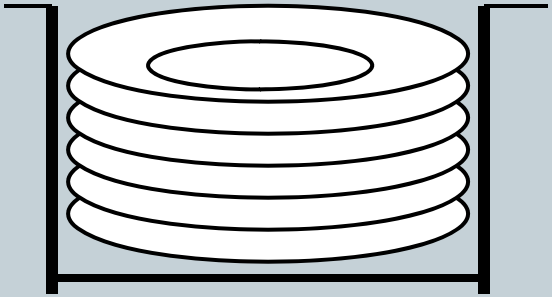
โครงสร้างข้อมูลแบบ

1

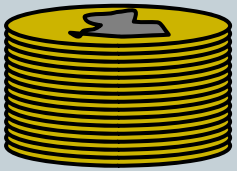
STACK

สแตก (Stack)

- สแตก เป็นโครงสร้างข้อมูลที่ต่อเนื่อง
- สแตก มีการเพิ่มหรือลบข้อมูลออกได้ทางเดียว
- รายการที่เข้าหลังสุดจะเป็นรายการที่ถูกออกก่อน
- Last in first out : LIFO
- สแตกเป็นได้ทั้งลิสต์และอาร์เรย์



สแตกของจาน

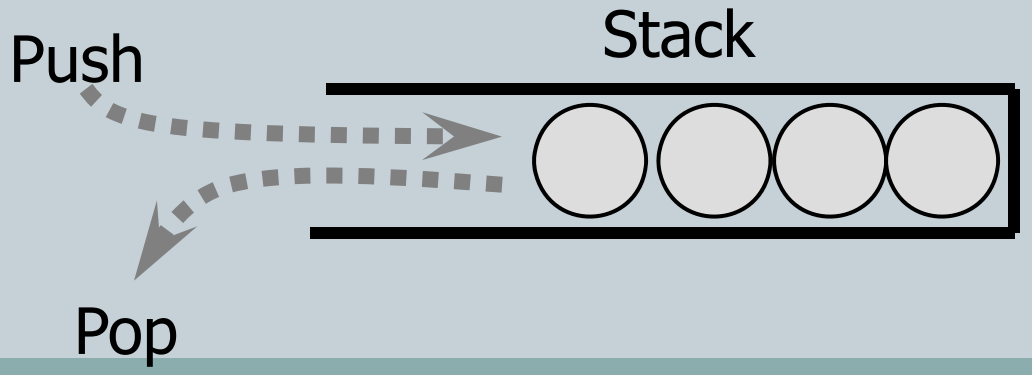


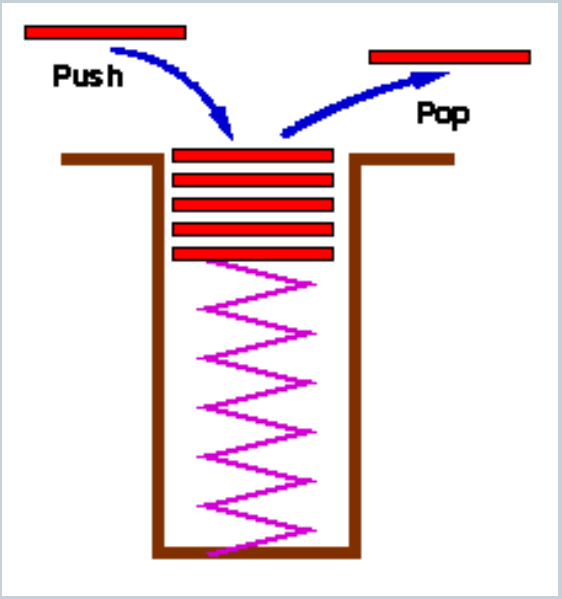
สแตกของเหรียญ

สแตกของหนังสือ

สแตก (Stack)

