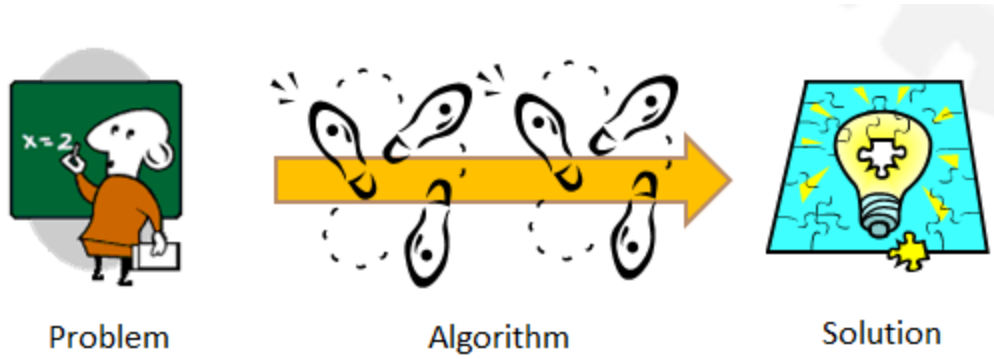
บทที่ 3 ระเบียบวิธีการพัฒนาโปรแกรม

วัตถุประสงค์

1) เพื่อให้นักศึกษาเข้าใจวิธีการออกแบบโปรแกรม
2) เพื่อให้นักศึกษาสามารถเขียนผังงานโปรแกรมเพื่อแก้ปัญหาอย่างง่ายได้

3.1 อัลกอริทึม (Algorithm)



รูปที่ 1 อัลกอริทึม

อัลกอริทึม คือ ขั้นตอนหรือกระบวนการแก้ปัญหา การเขียนโปรแกรมคอมพิวเตอร์โดยทั่วไปแล้วเป็นการเขียนโค้ดเพื่อแก้ปัญหา เราต้องการเขียนโปรแกรมให้คอมพิวเตอร์ทำงานแทนเพราะคอมพิวเตอร์สามารถทำงานจำพวกการคำนวณได้เร็วกว่ามนุษย์และมีหน่วยความจำที่มีประสิทธิภาพมากกว่า แต่คอมพิวเตอร์ยังไม่สามารถ “คิด” เองได้ ดังนั้นในการเขียนโปรแกรม เราต้องมีอัลกอริทึมหรือวิธีแก้ปัญหาก่อน เพื่อจะได้สั่งให้คอมพิวเตอร์ทำงานเพื่อแก้ปัญหาตามที่เราต้องการได้

คุณสมบัติของอัลกอริทึม มีดังนี้

1) มีข้อมูลนำเข้าหรืออินพุต (input) เป็นจำนวนเท่าใดก็ได้หรือไม่มีก็ได้
2) มีข้อมูลส่งออกหรือเอาต์พุต (output) อย่างน้อย 1 ตัว
3) มีจุดสิ้นสุดของขั้นตอน ไม่วนไปเรื่อยๆ โดยไม่มีที่สิ้นสุด
4) ขั้นตอนแต่ละขั้นตอนของอัลกอริทึมต้องชัดเจน
5) ขั้นตอนแต่ละขั้นตอนของอัลกอริทึมต้องเป็นขั้นตอนพื้นฐานและง่ายต่อการทำ

ตัวอย่างที่ 1 หาค่า **x** จากสมการ

ถ้าเราต้องการหาค่า x จากสมการ 2x + 1 = 0 เราจะมีวิธีการแก้ปัญหาเป็นขั้นตอนดังนี้

ขั้นตอนที่ 1 ลบ 1 เข้าไปทั้งสองข้าง จะได้

$$2x + 1 - 1 = 0 - 1$$

$$2x = -1$$

ขั้นตอนที่ 2 หาร 2 เข้าไปทั้งสองข้าง จะได้

$$2x / 2 = -1/2$$

$$x = -1/2$$

แล้วเราก็จะได้คำตอบว่า $x = -1/2$

จากตัวอย่างนี้ เราจะเห็นได้ว่า ปัญหาที่เราต้องการแก้คือการหาค่า x จากสมการ ส่วนอัลกอริทึมคือวิธีการแก้ปัญหา นั่นคือส่วนของขั้นตอนที่ 1 และ 2 และคำตอบคือ x = -1/2

