HW3

If your program takes more than a certain reasonable amount of time to finish, you get 0 points. Every program should be compilable and runnable in cspro. Every program should be written in C (at this time, No C++) and satisfy the requirements in the program specification. Copy check will be done.

1. Textbook (2nd Eds) p.170 #7. Write a user-interactive program that **reads** two polynomials from the stdin, which is represented by **circular lists with header nodes**, and **prompts** to the user a selection for the operation to be applied to the two polynomials and **outputs** the proper result(s) to the stdout. (For example, after reading the two polynomials a(x) and b(x), the program asks choosing an operation. If the user chooses addition, then it outputs a(x)+b(x). If the choice is multiplication, it outputs a(x)*b(x), etc.. If the choice corresponds to evaluation at x=x0, then it outputs the values of a(x0) and b(x0) at x0 for each input polynomial.) As in the description of the problem in the book, you need to the following:

Design and build a linked allocation system to represent and manipulate polynomials. You should use circularly linked lists with header nodes. Each term of the polynomial will be represented as a node, using the following structure:

coef	expon	link

In order to erase polynomials efficiently, use the available space list and associated functions discussed in this section.

Write the following functions:

pread. Read in a polynomial and convert it to its circular representation. Return a pointer to the header node of this polynomial.

pwrite. Output the polynomial using a form that clearly displays it.

padd. Compute $c = a + b$. Do not change either a or b.

psub. Compute $c = a - b$. Do not change either a or b.

pmult. Compute $c = a*b$. Do not change either a or b.

eval. Evaluate a polynomial at some point, a, where a is a floating point constant. Return the result as a floating point.

perase. Return the polynomial represented as a circular list to the available space list.

조건: polynomial A, B 는 모두 최대 50 개의 Term 을 가질 수 있음, 가장 큰 exp 부터 가장 작은 exp 까지 차례대로 입력

예제
입력 및 **출력**

Enter number of terms in polynomial A: 3
Enter terms for A (coef exp):
3 4
2 2
1 0

Enter number of terms in polynomial B: 3
Enter terms for B (coef exp):
1 3
-2 2
5 0

Choose operation:
1. Addition
2. Subtraction
3. Multiplication
4. Evaluation
5. Quit
Enter choice: 1
Result: 3x^4 + 1x^3 + 6

Choose operation:
1. Addition
2. Subtraction
3. Multiplication
4. Evaluation
5. Quit

```
Enter choice: 2
Result: 3x^4  - 1x^3 + 4x^2 -4

Choose operation:
1.  Addition
2.  Subtraction
3.  Multiplication
4.  Evaluation
5.  Quit
Enter choice: 3
Result: 3x^7 - 6x^6 + 2x^5 + 11x^4 + 1x^3 + 8x^2 + 5

Choose operation:
1. Addition
2. Subtraction
3. Multiplication
4. Evaluation
5. Quit
Enter choice: 4
Enter x value: 2
A(2) = 57
B(2) = 5

Choose operation:
1. Addition
2. Subtraction
3. Multiplication
4. Evaluation
5. Quit
Enter choice: 5
Good Bye
```

2. Given a matrix in "input.txt" file given as in Figure 4.20, write a program that reads the matrix by the linked representation as in Figure 4.19 (강의자료 ch4 188p) and computes the transpose of the matrix and writes the answer into "output.txt" file. You must write a function
matrix_pointer mtranspose(matrix_pointer node) to compute the transpose.

입력: 첫 번째 줄에는 [matrix row] [matrix column] [0 이 아닌 원소의 개수]. 두 번째 줄부터 [row] [column] [value]. 각 값은 띄어쓰기로 구분.
출력: 첫 번째 줄은 입력과 같이 [matrix row] [matrix column] [0 이 아닌 원소의 개수]를 띄어쓰기로 구분하여 출력. 두 번째 줄부터 입력 matrix 를 transpose 한 matrix 출력 (**반드시 row major order 이며, row 가 같을 경우 column major**).
조건: 교재(강의자료) mread()와 mwrite()를 적절히 변형하여 작성. 전역 변수는 기존 matrix 의 header node 를 저장하는 hdnode[]와 transpose 한 matrix 의 header node 를 저장하는 hdnode_t[]만 가능. MAX_SIZE 는 50(교재와 동일)으로 가정.

예제	
입력 (input.txt)	출력 (output.txt)
4 5 6 0 2 11 0 4 6 1 0 12 1 1 7 2 1 -4 3 3 -15	5 4 6 0 1 12 1 1 7 1 2 -4 2 0 11 3 3 -15 4 0 6