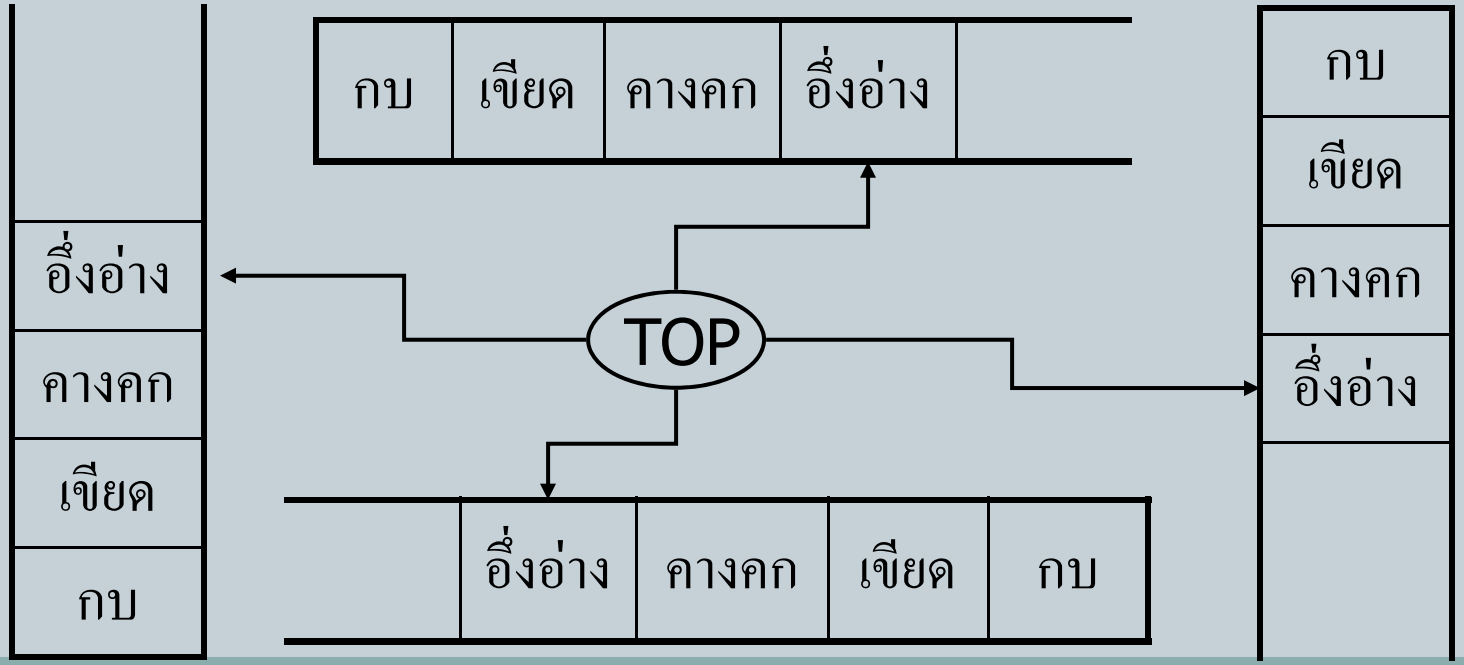
- สแตก มีการจัดการเพิ่มหรือลบข้อมูลจากด้านบน (TOP) ของสแตก
- ข้อมูลที่นำออกจากสแตกจะเรียงตามลำดับแบบกับลำดับกับการเพิ่มเข้า
- คำที่ใช้กับสแตกมี 2 คำ คือ
  - Push ใช้สำหรับการเพิ่มข้อมูลลงในสแตก
  - Pop ใช้สำหรับการดึงข้อมูลออกจากสแตก(ลบ)





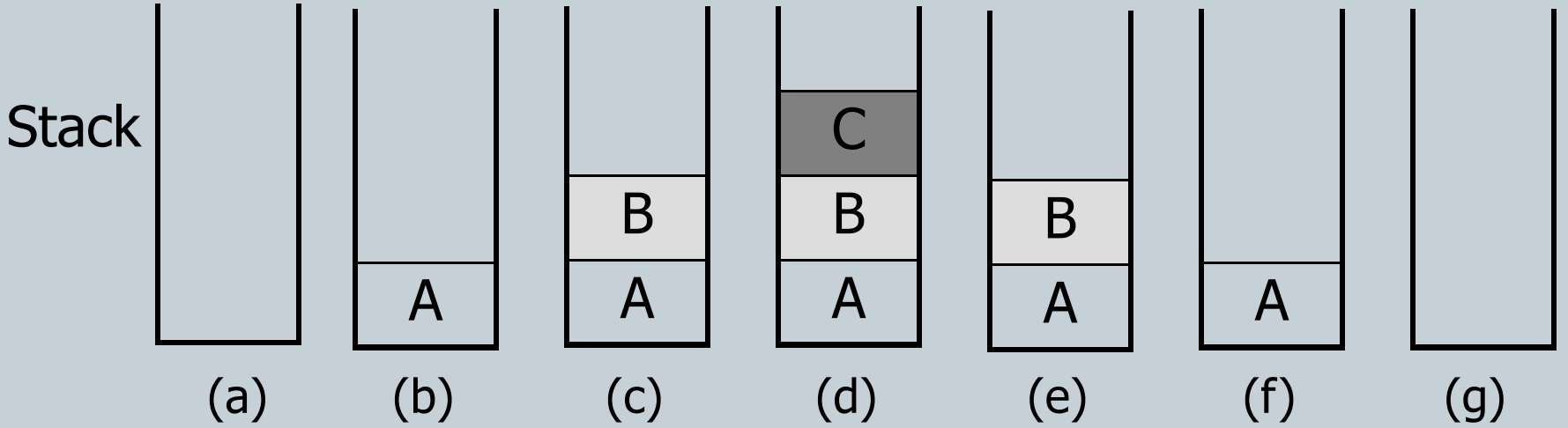
สแตก (Stack)

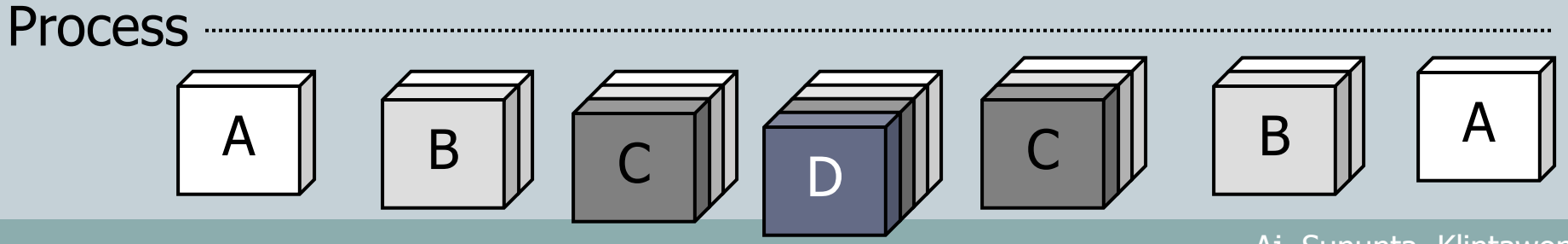
- ตัวอย่างการเก็บข้อมูลลงในสแตก ทั้ง 4 แบบ
  เมื่อมีข้อมูล 4 รายการต่อไปนี้ถูกเก็บลงในสแตกว่าง ตามลำดับ
  (กบ,เขียด,คางคก,อึ่งอ่าง)



