Lab 1 CS201 (2020)

Read the assignment carefully. There are 3 exercises in this assignment; 3 programs to be submitted.

- Single .c file for each exercise. Program must be compilable using gcc as we will be likely using linux platform to evaluate your assignments
- You MUST follow the Filename format :
    - L01a_<Your First name>_<Entrynumber>_CS201_2020.c for part A,
        - For e.g. L01a_Raman_2019CSB9999_CS201_2020.c for part A exercise
    - L01b_<Your First name>_<Entrynumber>_CS201_2020.c for part B,
        - For e.g. L01b_Raman_2019CSB9999_CS201_2020.c for part B exercise
    - L01c_<Your First name>_<Entrynumber>_CS201_2020.c for part C,
        - For e.g. L01c_Raman_2019CSB9999_CS201_2020.c for part C exercise
- Input outputs that you tried or used for testing your code can be included in your program file at end within comments (using /* */).
- If you know you code is not working for some cases etc, do mention that clearly in comments section at the beginning of your code (using /* ….. */)
- There shall not be any identifiable information within the program. No reference to your name, roll number etc. shall appear in your code.
- Deadline are as follows:
    - For part A: 4th Sep 2020, 11:59:59 PM
    - For part B: 6th Sep 2020, 11:59:59 PM
    - For part C: 9th Sep 2020, 11:59:59 PM
    - While submitting, exercise due care and caution to submit the appropriate code (your last updated program, not your intermediate trial codes) and with proper filename.
    - Instructions how and where to submit the assignments on Moodle/via Google Form or Google Classroom will be shared with you soon.
- No Plagiarism Please. Please note that we may do plag check now or later even after endsem, and you will be then (severely) penalized. You must do your own coding and neither take nor provide codes from/to anyone else.
    - Those who complete their works, they are kindly requested to not share their codes in name of friendship or so. You may get severely penalized for the same and by sharing codes, you are not really helping them either.
    - We sincerely request and hope that you will try your level best and not adopt any unfair means so that we need NOT go through the painful experience of penalizing you.

==========**EXERCISE A (Using Linked List concepts)** ==========

Programming Assignment (Very Basic Assignment):

1. Write a program to perform basic operations as mentioned below on a linked list, where each node consists of an integer data value and a pointer to next node. The program would ask for user choice (an integer value) and perform the operation as per user choice. Assume "head" is the pointed pointing to the head of the linked list. Initially it is Null as linked list is empty. You may preferably write all these operations within the main function only. You may write different functions for different operations (It is recommended but it is not mandatory.

Choice	**Operation**	**Details**
0	Exit	Exit from the program. A good programmer would free all the space before exiting
1	Insert in beginning	User is prompted to input data value and then a new node with that data value is added in the beginning of the linked list.
2	Insert at end	User is prompted to input data value and then a new node with that data value is added at the end of the linked list.
3	Delete first node	Delete first node of the linked list. If no such node, print "Cannot delete as NO nodes in the linked-list"
4	Delete last node	Delete last node of the linked list. If no such node, print "Cannot delete as NO nodes in the linked-list"
5	Delete specific node	User is prompted to input data value and then there is search to find the node with that data value in the linked list. If there is any such one node in the linked list, delete that node from the linked list. If there are many such nodes, consider only the first such node. If no such node exists, print "Cannot delete as no such node in the linked-list"
6	Find node	User is prompted to input data value and then there is search to find the node with that data value in the linked list. If there is any such one node in the linked list, print the position of that node in the linked list and also ~~its memory address~~ how many units apart the allocated memory of this node is from that of current head of the linked list.. If there are many such nodes, consider the first occurrence that is identified. If no such node exists, print "Cannot find any such node in the linked-list"
7	Size	Print the number of nodes in the linked list. Print 0 if no nodes in the linked list or say when head==NULL
8	Display	Print all nodes values in the linked list in the manner as mentioned below in the sample input/output below If linked list is empty, Print "Empty"
Any other integer	Wrong Input	Print "Wrong Input. \nEnter your choice again: "

