

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

เครื่องหมาย	ตัวอย่างนิพจน์	ความหมาย	ผลลัพธ์
+	$5+2$	5 บวก 2	7
-	$5-2$	5 ลบ 2	3
*	$5*2$	5 คูณ 2	10
/	$5/2$	5 หารด้วย 2	2
^	5^3	5 ยกกำลัง 3	125

ข้อความรับเข้าจะเป็นข้อความที่เขียนตัวดำเนินการและตัวถูกดำเนินการเรียงกันไม่เว้นวรรค ใส่เป็นรูปแบบ prefix เช่น $*+2-745$ เป็นต้น

- (1) ให้นำนิพจน์รูปแบบ prefix แปลงเป็น infix
- (2) ให้นำนิพจน์รูปแบบ prefix แปลงเป็น postfix
- (3) ทำการหาค่าผลลัพธ์ที่ได้จากนิพจน์

ตัวอย่างข้อมูลเข้าและผลลัพธ์

Input	Output
+23	2+3 23+ 5
+-89*23	8+9-2*3 89+23*- 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