โจทย์ปัญหาแบบฝึกหัด Stack-Queue

โจทย์ปัญหา : สแตก 1 (แปลงเลขฐาน) : base

จงเขียนโปรแกรมเพื่อทำการแปลงค่าจากเลขฐานหนึ่งไปเป็นอีกเลขฐานหนึ่ง โดยโปรแกรมจะรับค่าข้อมูลเข้าที่อยู่ในรูปแบบดังนี้

n s d

โดยที่ n คือ เลขจำนวนเต็มบวกที่อยู่ในฐาน s และโปรแกรมจะทำการแปลงและแสดงค่าดังกล่าวเป็นตัวเลขที่อยู่ในฐาน d ซึ่ง s และ d จะมีค่าอยู่ในช่วง 2-16 (สำหรับเลขฐานที่มากกว่าฐานสิบ จะใช้สัญลักษณ์ a-f แทนตัวเลขของค่า 10-15 ตามลำดับ

โปรแกรมจะวนการทำงานข้างต้นไปเรื่อยๆ จนกว่าจะป้อนค่าข้อมูลเข้าเป็น 0 0 0

ตัวอย่าง

```
254 10 16
fe
a4 16 2
10100100
14 10 8
16
101111110000011 2 8
57603
7105 8 16
e45
0 0 0
```

โจทย์ปัญหา : สแตก 2 (ตรวจสอบเครื่องหมายสมดุล):bsymbol

จงเขียนโปรแกรมเพื่อทำการตรวจสอบข้อความซึ่งประกอบด้วยเครื่องหมายสัญลักษณ์เปิดปิด, ตัวอักษรภาษาอังกฤษ, ตัวเลข และอักขระพิเศษต่างๆ (เช่น % $; # & ! @ = + - * / เป็นต้น) โดยให้โปรแกรมทำการตรวจสอบว่าเครื่องหมายสัญลักษณ์เปิดปิดที่อยู่ภายในข้อความทั้งหมดนั้นมีความสมดุลกันหรือไม่ โดยเครื่องหมายสัญลักษณ์เปิดปิดที่กำหนดให้สามารถใช้งานได้คู่กัน คือ

( ) { } [ ] " " ' ' < >

ตัวอย่าง

```
(ab)[x{y}'AB']<pk>
Balance!
16P2"sb(h[x)]"
Not Balance!
a'ip59"(io)
Not Balance!
printf("%5d : %s",data,"Happy");
Balance!
```

รหัสเทียมสำหรับการตรวจสอบ Balancing Symbol

```
สร้าง Stack (ว่าง)
while (ยังมีข้อมูลในข้อความเหลืออยู่) {
        อ่านข้อมูลในข้อความลำดับถัดไปมา 1 ค่า
        if (ข้อมูลนั้นเป็นสัญลักษณ์เปิด)
                นำข้อมูลนั้นใส่ลงใน Stack
        otherwise if (ข้อมูลนั้นเป็นสัญลักษณ์ปิด) {
                        if (Stack ว่าง) {
                                (ยุติ error) สรุปได้ว่าข้อความนี้มีเครื่องหมายที่ไม่สมดุล
                        otherwise {
                                นำสัญลักษณ์เปิดออกจาก Stack
                                if (สัญลักษณ์เปิดที่นำออกมาจาก Stack ไม่ใช่คู่กับสัญลักษณ์ปิดที่อ่านเข้ามา)
                                        (ยุติ error) สรุปได้ว่าข้อความนี้มีเครื่องหมายที่ไม่สมดุล
                         }
        }
}
if (Stack ว่าง)
        สรุปได้ว่าข้อความนี้มีเครื่องหมายที่สมดุล
otherwise
        (ยุติ error) สรุปได้ว่าข้อความนี้มีเครื่องหมายที่ไม่สมดุล
```

โจทย์ปัญหา : สแตก 3 (แปลง Infix -> Postfix):postfix

จงเขียนโปรแกรมเพื่อรับค่าข้อความที่เป็น Infix Expression (จะมีวรรคคั่นหรือไม่ก็ได้) จากนั้นทำการแปลงให้เป็น Postfix Expression โดยผลลัพธ์ให้คั่นข้อมูลแต่ละตัวด้วยวรรค