การทำงานของสแตก

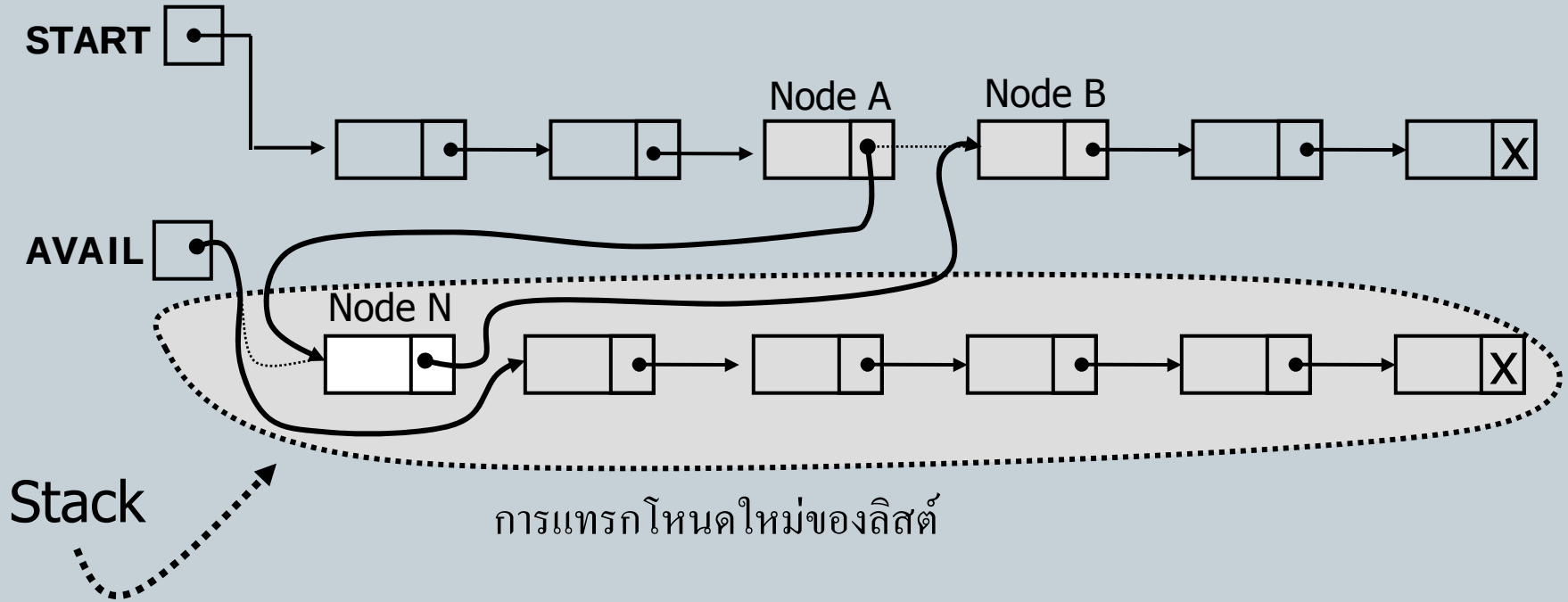
- แสตกมีการใช้สำหรับแสดงลำดับการประมวลผลข้อมูล เมื่อต้องการข้ามขั้นตอนบางขั้นตอนไปกระทำขั้นตอนอื่นให้จบก่อนแล้วจึงกลับมาทำขั้นตอนเดิมต่อ





การทำงานของสแตก

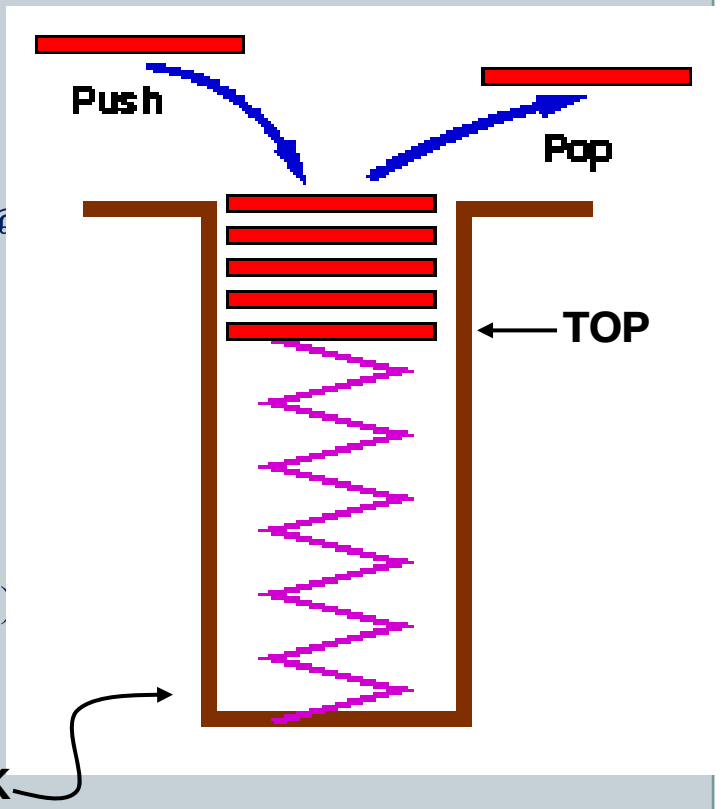
- การทำงานของลิสต์ AVAIL โหนดที่ถูกดึงออก และโหนดว่างตัวใหม่จะแทรกลงในตำแหน่งเริ่มต้นของ AVAIL เสมอ ซึ่งเป็นการทำงานแบบ สแตก



