Programming Exercise 3: โครงสร้างข้อมูล: ต้นไม้

(โอลิมปิกวิชาการ ค่ายสอง ม. ศิลปากร วันที่สองของโครงสร้างต้นไม้, โดย ดร.รัชดาพร คณาวงษ์)

Problem 1: ต้นไม้นิพจน์ (Expression Tree)

นิพจน์ทางคณิตศาสตร์จะประกอบด้วยตัวดำเนินการ (operator) และตัวถูกดำเนินการ (operand) โดยรูปแบบการเขียนนิพจน์สามารถเขียนได้ 3 แบบคือ

1. infix คือการเขียนที่คุ้นเคยที่สุด โดยให้ตัวดำเนินการอยู่ระหว่างตัวถูกดำเนินการ เช่น 2 + 3 เป็นต้น
2. prefix คือการเขียนที่เครื่องเข้าใจ โดยจะเขียนตัวดำเนินการอยู่หน้าตัวถูกดำเนินการ เช่น + 2 3 เป็นต้น
3. postfix คือการเขียนที่ให้ตัวถูกดำเนินการเขียนเรียงก่อนแล้วจึงให้ตัวดำเนินการปิดท้าย เช่น 2 3 – เป็นต้น

Task

ให้รับค่าข้อความนิพจน์ โดยที่มีเครื่องหมายดำเนินการเพียง ‘+’ ‘-’ ‘*’ และ ‘/’ เท่านั้น และตัวถูกดำเนินการเป็นเลขโดด ข้อความรับเข้าจะเป็นข้อความที่เขียนตัวดำเนินการและตัวถูกดำเนินการเรียงกันไม่เว้นวรรค ใส่เป็นรูปแบบ prefix เช่น “*+2-745” เป็นต้น

(1) ให้นำนิพจน์รูปแบบ prefix แปลงเป็น infix

(2) ให้นำนิพจน์รูปแบบ prefix แปลงเป็น postfix

(3) ทำการหาค่าผลลัพธ์ที่ได้จากนิพจน์

ตัวอย่างข้อมูลเข้าและผลลัพธ์

Input	Output
Enter equation in Prefix form: +23	Prefix : +23 Infix : 2+3 Postfix : 23+ Evaluated Result : 5
Enter equation in Prefix form: -+89*23	Prefix : -+89*23 Infix : 8+9-2*3 Postfix : 89+23*- Evaluated Result : 11

ข้อเสนอแนะ

สร้างคลาสโหนด และคลาสโหนดสแตกดังนี้

```
/** class TreeNode **/
class TreeNode
{
    public:
        char data;
        TreeNode *left, *right;
        /** constructor **/
        TreeNode(char data)
        {
            this->data = data;
            this->left = NULL;
            this->right = NULL;
        }
};

/** class StackNode **/
class StackNode
{
    public:
        TreeNode *treeNode;
        StackNode *next;
        /** constructor **/
        StackNode(TreeNode *treeNode)
        {
            this->treeNode = treeNode;
```

```
            next = NULL;
        }
};
```

ในการหาค่าผลลัพธ์จะต้องใช้สแตกในการ push และ pop ต้นไม้ย่อย