ตัวอย่าง

Infix Expression	Postfix Expression
a + b	a b +
a + b * c	a b c * +
a * b + c	a b * c +
(a + b) * c	a b + c *
(a – b) * (c + d)	a b – c d + *
(a + b) * (c – d / e) + f	a b + c d e / - * f +
25+30/5–4*(22–19)	25 30 5 / + 4 22 19 - * -

รหัสเทียมสำหรับการแปลง Infix เป็น Postfix

```
สร้าง Stack (ว่าง)
while (ยังมีข้อมูลในข้อความ Infix เหลืออยู่) {
        อ่านข้อมูลในข้อความ Infix ลำดับถัดไปมา 1 ค่า
        if (ข้อมูลนั้นเป็น Operand)
                แสดงค่าข้อมูลนั้นออกไป
        if (ข้อมูลนั้นเป็นเครื่องหมายสัญลักษณ์เปิด)
                นำข้อมูลที่เป็นเครื่องหมายสัญลักษณ์เปิดนั้นใส่ลงใน Stack
        if (ข้อมูลนั้นเป็นเครื่องหมายสัญลักษณ์ปิด) {
                while (Stack ไม่ว่าง และ
                        ข้อมูลใน Stack ที่อยู่ข้างบนสุดไม่ใช่เครื่องหมายสัญลักษณ์เปิด)
                        นำข้อมูลใน Stack ออกมาและแสดงค่าข้อมูลนั้นออกไป
                นำข้อมูลใน Stack ที่เป็นเครื่องหมายสัญลักษณ์เปิดออกมาและละทิ้งไป
        }
        if (ข้อมูลนั้นเป็น Operator) {
                if (empty ว่าง หรือ ข้อมูลใน Stack ที่อยู่ข้างบนสุดเป็นเครื่องหมายสัญลักษณ์เปิด)
                        นำข้อมูลที่เป็น operator นั้นใส่ลงใน Stack
                otherwise {
                        while (Stack ไม่ว่าง และ
                                ข้อมูลใน Stack ที่อยู่ข้างบนสุดไม่ใช่เครื่องหมายสัญลักษณ์เปิด และ
                                Operator ที่อ่านเข้ามานั้นมีความสำคัญน้อยกว่าหรือเท่ากับ Operator ที่อยู่ด้านบนของ Stack)
                                   นำ Operator ออกมาจาก Stack และแสดงค่าออกไป
                        นำ Operator ตัวล่าสุดใส่ลงใน Stack
                }
        }
}
while (Stack ไม่ว่าง)
        นำ Operator ออกมาจาก Stack และแสดงค่าออกไป
```

โจทย์ปัญหา : สแตก 4 (เครื่องคิดเลข): pfixresult

จงเขียนโปรแกรมเพื่อรับค่าข้อความที่เป็น Infix Expression ที่มี operand เป็นค่าตัวเลข โดยในขั้นแรกให้ทำการแปลง Infix Expression นี้เป็น Postfix Expression จากนั้นให้ทำการประมวลผล Postfix Expression ดังกล่าวเพื่อแสดงคำตอบเป็นผลลัพธ์

ตัวอย่าง

Infix Expression	Result
25+30/5–4*(22–19)	19
((3+7)/5–7)*3	-15
30-5+50/2–40/(9-5)	40

รหัสเทียมสำหรับการประมวลผล Postfix Expression

```
สร้าง Stack (ว่าง)
while (ยังมีข้อมูลในข้อความ Postfix เหลืออยู่) {
        อ่านข้อมูลในข้อความ Postfix ลำดับถัดไปมา 1 ค่า
        if (ข้อมูลเป็น Value)
                นำค่า Value ดังกล่าวใส่ลงใน Stack
        if (ข้อมูลเป็น Operator) {
                นำค่า Value ออกมาจาก Stack จำนวนตามเท่าที่ Operator ต้องการ
                if (ไม่เกิดปัญหาข้อมูลในการนำค่าออกมาจาก Stack ข้างต้น) {
                        ทำการประมวลผล Operator กับ Operand ซึ่งเป็นค่า Value ที่ได้นำออกมาจาก Stack ข้างต้น
                        นำผลลัพธ์ของการประมวลผลคำสั่งที่คำนวณได้ใส่ลงใน Stack
                } otherwise
                        (ยุติ error) ไม่สามารถประมวลผลได้สำเร็จ
        }
}
if (มีข้อมูลใน Stack เหลือแค่เพียงหนึ่งค่าเท่านั้น)
        ค่าที่อยู่ใน Stack นั้นเป็นผลลัพธ์ที่ได้จากการประมวลผล
otherwise
        (ยุติ error) ไม่สามารถประมวลผลได้สำเร็จ
```