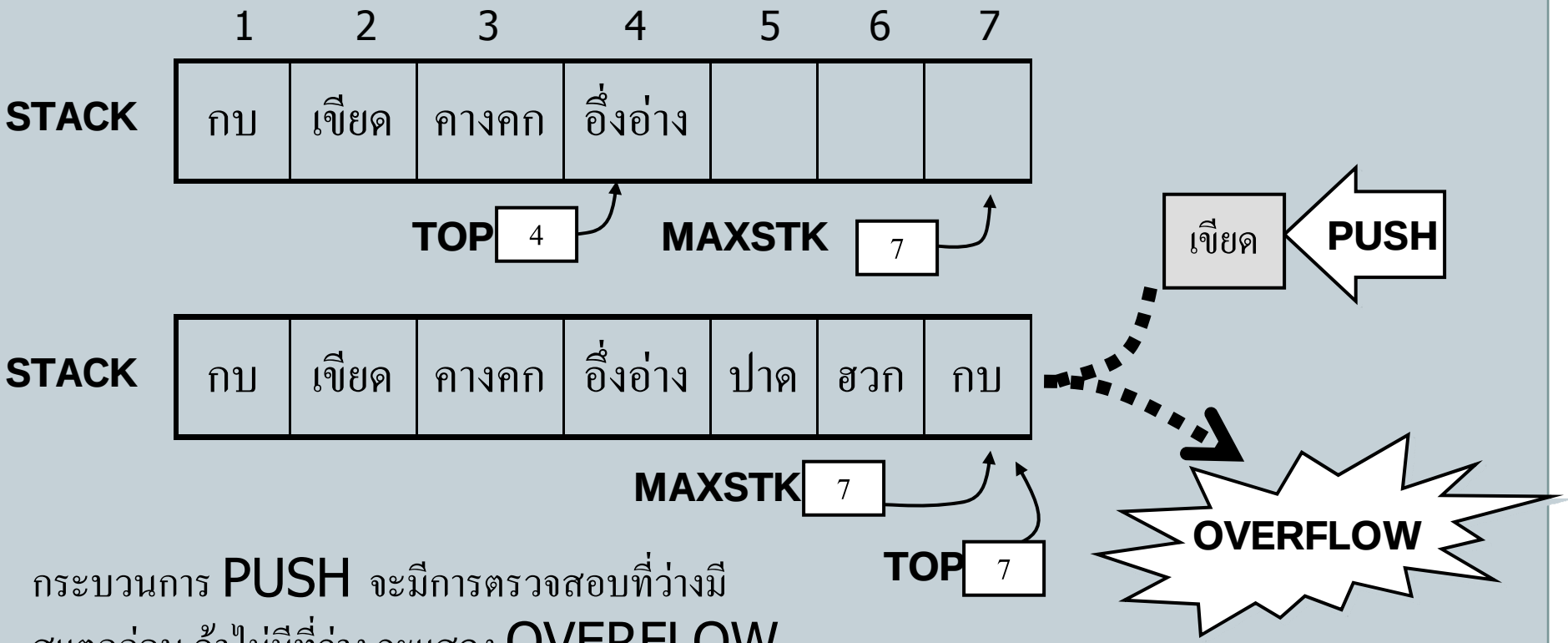
การแทรกโหนดใหม่ของลิสต์

สแตกในรูปอาร์เรย์

- สแตกสามารถเก็บได้ทั้ง แบบลิสต์ทางเดียว หรือ อาร์เรย์ ก็ได้
- รู้จักคำที่เกี่ยวข้องกับสแตก
  - PUSH กระบวนการใช้เพิ่มข้อมูลลงในสแตก
  - POP กระบวนการใช้ดึงข้อมูลออกจากสแตก(
  - TOP ตัวแปรเก็บตำแหน่งบนสุดของสแตก
  - MAXSTK ตัวแปรเก็บค่าสูงสุดที่จะเก็บลงในสแตก
  - OVERFLOW ไม่มีที่ว่างในสแตกสำหรับเพิ่มข้อมูล (สแตกเต็ม)
  - UNDERFLOW ไม่มีข้อมูลในแสตกเลย (สแตกว่าง)
  - ITEM เป็นข้อมูลที่จำเข้า-ออกจากสแตก