ตัวอย่างที่ 2 ตู้กดเงิน

มีลูกค้าต้องการกดเงินจากตู้กดเงินเป็นจำนวน x บาท โดยใช้บัตรกดเงิน (บัตรเอทีเอ็ม) โดยปกติแล้ว ตู้กดเงินจะให้เงินกับลูกค้าถ้า 1) รหัสในการกดถูกต้อง และ 2) เงินในบัญชีของลูกค้าเพียงพอกับจำนวนที่ลูกค้าต้องการ ขั้นตอนการทำงานของตู้กดเงินเป็นดังนี้

ขั้นตอนที่ 1 รับบัตรเอทีเอ็มของลูกค้าเข้าไปในเครื่อง
ขั้นตอนที่ 2 รับรหัสจากลูกค้า
ขั้นตอนที่ 3 ตรวจสอบว่ารหัสที่ลูกค้ากดตรงกับรหัสที่บันทึกไว้หรือไม่
  ขั้นตอนย่อยที่ 3.1 ถ้ารหัสถูกต้อง ให้ทำขั้นตอนที่ 4 ต่อ
  ขั้นตอนย่อยที่ 3.2 ถ้ารหัสไม่ถูกต้อง ให้ข้ามไปทำขั้นตอนที่ 6
ขั้นตอนที่ 4 รับจำนวนเงินที่ลูกค้าต้องการจะกด
ขั้นตอนที่ 5 ตรวจสอบว่าจำนวนเงินคงเหลือในบัญชีของลูกค้าเพียงพอสำหรับจำนวนที่ลูกค้ากดหรือไม่
  ขั้นตอนย่อยที่ 5.1 ถ้าเพียงพอ ให้เงินกับลูกค้า
  ขั้นตอนย่อยที่ 5.1 ถ้าไม่เพียงพอ แจ้งลูกค้าว่าจำนวนเงินไม่พอ แล้วไม่ให้เงินกับลูกค้า
ขั้นตอนที่ 6 จบการทำรายการ

จากตัวอย่างนี้ เราจะเห็นได้ว่า ปัญหาที่เราต้องการแก้คือการทำงานของตู้กดเงิน ส่วนอัลกอริทึมคือการแก้ปัญหาในขั้นตอนที่ 1-6 โดยการแก้ปัญหาจะมีเงื่อนไขต่างๆ เข้ามาร่วมด้วย ซึ่งจะแตกต่างจากตัวอย่างที่ 1 ซึ่งไม่มีเงื่อนไขเข้ามาเกี่ยวข้อง เมื่อมีเงื่อนไขเข้ามาในขั้นตอน จึงจำเป็นจะต้องมีการตรวจสอบเงื่อนไข และมีขั้นตอนย่อยเพื่อทำขั้นตอนต่อไปตามผลของการตรวจสอบเงื่อนไขนั้น

ตัวอย่างที่ 3 การคำนวณค่าแฟกทอเรียล **(factorial)** ของ **x**

เมื่อกำหนดค่า x ให้และเราต้องการหาค่าแฟกทอเรียล (factorial) ของ x หรือ x!

แฟกทอเรียลของ x คือค่า x (x – 1)(x - 2)... 1 หรือคือผลคูณของ x, x-1, x-2 ไปเรื่อยๆ จนถึง 1

เช่น 5! คือค่า 5 * 4 * 3 * 2 * 1 = 120 เรามีขั้นตอนการแก้ปัญหาดังนี้

ขั้นตอนที่ 1 กำหนดค่า x หรือรับค่า x มาจากผู้ใช้
ขั้นตอนที่ 2 กำหนดค่าผลลัพธ์ปัจจุบัน (fac) เป็น 1 ก่อน
ขั้นตอนที่ 3 ทดสอบว่าค่า x มากกว่า 1 หรือไม่
ขั้นตอนที่ 3.1 ถ้า x มากกว่า 1 ให้เอาผลลัพธ์ปัจจุบัน (fac) คูณ x แล้วเก็บค่าไว้ที่ fac เหมือนเดิม
ขั้นตอนที่ 3.2 ถ้า x ไม่มากกว่า 1 ให้ไปที่ขั้นตอนที่ 6
ขั้นตอนที่ 4 ลดค่า x ลง 1 ค่า เช่นถ้า x = 5 ให้กำหนดค่า x = 4
ขั้นตอนที่ 5 ทำซ้ำขั้นตอนที่ 3
ขั้นตอนที่ 6 จบการทำรายการ