Sample Input / Output (Note that user input values are mentioned in bold here for your convenience):

Enter the choice:
1
Enter the data value:
10
Enter the choice:
2
Enter the data value:
20
Enter the choice:
1
Enter the data value:
16
Enter the choice:
7
Size of the linked list = 3
Enter the choice:
8
Linked list: 16 => 10 => 20
Enter the choice:
9
Wrong Input.
Enter your choice again:
3
Enter the choice:
8
Linked list: 10 => 20
Enter the choice:
2
Enter the data value:
15
Enter the choice:
1
Enter the data value:
40
Enter the choice:
4
Enter the choice:
8
Linked list: 40 => 10 => 20
Enter the choice:
6
Enter the data value:
10
Value 10 node is at position 2 in the linked list.
Difference in its allocated memory to that of current head : -6

```
Enter the choice:
5
Enter the data value:
10
Enter the choice:
6
Enter the data value:
10
        Cannot find any such node in the linked-list
Enter the choice:
8
        Linked list: 40 => 20
Enter the choice:
7
        Size of the linked list = 2
Enter the choice:
0
```

//Note: We will also share with you some sample input output files and you are requested to follow the proper input output format, As we will be using similar files and the commands of the following nature (in some automated or semi-automated way) to evaluate your submitted program:

$gcc l01a_Raman_2019CSB9999.c -o l01a_2019CSB9999

$./l01a_2019CSB9999 < l01a_input1 > l01a_2019CSB9999_output1

$./l01a_2019CSB9999 < l01a_input2 > l01a_2019CSB9999_output2

$./l01a_2019CSB9999 < l01a_input3 > l01a_2019CSB9999_output3

$diff l01a_2019CSB9999_output1 l01a_output1

$diff l01a_2019CSB9999_output2 l01a_output2

$diff l01a_2019CSB9999_output3 l01a_output3

==========**EXERCISE B (Use Linked List concepts)** ==========

Using linked list, write a program to add, subtract and evaluate polynomials: P1(x), P2(x), P3(x) and P4(x) where P1(x) and P2(x) are input polynomials and P3(x) = P1(x)+P2(x) and P4(x)=P1(x)-P2(x). Each node in the linked list correspond to a term in the polynomial. So, in your node structure - you may have two data components – int pow, coeff ; and one pointer to the next node.

Input Format:

First line mentions K i.e. the number of test cases. Then there are three lines for each test case, In the first two lines of a test case, First number indicate the highest degree of polynomials N and then there are N+1 integers which are the coefficients of polynomial terms in descending order.

In the third (and last line) of a test case, there is one integer i.e. value of x for which you need to evaluate the polynomials. (Constraints: $0 \le K \le 50$, $0 \le N \le 9$, $-2 \le x \le 2$, and Input coefficient terms would be between -100 to +100; Assume you can safely do calculations for each polynomial term without worrying about underflow/overflow issues).

Sample Input 1:

```
1
7 1 0 0 0 10 -3 0 1
3 4 0 0 -2
2
```

Explanation of Input Format (Considering 2nd Polynomial mentioned above)

3	4	0	0	-2
Highest degree of polynomial	Coefficient of x^3	Coeff of x^2	Coeff of x^1	Coeff of x^0

As per above format: 7 1 0 0 0 10 -3 0 1===> $x^7 + 10x^3 - 3x^2 + 1$

3 4 0 0 2 ===> $4x^3 - 2$

2 ===> value of x should be in range of -2 to 2.

Sample Output 1:

```
P1(x) : 1x^7 + 10x^3 - 3x^2 + 1
P2(x) : 4x^3 - 2
P3(x) = P1(x) + P2(x) : 1x^7 + 14x^3 - 3x^2 - 1
P4(x) = P1(x) - P2(x) : 1x^7 + 6x^3 - 3x^2 + 3
P1(2) = 197
P2(2) = 30
P3(2) = 227
P4(2) = 167
```

Note: There is a single space before and after =, +, -, and : in the output.