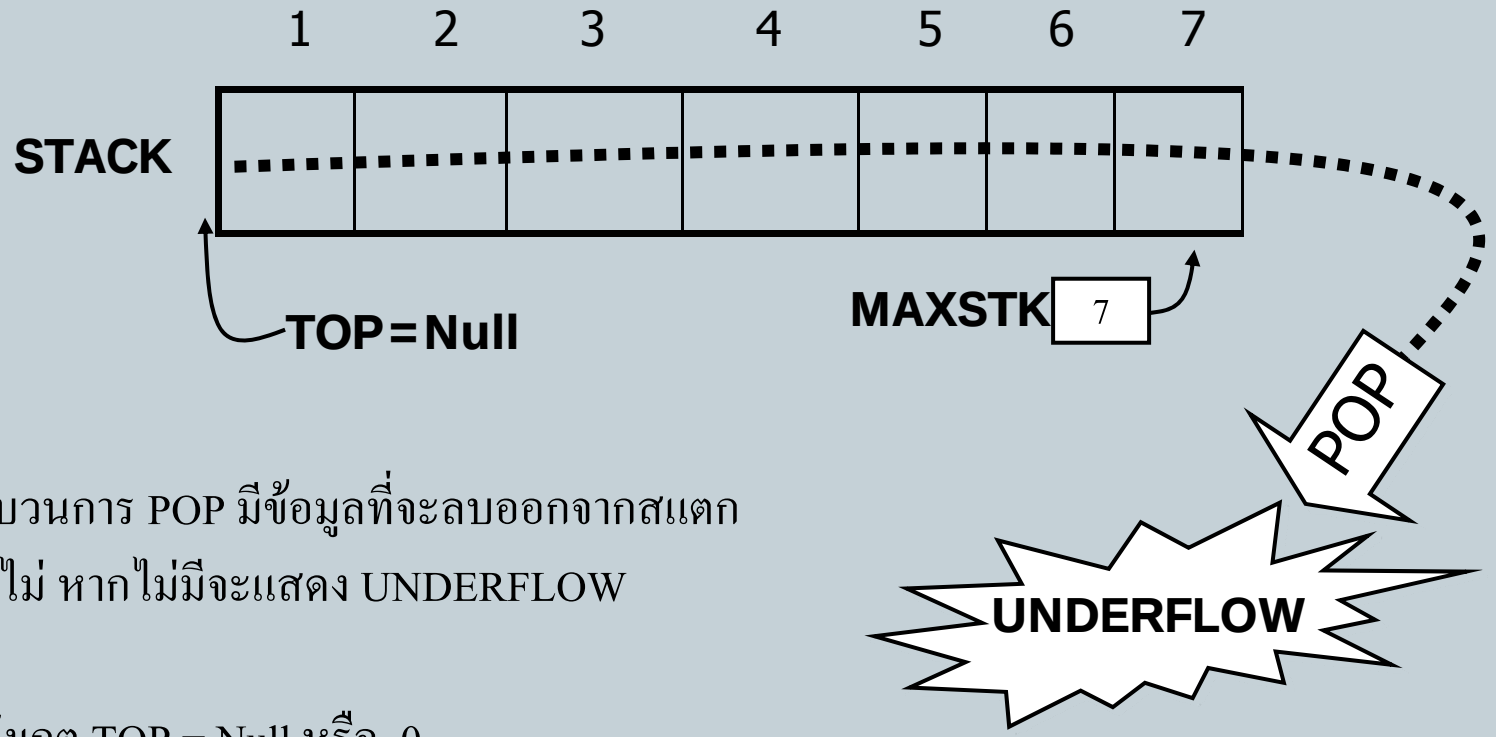


สแตกในรูปอาร์เรย์



กระบวนการ PUSH จะมีการตรวจสอบที่ว่างมีสแตกก่อน ถ้าไม่มีที่ว่าง จะแสดง OVERFLOW ข้อสังเกต (TOP = MAXSTK)

สแตกในรูปอาร์เรย์



กระบวนการ POP มีข้อมูลที่จะลบออกจากสแตก
หรือไม่ หากไม่มีจะแสดง UNDERFLOW

ข้อสังเกต TOP = Null หรือ 0

PUSH STACK

Procedure 5.1 **PUSH**(STACK,TOP,MAXSTK,ITEM)

1. [ตรวจสอบสแตกเต็มหรือไม่?]

   If TOP = MAXSTK then :

   Print: OVERFLOW, and Return.

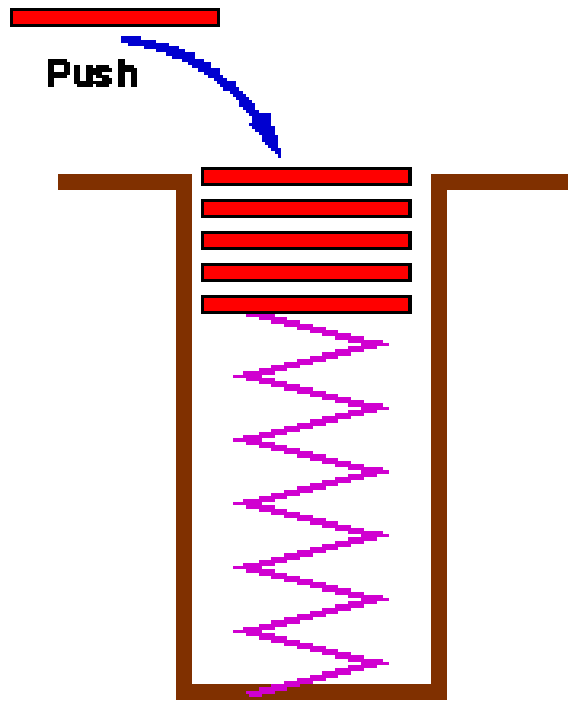
2. Set TOP := TOP+1.

   [เพิ่มตำแหน่งให้ TOP ไป 1 ตำแหน่ง]

3. Set STACK[TOP] := ITEM.

   [แทรกข้อมูลใหม่ลงใน TOP ตำแหน่งใหม่]

4. Return



POP STACK

Procedure 5.2 **POP**(STACK,TOP,ITEM)

1. [สแตกมีข้อมูลที่จะให้เอาออกหรือไม่?]

   If TOP = Null then :

   Print: UNVERFLOW, and Return.