สำหรับตัวอย่างนี้ นักศึกษาอาจจะงงเล็กน้อยเพราะมีการวนทำซ้ำเกิดขึ้น ดังนั้นจะขออธิบายเพิ่มเติมโดยใช้ตัวอย่างการหา 4! ตามตารางข้างล่างนี้

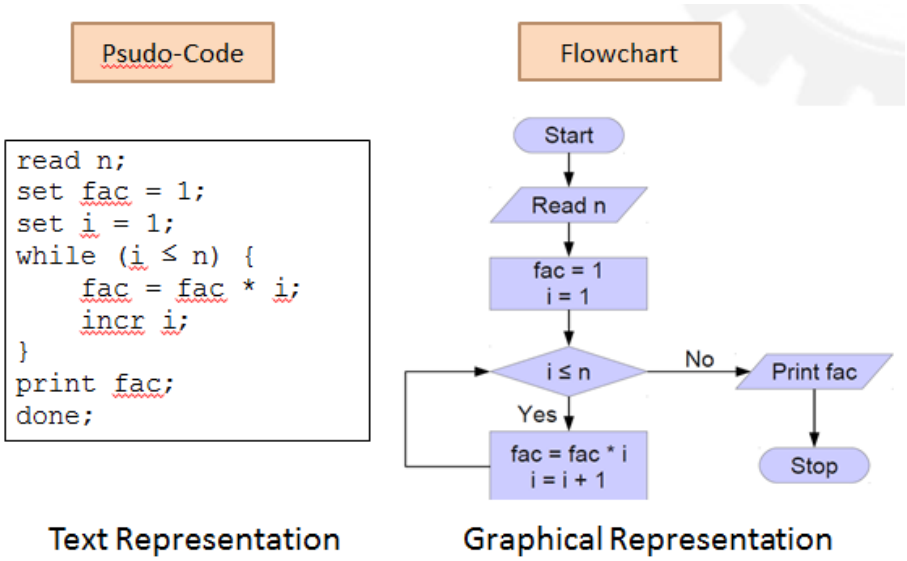
	รอบที่ 1	**รอบที่ 2**	**รอบที่ 3**	**รอบที่ 4**
ขั้นตอนที่ 1 กำหนดค่า x หรือรับค่า x มาจากผู้ใช้	x = 4			
ขั้นตอนที่ 2 กำหนดค่าผลลัพธ์ปัจจุบัน (fac) เป็น 1 ก่อน	fac = 1			
ขั้นตอนที่ 3 ทดสอบว่าค่า x มากกว่า 1 หรือไม่	x > 1?	x > 1?	x > 1?	x > 1?

	รอบที่ 1	**รอบที่ 2**	**รอบที่ 3**	**รอบที่ 4**
ขั้นตอนที่ 3.1 ถ้า x มากกว่า 1 ให้เอาผลลัพธ์ปัจจุบัน (fac) คูณ x แล้วเก็บค่าไว้ที่ fac เหมือนเดิม	fac = 1 * 4 = 4	fac = 4 * 3 = 12	fac = 12 * 2 = 24	
ขั้นตอนที่ 3.2 ถ้า x ไม่มากกว่า 1 ให้จบการทำรายการ โดยไปที่ขั้นตอนที่ 6				ไปทำขั้นตอนที่ 6
ขั้นตอนที่ 4 ลดค่า x ลง 1 ค่า	x = 4 – 1 = 3	x = 3 – 1 = 2	x = 2 – 1 = 1	
ขั้นตอนที่ 5 ทำซ้ำขั้นตอนที่ 3	ขึ้นรอบที่ 2	ขึ้นรอบที่ 3	ขึ้นรอบที่ 4	
ขั้นตอนที่ 6 จบการทำรายการ				จบการทำรายการ ได้ค่า fac = 24

