Assignment 0– Implementation of basic data structures

1. (Implementation of a doubly linked list) We can represent a polynomial with a form of a doubly linked list using the following structure:

```
struct Node {

            int degree;

            int coefficient;

            struct Node *next;   // link to the next node. NULL when the last node

            struct Node *prev;   // link to the previous node. NULL when the first node

};
```

The structure 'Node' represents a term in a polynomial. Using the 'Node' structure, implement the followings:

```
Input (degree) (coefficient): 2 2

Input (degree) (coefficient): 1 1

Input (degree) (coefficient): 3 3

Input (degree) (coefficient): -1 -1

Done!!
```

1. Node *inputpoly(void): it takes at least a pair of (coefficient, degree) data using keyboard and outputs the corresponding polynomial represented as a doubly linked list with the above Node structure. We suppose only integral numbers are taken. The left is an example of running the inputpoly function. If both degree and coefficient inputs are negative, the function returns the point to the first node of the polynomial. Disregard the final input. Also, take input again if one of the inputs is negative and the other is positive.

2. void printNode(Node *inp): it takes the head node of the doubly linked list and print out the representing polynomials. The left figure is the execution example of running printNode(inputpoly());

```
Input (degree) (coefficient): 2 2

Input (degree) (coefficient): 1 1

Input (degree) (coefficient): 3 3

Input (degree) (coefficient): -1 -1

Done!!

1 x + 2 x^2 + 3 x^3
```

3. Node *multiply(Node *a, Node *b): This function multiplies the two polynomials ‘a’ and ‘b’ and returns the multiplication result.

* Your program should use memory efficiently. No memory leaks should occur.

* Your program should not be crashed for bogus inputs.

* You need to implement main() function to be used to check your implementation works well.

Grading criteria	Percentages
Implementation of each functions	60%
Explanation of your implementation with comments	20%
Quality of your report	20%

** Deliverables

- Report
    - ■ Overview of your implementation
    - ■ Explanation of important part of your code
    - ■ Way to compile your code in the ubuntu environment.
- Source code (mandatory) and makefile (optional: only if you use it)

** Environmental requirements

1) You implement the assignment in Ubuntu operating system

    A. Need to demonstrate how to compile and execute it in your report

2) You use gcc or g++ compiler for complilation