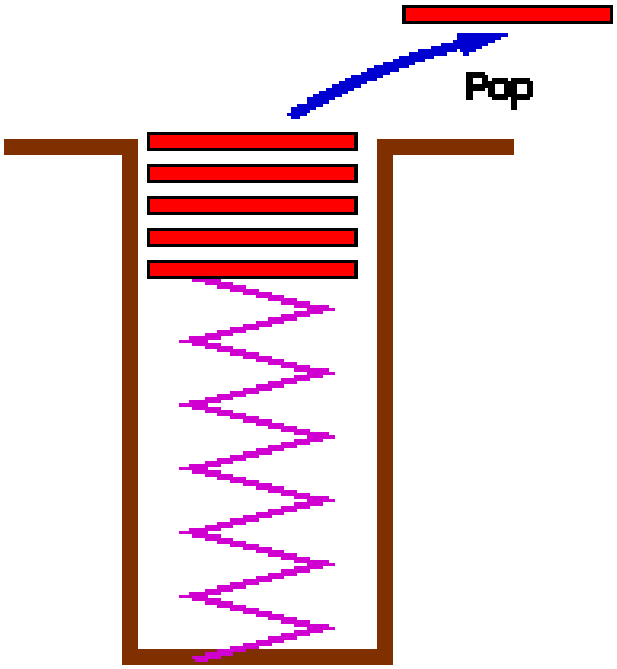
2. Set ITEM := STACK[TOP].

[ถ่ายข้อมูลจากตำแหน่ง TOP ให้ ITEM เก็บข้อมูลไว้]

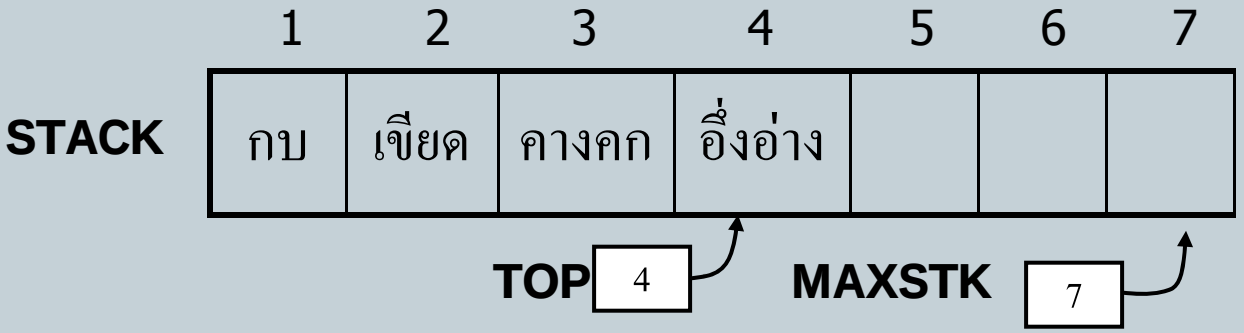
3. Set TOP := TOP-1.

   [ลดตำแหน่งให้ TOP ลง 1 ตำแหน่ง]

4. Return



PUSH STACK

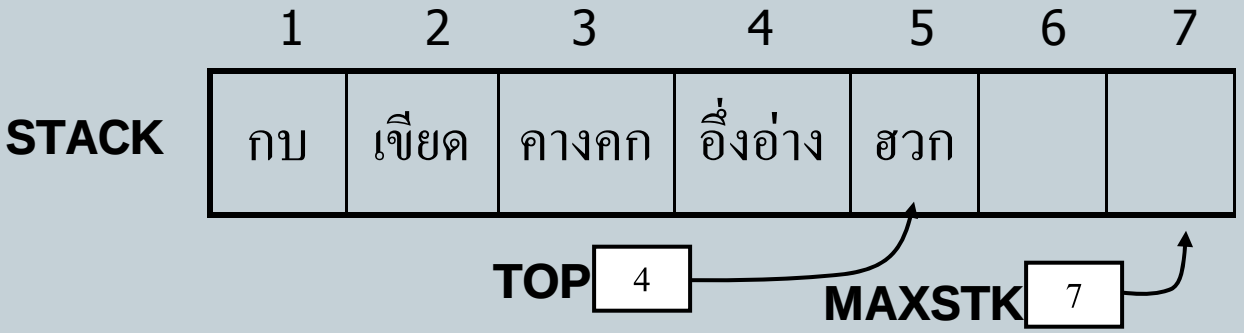


จากรูป จะแสดงการทำงานของ PUSH เมื่อต้องการเพิ่ม “ฮวก” เข้าไปในสแตก (ITEM=ฮวก)

PUSH (STACK,TOP,MAXSTK,ฮวก)

1. เนื่องจาก TOP = 4 จะไปทำงานที่ Step 2
2. TOP = 4+1 = 5
3. STACK[TOP]=STACK[5]=ฮวก
4. จบการทำงาน