เราจะเห็นได้ว่า ในตัวอย่างนี้ ปัญหาที่เราต้องการแก้คือการหาค่าแฟกทอเรียลของ x ส่วนอัลกอริทึมคือขั้นตอนการหาค่าแฟกทอเรียลของ x ซึ่งจะมีการตรวจสอบเงื่อนไขเพื่อการวนซ้ำด้วย อัลกอริทึมของการหาแฟกทอเรียลของ x จะซับซ้อนกว่าอัลกอริทึมของปัญหาในตัวอย่างที่ 1 และ ตัวอย่างที่ 2 แต่เราจะสังเกตได้ว่า ในแต่ละขั้นตอนนั้นเป็นขั้นตอนพื้นฐานตามคุณสมบัติของอัลกอริทึมที่ระบุไว้ข้างต้นเหมือนกัน

3.2 ผังงาน (Flowchart)

การเขียนเพื่อแสดงอัลกอริทึมสามารถทำได้หลายวิธี เช่นการเขียนโค้ดจำลอง (pseudo-code) หรือการเขียนผังงาน (flowchart) การเขียนโค้ดจำลองเป็นการแสดงอัลกอริทึมโดยผ่านตัวอักษรที่เป็นภาษาที่มนุษย์ทั่วไปเข้าใจได้ แต่ก็ใกล้เคียงกับการเขียนโค้ดจริงๆ ส่วนการเขียนผังงาน (flowchart) เป็นการแสดงอัลกอริทึมโดยผ่านรูปสัญลักษณ์ต่างๆ และตัวอักษร ดังแสดงในรูปที่ 2



รูปที่ 2 การแสดงอัลกอริทึมในรูปแบบโค้ดจำลองและรูปแบบผังงาน

3.2.1 ส่วนประกอบของผังงาน (flow chart components)

สัญลักษณ์ที่ใช้ในการเขียนผังงานมี 6 ชนิด ดังนี้

1) ลูกศร (arrow)

รูปที่ 3 สัญลักษณ์ลูกศร

ลูกศรมีหน้าที่ระบุลำดับการทำงานของขั้นตอนต่างๆ ในอัลกอริทึม ในการเขียนผังงาน เราจะใช้ลูกศรทางเดียวเท่านั้น โดยเราจะทำขั้นตอนจากหางของลูกศรก่อนและค่อยไปทำขั้นตอนที่หัวลูกศรชี้ไป ดังตัวอย่างในรูปที่ 4 ลูกศรชี้จากซ้ายไปขวา ดังนั้นเราจึงทำขั้นตอนด้านซ้ายก่อนแล้วจึงทำขั้นตอนทางด้านขวา

รูปที่ 4 ตัวอย่างการใช้ลูกศรในผังงาน

2) จุดเริ่มต้นและจุดสิ้นสุด (Terminator)

รูปที่ 5 สัญลักษณ์จุดเริ่มต้นและจุดสิ้นสุด

เราใช้สัญลักษณ์กล่องแคปซูลนี้เพื่อระบุจุดเริ่มต้นและจุดสิ้นสุดของอัลกอริทึม ซึ่งมีอย่างละหนึ่งในแต่ละอัลกอริทึม สำหรับจุดเริ่มต้น เราเขียนคำว่า Start ไว้ในกล่อง และสำหรับจุดสิ้นสุดเราเขียนคำว่า Stop ไว้

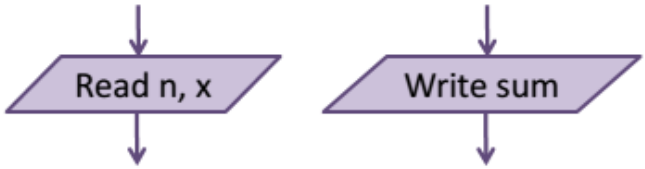
ในกล่อง กล่องสัญลักษณ์สำหรับจุดเริ่มต้น (start) จะมีลูกศรออกจากกล่อง 1 ลูกศรเท่านั้น ส่วนกล่องสัญลักษณ์สำหรับจุดสิ้นสุด (stop) จะมีลูกศรชี้เข้ากล่องเท่านั้น โดยไม่มีลูกศรชี้ออกไปเพราะสิ้นสุดอัลกอริทึมแล้ว ดังตัวอย่างข้างล่างนี้