Example

- **Infix:** 5 + ((1 + 2) * 4) – 3
- **Postfix:** 5 1 2 + 4 * + 3 –

Input	Operation	Stack	Comment
5	Push value	5	
1	Push value	1 5	
2	Push value	2 1 5	
+	Add	3 5	Pop two values (1, 2) and push result (3)
4	Push value	4 3 5	
*	Multiply	12 5	Pop two values (3, 4) and push result (12)
+	Add	17	Pop two values (5, 12) and push result (17)
3	Push value	3 17	
–	Subtract	14	Pop two values (17, 3) and push result (14)
	Result	**14**	

โจทย์ปัญหา : คิว 1 (ซ้ำหรือผสมให้ครบสิบ): deque

จงเขียนโปรแกรมโดยใช้งานโครงสร้างข้อมูลประเภท Deque เพื่อใช้สำหรับจัดเก็บจำนวนเต็ม โดยโปรแกรมจะวนทำการสุ่มตัวเลขจำนวนเต็มที่อยู่ในช่วง 0 ถึง 99 มาทีละ 1 จำนวน จากนั้นให้พิจารณาค่าที่รับเข้ามาใหม่กับค่าของข้อมูลตัวที่อยู่ต้นคิว กรณีทั้งสองค่ามีเลขหลักหน่วยตรงกัน หรือ เลขหลักหน่วยของทั้งสองค่ารวมกันแล้วเท่ากับ 10 หรือ เลขหลักหน่วยของทั้งสองมีค่าต่างกันไม่เกิน 1 ให้ทำการนำข้อมูลตัวที่อยู่ต้นคิวนั้นออกจากคิว มิฉะนั้นให้ทำการใส่ข้อมูลที่รับเข้ามาใหม่นั้นลงไปต่อท้ายคิว ในแต่ละรอบการของวนซ้ำนั้นให้แสดงค่าตัวเลขที่สุ่มได้และข้อมูลที่อยู่ในคิวทั้งหมดด้วย โปรแกรมจะยุติการทำงานก็ต่อเมื่อคิวไม่มีข้อมูลเหลืออยู่เลย หรือจนกว่าจะคิวจะมีข้อมูลจำนวนเกินกว่า 5 ตัว โดยให้โปรแกรมแสดงการทำงานตามตัวอย่าง

ตัวอย่าง

```
New 73 pushed.
Q(1): 73
New 52 kicks 73!
Q(0):
```

```
New 6 pushed.
Q(1): 6
New 84 kicks 6!
Q(0):
```

```
New 30 pushed.
Q(1): 30
New 87 pushed.
Q(2): 30 87
New 32 pushed.
Q(3): 30 87 32
New 27 pushed.
Q(4): 30 87 32 27
New 99 pushed.
Q(5): 30 87 32 27 99
New 43 pushed.
Q(6): 30 87 32 27 99 43
```

```
New 28 pushed.
Q(1): 28
New 60 pushed.
Q(2): 28 60
New 69 kicks 28!
Q(1): 60
New 31 kicks 60!
Q(0):
```

```
New 29 pushed.
Q(1): 29
New 80 pushed.
Q(2): 29 80
New 2 pushed.
Q(3): 29 80 2
New 30 pushed.
Q(4): 29 80 2 30
New 96 pushed.
Q(5): 29 80 2 30 96
New 98 kicks 29!
Q(4): 80 2 30 96
New 37 pushed.
Q(5): 80 2 30 96 37
New 4 pushed.
Q(6): 80 2 30 96 37 4
```