POP STACK



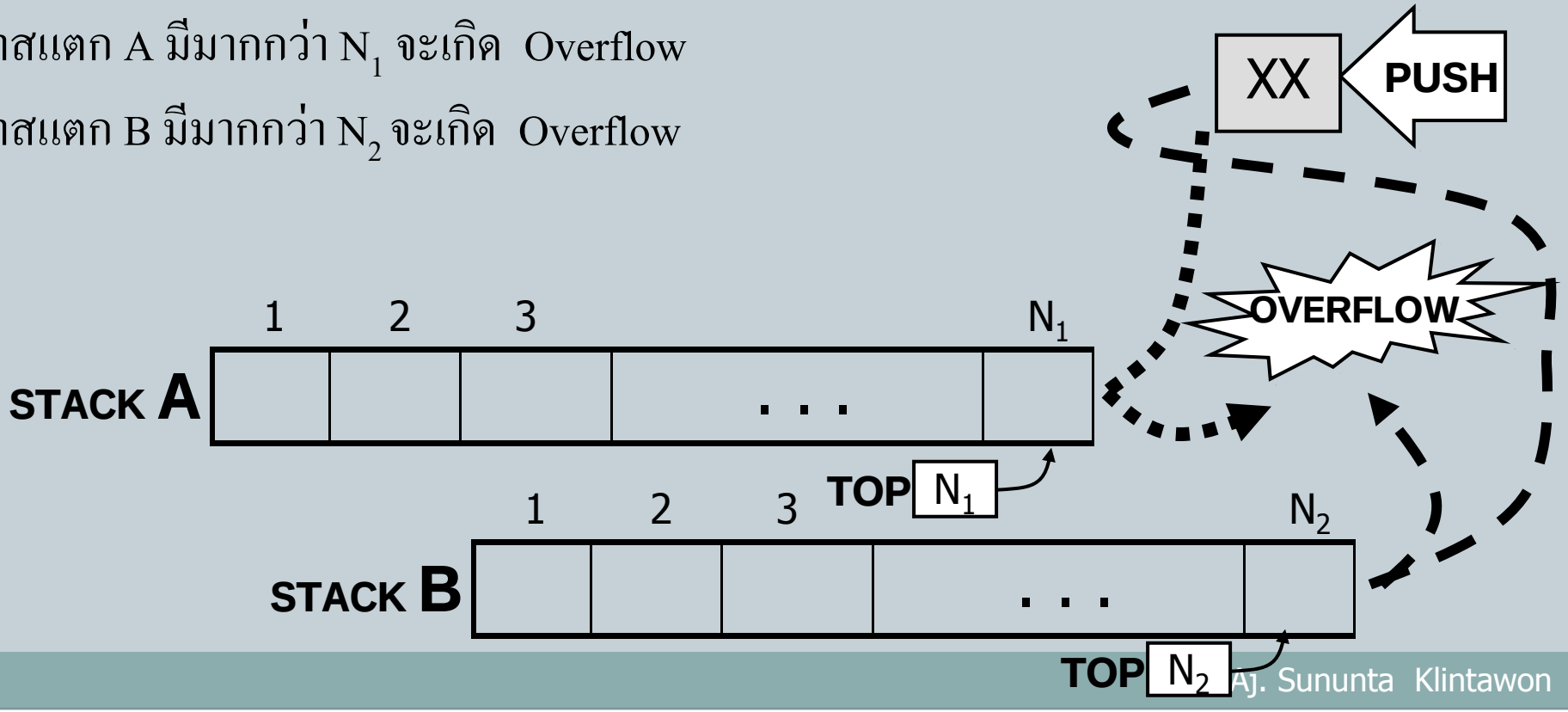
จากรูป จะแสดงการทำงานของ POP เมื่อต้องการลบข้อมูลออกจากสแตก 1ตัว

POP (STACK,TOP,ITEM)

1. เนื่องจาก TOP = ...
2. ITEM = ...
3. TOP = ...
4. จบการทำงาน

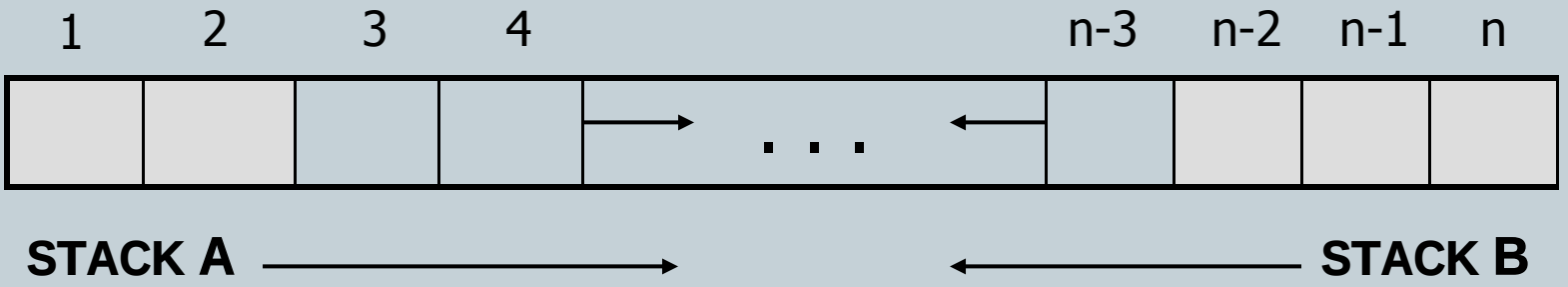
การลดปัญหา Overflow

- สมมติมี 2 สแตก คือ A และ B โดยที่
  - สแตก A มีจำนวนเก็บข้อมูลได้ N_1 ตัว
  - สแตก B มีจำนวนเก็บข้อมูลได้ N_2 ตัว
- ถ้าสแตก A มีมากกว่า N_1 จะเกิด Overflow
- ถ้าสแตก B มีมากกว่า N_2 จะเกิด Overflow



การลดปัญหา Overflow

- ถ้ากำหนดอาร์เรย์เดียว คือ สแตกมีจำนวนสมาชิกได้ $n = N_1+N_2$ สำหรับสแตก A และ B ใช้งาน
- สแตก A เริ่มจาก STACK[1] และเพิ่มไปทางขวามือ
- สแตก B เริ่มจาก STACK[n] และเพิ่มไปทางซ้ายมือ
- การเกิด Overflow เมื่อ สแตก A และ B รวมกันแล้วมีสมาชิกมากกว่า n
- สแตก A สามารถเข้าไปใช้พื้นที่ของสแตก B ได้ และเช่นเดียวกัน สแตก B ก็สามารถเข้าไปใช้พื้นที่ของสแตก A ได้ตราบใดที่ยังไม่ Overflow