รูปที่ 6 ตัวอย่างการใช้จุดเริ่มต้นและจุดสิ้นสุดของอัลกอริทึมในผังงาน

3) อินพุต (Input) และ เอาต์พุต (output)

รูปที่ 7 สัญลักษณ์อินพุตและเอาต์พุต

สัญลักษณ์ที่เราใช้แทนการระบุการรับอินพุตและการแสดงเอาต์พุตคือรูปสี่เหลี่ยมด้านขนาน สำหรับการรับอินพุต x มาจากผู้ใช้ เราเขียนว่า Read x ไว้ในรูปสี่เหลี่ยมด้านขนานนี้ และสำหรับการแสดงเอาต์พุต x เราเขียนว่า Print x หรือ Write x ไว้ในรูปสี่เหลี่ยมด้านขนาน ถ้ามีการรับอินพุตหรือการแสดงเอาต์พุตมากกว่าหนึ่งตัวต่อๆ กันโดยไม่มีขั้นตอนชนิดอื่นมากั้น เราสามารถเขียนรวมกันไปได้ในกล่องเดียวกันเลย แต่แนะนำให้เขียนการรับอินพุตแยกกล่องกับการแสดงเอาต์พุตเพื่อไม่ให้สับสน นอกจากนี้กล่องอินพุตและเอาต์พุตจะมีลูกศรออกจากกล่อง 1 ลูกศรเท่านั้น ส่วนลูกศรเข้ามายังกล่องสามารถมีมากกว่า 1 ลูกศรได้ ทั้งนี้ขึ้นอยู่กับอัลกอริทึมที่ผังงานแสดง ดังตัวอย่างในรูปที่ 8



รูปที่ 8 ตัวอย่างการเขียนการรับอินพุตและแสดงเอาต์พุตในผังงาน

4) กระบวนการ (Process)

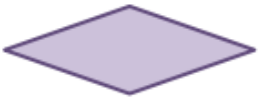
รูปที่ 9 สัญลักษณ์แสดงกระบวนการต่างๆ

เราใช้กล่องสี่เหลี่ยมเพื่อแสดงกระบวนการต่างๆ ในอัลกอริทึม เช่นการคำนวณ หรือการให้ค่าตัวแปร โดยเรานิยมใช้สัญลักษณ์ทางคณิตศาสตร์ในการเขียนอธิบายกระบวนการนั้นๆ ถ้ามีกระบวนการมากกว่าหนึ่งกระบวนการต่อๆ กันโดยไม่มีขั้นตอนชนิดอื่นมากั้น เราสามารถเขียนรวมกันไปในกล่องเดียวได้เลย นอกจากนี้

กล่องกระบวนการ (process) จะมีลูกศรออกจากกล่อง 1 ลูกศรเท่านั้น ส่วนลูกศรเข้ามายังกล่องสามารถมีมากกว่า 1 ลูกศรได้ ทั้งนี้ขึ้นอยู่กับอัลกอริทึมที่ผังงานแสดง ดังตัวอย่างในรูปที่ 10

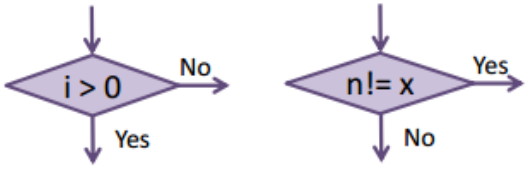
รูปที่ 10 ตัวอย่างการเขียนกระบวนการ (การคำนวณและการให้ค่า) ในผังงาน

5) เงื่อนไขหรือจุดให้ตัดสินใจ (Decision)



