CS 101 Fall 2020 - Quiz 1-B

17/9/2020 - 15 Minutes **Name:** **ID number:**

Problem 1 (5pts) Notes of discussion I promise that I will complete this QUIZ independently, and will not use mobile phones, computers and other electronic products during the QUIZ, nor will I communicate with other students during this QUIZ.

True or False: I have read the notes and understood them.

1
T

Problem 2 (5pts) XOR Linked-List

Each node of the ordinary doubly linked list has three elements, which store the address of the **previous** node, the address of the **next** node, and the **value**. If we combine the previous address and the next address into one **neighbor** address through **XOR** operation, we can greatly reduce the storage space cost. Such a data structure is called an XOR Linked-List.

Each node in the XOR Linked-List contains two elements: `node.value` and `node.neighbor`, in addition, `node.neighbor = node.next XOR node.prev`. There is an example of bitwise XOR:

`00110000 XOR 01010000 = 01100000`.

As shown in the figure below, there is an XOR linked-list with three nodes.

$$\cdots <--> \texttt{A} <--> \texttt{B} <--> \texttt{C} <--> \cdots$$

The following three sub-questions are independent of each other. You can answer this question in any way you like, including but not limited to: pseudocode, Python/C code, natural language, etc. Please make sure your description is clear and easy to understand.

In each question, you know the addresses of B and C, and you can access the `neighbor` and `value` elements through this address. Please complete the following three sub-questions.

(1) Knowing the address of node B and the address of node C, how to get the address of A?

```
A = C XOR B->neighbor
```

(2) Knowing the address of node B, the address of node C and the address of a new node D, how to insert node D between B and C?

```
D->neighbor = B XOR C
B->neighbor = B->neighbor XOR C XOR D
C->neighbor = C->neighbor XOR B XOR D
```

(3) Knowing the address of node B and the address of node C, how to delete node B from the XOR Linked-List?

```
A = C XOR B->neighbor
A->neighbor = A->neighbor XOR B XOR C
C->neighbor = C->neighbor XOR B XOR A
delete(B)
```

Hint: If you get puzzled, think about the properties of the XOR operation, such as: `X XOR Y XOR X = Y`.

CS 101 Fall 2020 - Quiz 1-B

17/9/2020 - 15 Minutes **Name:** **ID number:**

Problem 3(5pts) Algorithm Design

(1) Try to convert the polynomial below into the array form which is talked in the class. Note the exponents should be descending.

$$114x^{514} + 19x^{19} + 81x$$

index	0	1	2
coefficient	114	19	81
exponent	514	19	1

(2) Try to do addition on the two polynomial A and B below and store the result in C. Each polynomial is stored in the struct PLY.

```
struct PLY {
    int exponent[VERY_LARGE];
    int coefficient[VERY_LARGE];
    int len;
};

PLY add(PLY &A, PLY &B) {
    PLY C;
    int i = 0;
    int j = 0;
    int k = 0;
    while (i < A.len or j < B.len) {
        if (j >= B.len or i < A.len and A.exponent[i] > B.exponent[j]) {
            C.exponent[k] = A.exponent[i];
            C.coefficient[k] = A.coefficient[i];
            k++;
            i++;
        } else if (i >= A.len or j < B.len and A.exponent[i] < B.exponent[j]) {
            C.exponent[k] = B.exponent[j];
            C.coefficient[k] = B.coefficient[j];
            k++;
            j++;
        } else if (A.exponent[i] == B.exponent[j]) {
            C.exponent[k] = A.exponent[i];
            C.coefficient[k] = A.coefficient[i] + B.coefficient[j];
            k++;
            i++;
            j++;
        }
    }
    C.len = k;
    return C;
}
```