Home Work

- กำหนดให้ สแตกมีจำนวนทั้งหมด 8 ช่อง และมีข้อมูลดังรูป

จงอธิบายผลของสแตกจากการกระทำตามลำดับต่อไปนี้

- PUSH(STACK,TOP,MAXSTK,NN)
- POP(STACK,TOP,ITEM)
- POP(STACK,TOP,ITEM)
- PUSH(STACK,TOP,MAXSTK,SS)
- POP(STACK,TOP,ITEM)
- POP(STACK,TOP,ITEM)
- POP(STACK,TOP,ITEM)

เขียนเป็นแผนภาพ
พร้อมอธิบายผลที่เกิดขึ้น

การแปลงนิพจน์

INFIX เป็น POSTFIX

รูปแบบนิพจน์คณิตศาสตร์

- นิพจน์ Infix ➔ คือนิพจน์ที่อยู่ในรูปแบบเครื่องหมายดำเนินการ (operator) อยู่ระหว่างตัวถูกดำเนินการ (operands) เช่น A+B
- นิพจน์ Prefix ➔ คือนิพจน์ที่อยู่ในรูปแบบเครื่องหมายดำเนินการ อยู่หน้าตัวถูกดำเนินการ เช่น +AB
- นิพจน์ postfix ➔ คือนิพจน์ที่อยู่ในรูปแบบเครื่องหมายดำเนินการอยู่หลังตัวถูกดำเนินการ เช่น +AB

ลำดับการทำงานของตัวดำเนินการทางคณิตศาสตร์

1. เครื่องหมายวงเล็บ (ไม่ใช่ตัวดำเนินการ)
2. ตัวดำเนินการยกกำลัง (^)
3. ตัวดำเนินการคูณ (*) และหาร (/)
4. ตัวดำเนินการบวก (+) และลบ (-)

โดยตัวดำเนินการที่มีลำดับเดียวกัน ให้ทำงานจากซ้ายไปขวา ยกเว้นตัวดำเนินการยกกำลัง ให้ทำงานจากขวาไปซ้าย ตัวอย่างเช่น

A+B+C หมายถึง (A+B)+C

A^B^C หมายถึง A^(B^C)

อัลกอริทึมเพื่อแปลงนิพจน์ Infix เป็น Postfix

กำหนดให้มีส่วนที่เกี่ยวข้องกัน ได้แก่ ข้อมูลเข้า เป็นนิพจน์ Infix ข้อมูลออกหรือผลลัพธ์ เป็นนิพจน์ postfix และมีสแตกเพื่อเก็บตัวดำเนินการ หลังจากนั้นให้อ่านข้อมูลเข้าทีละตัวอักษร ซึ่งการแปลงนิพจน์ Infix เป็น Postfix มีขั้นตอนดังนี้

1. อ่านค่าอินพุตถ้าเป็น operand ให้นำไปไว้ที่เอาต์พุต
2. อ่านค่าอินพุตถ้าเป็น operator ให้ทำดังนี้

2.1 นำ operator ลงสู่สแตกถ้าสแตกว่าง (เรียกสแตกที่เก็บนี้ว่า operator stack เรียกย่อว่า opst)

2.2 ถ้า opst ไม่ว่างให้เปรียบเทียบค่าของ operator ที่อินพุตกับ operator ในสแตกโดยเทียบค่าในตาราง

- ถ้าค่าของ operator ที่เป็นอินพุตมีค่าน้อยกว่าหรือเท่ากับ ค่าของ operator ที่อยู่ส่วนบนสุดของสแตกก็ให้ pop stack ออกสู่เอาต์พุตจนกว่าค่าของ operator ที่เป็นอินพุตมีค่ามากกว่า ค่าของ operator ที่อยู่ส่วนบนสุดของสกตแหรือจนกว่าสแตกว่าง แล้ว push operator ที่เป็นอินพุต ลงในสแตก
- ถ้าค่าของ operator ที่เป็นอินพุตมีค่ามากกว่าค่าของ operator ที่อยู่ส่วนบนสุดของสแตกก็ให้ push operator ลงสแตก

3. อ่านค่าอินพุตถ้าเป็นเครื่องหมายวงเล็บเปิด ‘(’ ก็ให้ทำการ pop สแตก
4. อ่านค่าอินพุตถ้าเป็นเครื่องหมายวงเล็บเปิด ‘)’ ก็ให้ทำการ pop สแตกออกสู่เอาต์พุตจนกว่าจะ pop สแตกเจอเครื่องหมายวงเล็บเปิดใหม่แล้วทิ้งเครื่องหมายวงเล็บปิด และเปิดคู่เติมโดยไม่ต้องแสดงเครื่องหมายวงเล็บปิดและเปิดในเอาต์พุต
5. ถ้าอ่านค่าอินพุตหมดแล้วให้ตรวจในสแตกว่าว่างหรือไม่ถ้าไม่ว่างให้ pop สแตกออกมาสู่เอาต์พุตจนสแตกว่าง

ตัวอย่าง

A+B-(C*D)

Input	Output	opst
A	ว่างเปล่า	ว่างเปล่า
+	A	ว่างเปล่า
B	A	+
-	AB	+
(	AB+	-
C	AB+	-(
*	AB+C	-(
D	AB+C	-(*
)	AB+CD	-(*
ว่างเปล่า	**AB+CD*-**	ว่างเปล่า