รูปที่ 11 สัญลักษณ์แสดงเงื่อนไขหรือจุดให้ตัดสินใจ

เป็นเรื่องปกติที่ในวิธีการแก้ปัญหาหนึ่งจะมีเงื่อนไขเพื่อให้เราเลือกทำสิ่งที่ต่างกันตามค่าที่กำหนดให้ ส่วนมากลักษณะของเงื่อนไขจะเป็นการเปรียบเทียบค่าต่างๆ ในการเขียนผังงาน เงื่อนไขสามารถเป็นได้หนึ่งในสองค่าคือเงื่อนไขเป็นจริงหรือเงื่อนไขเป็นเท็จ ตัวอย่างเงื่อนไข เช่น ถ้า x>0 เป็นจริง ให้ทำขั้นตอน A ไม่เช่นนั้นให้ทำขั้นตอน B เราแสดงเงื่อนไขหรือจุดที่ให้ตัดสินใจโดยใช้กล่องรูปสี่เหลี่ยมขนมเปียกปูนและเขียนเงื่อนไขไว้ในกล่อง กล่องเงื่อนไขนี้จะแตกต่างจะกล่องอื่นคือมีลูกศรออกจากกล่อง 2 ลูกศรในขณะที่กล่องชนิดอื่นมีลูกศรเดียว ทั้งนี้เพราะลูกศรตัวแรกนำไปสู่ขั้นตอนต่อไปในกรณีที่เงื่อนไขนั้นเป็นจริงและลูกศรอีกตัวนำไปสู่ขั้นตอนต่อไปในกรณีที่เงื่อนไขนั้นเป็นเท็จ ดังนั้นเวลาเขียนลูกศรเราต้องเขียน Yes หรือ True กำกับไว้ในกรณีที่เงื่อนไขนั้นเป็นจริง และเขียน No หรือ False กำกับไว้ในกรณีที่เงื่อนไขนั้นเป็นเท็จเสมอ ดังแสดงในรูปที่ 12



รูปที่ 12 ตัวอย่างการเขียนเงื่อนไขและเขียนผลของเงื่อนไขกำกับในทางเลือกต่างๆ

6) ตัวเชื่อมต่อ (Connector)

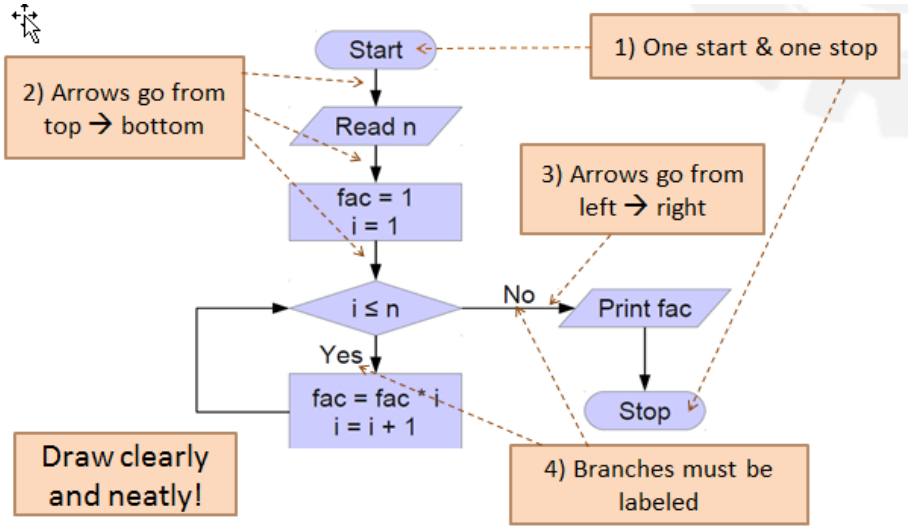
รูปที่ 13 สัญลักษณ์แสดงตัวเชื่อมต่อ

ในกรณีที่ผังงานซับซ้อนหรือใหญ่เกินพื้นที่ที่กำหนดไว้ เราจำเป็นจะต้องขยายผังงานออกไปยังพื้นที่อื่น เราต้องใช้ตัวเชื่อมต่อ (connector) ต่อผังงานส่วนต่างๆ เข้าด้วยกัน ตัวเชื่อมต่อมีสัญลักษณ์เป็นรูปวงกลมและมีตัวอักษรหรือข้อความสั้นๆ กำกับไว้ข้างใน โดยการเชื่อมต่อผังงานนั้นทำได้โดย ถ้าต้องการต่อผังงานไปที่อื่น