Explanation of Output Format:

1x^7 + 14x^3 - 3x^2 - 1 ====> $f_1(x)+f_2(x)$
1x^7 + 6x^3 - 3x^2 + 3 ====> $f_1(x) - f_2(x)$
197 30 227 167 ====> $f_1(2)$ $f_2(2)$ $(f_1(2) + f_1(2))$ $(f_1(2) - f_2(2))$

Note that there should be **NO** nodes in the linked list for the polynomial terms having 0 coefficient value. In the above example, the polynomials f1 and f2 were to be represented using 4 and 2 nodes in the linked list, respectively.

Sample Input 2	**Sample Output 2**
3 7 1 0 0 0 10 -3 0 1 3 4 0 0 -2 2 7 1 0 0 0 10 -3 0 1 3 4 0 0 -2 1 7 1 0 0 0 10 -3 0 1 7 1 0 0 0 10 -3 0 1 2	P1(x) : 1x^7 + 10x^3 - 3x^2 + 1 P2(x) : 4x^3 - 2 P3(x) = P1(x) + P2(x) : 1x^7 + 14x^3 - 3x^2 - 1 P4(x) = P1(x) - P2(x) : 1x^7 + 6x^3 - 3x^2 + 3 P1(2) = 197 P2(2) = 30 P3(2) = 227 P4(2) = 167 P1(x) : 1x^7 + 10x^3 - 3x^2 + 1 P2(x) : 4x^3 - 2 P3(x) = P1(x) + P2(x) : 1x^7 + 14x^3 - 3x^2 - 1 P4(x) = P1(x) - P2(x) : 1x^7 + 6x^3 - 3x^2 + 3 P1(1) = 9 P2(1) = 2 P3(1) = 11 P4(1) = 7 P1(x) : 1x^7 + 10x^3 - 3x^2 + 1 P2(x) : 1x^7 + 10x^3 - 3x^2 + 1 P3(x) = P1(x) + P2(x) : 2x^7 + 20x^3 - 6x^2 + 2 P4(x) = P1(x) - P2(x) : 0 P1(2) = 197 P2(2) = 197 P3(2) = 394 P4(2) = 0

Remarks: There are three test cases in 2nd Sample Input.

Remember to appropriate free the memory space after each test case.

// Input would be read from terminal i.e. Stdin and not from any file.

==========EXERCISE C (Use Linked List concepts) ==========

Consider a new function # (somewhat related to factorial !) that is defined for any non-negative integer **n** as follows:

n# = (1^1) * (2^2) * (3^3) * .....* ((n-1) ^ (n-1)) * (n^n)

//Remember that **n!** = 1*2*3*4*…..*(n-1)*n and is different **0# = 1 1#=1 2#=4 2!=2**

Given a non-negative number n and positive number k (atmost three digit), Write a program in C that computes and provides the value of n# ,
Z_n i.e. the number of end zeroes in n# value, and
$Q_{n,k}$ i.e. the number of times k pattern appears in n#.

Constraints and Sample input/output -
First line of the input file mentions positive value T that denotes the number of test cases. Then follows T lines and each of them mentions two values (single space separated) n_i and k_i
Constraints/Assumptions:
All these input values are non-negative.
T<500, n_i <= 200 and k_i <= 999

Input would be read from terminal i.e. Stdin and not from any file.

Input Format:	**Output format (T lines each having 3 values, single space separated):**
T N_1 k_1 N_2 k_2 ... ... N_T k_T	Z_{N1} $Q_{N1,k1}$ N_1# Z_{N2} $Q_{N2,k2}$ N_2# ... ... Z_{NT} $Q_{NT,kT}$ N_T#
Sample Input: 6 3 1 5 7 9 941 9 09 7 3 9 22	**Sample Output** 0 1 108 5 0 86400000 5 2 215779412229418562091680268288000000 5 1 215779412229418562091680268288000000 5 3 3319766398771200000 5 2 215779412229418562091680268288000000

======