xx*), shares held, and current price. Use the Internet or your local paper to gather data on at least 20 stocks. (You may use mutual funds in place of stocks.)

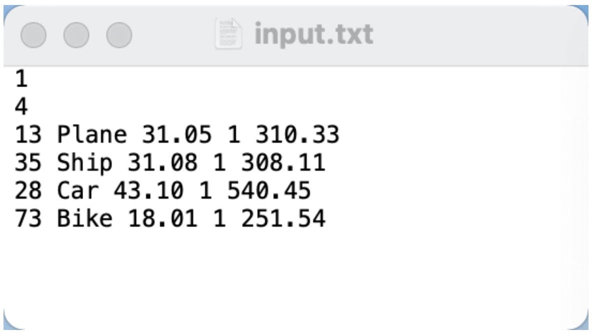
As each stock is read, insert it into a doubly linked multilinked list. The first logical list should be ordered on the stock code. The second logical list should be ordered on the gain or loss for the stock. Gain or loss is calculated as the current value of the stock (shares held times current price) minus the amount invested. Include at least one loss in your test data.

After building the lists, display a menu that allows the user to display each logical list forward or backward (a total of four options). Each display should contain an appropriate heading and column captions.

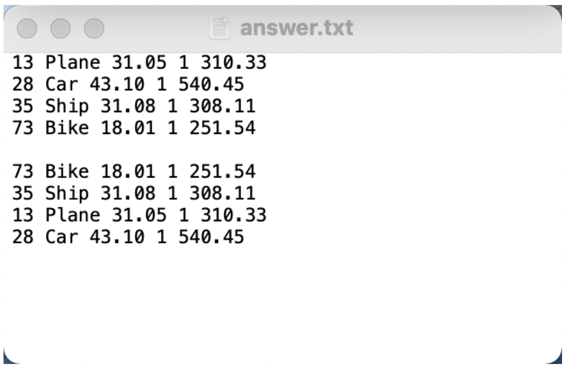
Run your program and submit a list of your input data and a printout of each display option.

- Input: 請讀取[input.txt]作為初始參數，參數之間會以' '(空白符)區隔
- Output: 請將Data list資料輸出為[answer.txt]，請依序呈現以下欄位：Code/ Name/ Amount/ Shares Held/ Price（每個欄位之間請以' '(空白符)區隔），請勿輸出額外資訊在answer.txt中（例如：Show list:/ Answer: ...）
- 請不要印出任何東西在terminal上
- 本題請以doubly linked multilinked list實作，可以使用STL list，不可使用其餘STL Container
- 排序方式為以下兩種：
  1. Display list sorted ascending by Stock code and display list sorted ascending by Gain/Loss
  2. Display list sorted descending by Stock code and display list sorted descending by Gain/Loss

[input.txt]
第一行：指定排序方式（為上面敘述的兩種方式）
第二行：Data list的資料比數（0<n<1000）
其餘行：Data list資料（依序為：Code/ Name/ Amount/ Shares Held/ Price）
(0< Code <10^6)
(0< Amount <10^6)
(0< Shares Held <10^6)
(0< Price <10^6)


```
1
4
13 Plane 31.05 1 310.33
35 Ship 31.08 1 308.11
28 Car 43.10 1 540.45
73 Bike 18.01 1 251.54
```

[answer.txt]
Data list資料欄位請依序呈現：Code/ Name/ Amount/ Shares Held/ Price（每個欄位之間請以’ ’(空白符)區隔）
*Gain or Loss = (Shares Held) * Price - Amount
*Amount: 呈現至小數點下第二位
*Price: 呈現至小數點下第二位


```
13 Plane 31.05 1 310.33
28 Car 43.10 1 540.45
35 Ship 31.08 1 308.11
73 Bike 18.01 1 251.54

73 Bike 18.01 1 251.54
35 Ship 31.08 1 308.11
13 Plane 31.05 1 310.33
28 Car 43.10 1 540.45
```

Submission - Deadline: 2022/11/25 13:20

題目形式：

- 手寫題可以用手寫拍照、打字的方式完成，但最後要統一轉成.pdf檔繳交
  檔名為HW5_學號.pdf。例如：HW5_0123456.pdf
- 程式題則繳交程式原始碼(.c檔/.cpp檔/.h檔 if needed)
  檔名為HW5_題號_學號.c / .cpp。例如：HW5_4_0123456.c / .cpp / .h

繳交方式：

- 將上述共五個檔案及h檔(if needed)(手寫題pdf檔*1+程式題c/cpp檔*4)直接上傳至e3
- 檔名 / 格式錯誤者扣該次作業總分10分。
- 程式部分輸出格式請照作業說明， 若不同會酌量扣分。

收作業規則：

- 遲交一個禮拜內分數打七折， 超過一個禮拜即不接受補交。
- 遲交期限內僅接受原本沒交作業的同學補交，不接受先前交過作業的同學再次補交，
  若要修改答案請在繳交期限內修改完畢。
- 請務必重新整理，確認檔案已成功上傳至e3。

如有任何問題，麻煩從e3來信給所有助教。