ให้เขียนตัวเชื่อมตัวเป็นขั้นตอนถัดไป คือชี้ลูกศรเข้าตัวเชื่อมต่อต่อจากขั้นตอนสุดท้ายในส่วนนั้น จากนั้นในพื้นที่ใหม่ (เช่น หน้าใหม่) ให้เขียนตัวเชื่อมต่อด้วยอักษรหรือข้อความเดียวกัน แล้วชี้ลูกศรออกจากตัวเชื่อมต่อไปยังขั้นตอนที่เราจะทำต่อไป เราก็จะรู้ได้ว่าขั้นตอนต่างๆ บนผังงานนั้นต่อกันเพราะมีตัวเชื่อมต่อที่มีอักษรเดียวกันกำกับอยู่ ดังแสดงตัวอย่างในรูปที่ 14

รูปที่ 14 ตัวอย่างการเชื่อมต่อผังงานที่มีขนาดใหญ่หรือซับซ้อน

3.2.2 กฎการเขียนผังงาน (Flow chart rules)



รูปที่ 15 กฎการเขียนผังงาน

การเขียนผังงานเพื่อให้อ่านง่ายและรวดเร็ว มีแนวทาง ดังนี้

1) ในหนึ่งผังงานจะมีจุดเริ่มแค่ 1 ที่ และจุดสิ้นสุดแค่ 1 ที่ โดยเรานิยมเขียนจุดเริ่มไว้ข้างบนสุด และจุดสิ้นสุดไว้ข้างล่าง
2) เรานิยมเขียนลูกศรจากบนลงล่าง นั่นคือขั้นตอนต่างๆ จะเริ่มจากด้านบนของพื้นที่และไหลลงด้านล่างในขั้นตอนถัดๆ ไป
3) ถ้าจำเป็นต้องเขียนขั้นตอนต่อไปในแนวนอน เช่นการเขียนขั้นตอนต่อจากเงื่อนไข ลูกศรจะวิ่งจากซ้ายไปขวา แต่เราก็สามารถเพิ่มเติมลูกศรที่วิ่งจากขวาไปซ้ายได้ เช่นจากกล่องเงื่อนไข เราสามารถเขียน

ขั้นตอนสำหรับเงื่อนไขที่เป็นจริงไว้ทางขวา (ลูกศรวิ่งจากซ้ายไปขวา) และขั้นตอนสำหรับเงื่อนไขที่เป็นเท็จไว้ทางซ้าย (ลูกศรวิ่งจากขวาไปซ้าย) ก็ได้

4) ลูกศรที่วิ่งออกจากกล่องเงื่อนไขจะต้องเขียน Yes/No หรือ True/False กำกับไว้เสมอ เพื่อเราจะได้รู้ว่าถ้าเป็นจริงจะไปขั้นตอนใดหรือถ้าเป็นเท็จจะไปขั้นตอนใด

ทั้งนี้ เราควรจะเขียนผังงานให้เรียบร้อย ชัดเจน และอ่านง่าย เพื่อจะนำผังงานนั้นไปใช้งานได้อย่างถูกต้อง

ข้อดีของการแสดงอัลกอริทึมด้วยผังงาน

1) การเขียนผังงานทำให้เราเข้าใจขั้นตอนวิธีการแก้ปัญหาได้ง่ายขึ้น
2) การเขียนผังงานเหมาะสำหรับการจัดทำเอกสารอธิบายกระบวนการต่างๆ รวมถึงการทำงานของโปรแกรม
3) เราสามารถใช้ผังงานเป็นแนวทางในการเขียนโค้ด ทำให้เขียนโค้ดได้ง่ายขึ้นเพราะมีขั้นตอนต่างๆ กำหนดไว้อย่างชัดเจนแล้ว

ถึงแม้การแสดงอัลกอริทึมด้วยผังงานจะมีข้อดีดังกล่าว แต่ก็มีข้อจำกัดคือความไม่เหมาะสมสำหรับโปรแกรมหรือกระบวนการที่ใหญ่มาก เพราะผังงานอาจจะใหญ่หรือสลับซับซ้อนเกินไปทำให้สับสนได้

3.3 ตัวอย่างของการเขียนอัลกอริทึมและผังงาน

อย่างที่กล่าวไปแล้วข้างต้น อัลกอริทึมคือขั้นตอนการแก้ปัญหา แต่ก่อนที่เราจะหาทางแก้ปัญหาได้ เมื่อได้ปัญหามา เราควรจะหาคำตอบให้กับสิ่งต่อไปนี้ก่อน เพื่อจะได้หาทางแก้ปัญหาได้ง่ายขึ้น

1) โจทย์ถามหาอะไร เราต้องหาอะไร นั่นคือ เอาต์พุตคืออะไร โดยปกติแล้วโจทย์จะระบุมาอย่างชัดเจน
2) โจทย์กำหนดให้อินพุตคืออะไร มีอินพุตหรือไม่ นั่นคือเราต้องรับอินพุตมาจากผู้ใช้หรือเปล่า โดยปกติแล้วโจทย์จะระบุมาอย่างชัดเจนถ้าต้องรับอินพุตจากแป้นพิมพ์หรือจากแฟ้มข้อมูล
3) โจทย์กำหนดอะไรอย่างอื่นมาให้บ้าง เช่นสูตรในการคำนวณเอาต์พุต คำอธิบายการทำงานของสิ่งต่างๆ หรือเงื่อนไขต่างๆ
4) ถ้าโจทย์ไม่ได้กำหนดวิธีการแก้ปัญหามาโดยตรง เรารู้หรือไม่ว่าจะแก้ปัญหาอย่างไร บางปัญหาเราจะต้องใช้ความรู้พื้นฐานที่เรามีเช่น พื้นฐานทางคณิตศาสตร์ ฟิสิกส์ หรือวิศวกรรม ในการหาทางแก้ปัญหา

เมื่อเรารู้คำตอบของ 4 ข้อข้างต้นแล้ว เราก็จะสามารถรู้ขั้นตอนของวิธีแก้ปัญหานั้นๆ และเขียนผังงานเพื่อแสดงวิธีแก้ปัญหานั้นได้

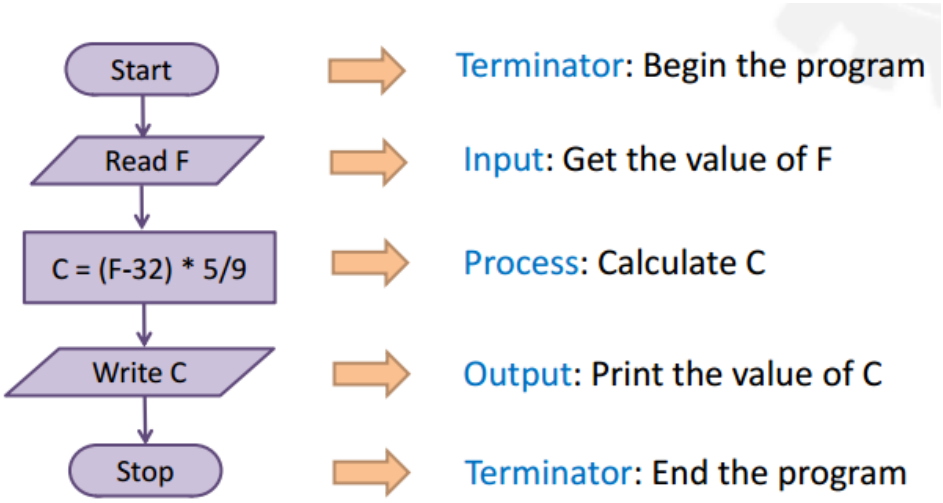
ตัวอย่างที่ 4 การแปลงองศาฟาเรนไฮต์เป็นองศาเซลเซียส

โจทย์ : จงรับอินพุตจากแป้นพิมพ์เป็นอุณหภูมิในหน่วยองศาฟาเรนไฮต์ แล้วแสดงอุณหภูมินั้นเป็นองศาเซลเซียสออกทางหน้าจอ

ก่อนจะเขียนผังงาน เราควรจะต้องตอบคำถาม 4 ข้อที่กล่าวไปข้างต้นแล้ว

1) เอาต์พุต คืออุณหภูมิในหน่วยองศาเซลเซียส
2) อินพุต คืออุณหภูมิในหน่วยองศาฟาเรนไฮต์
3) โจทย์ไม่ได้กำหนดสูตรมาให้
4) เราทราบว่าการแปลงจาก F เป็น C ทำได้โดยใช้สูตร C/5 = (F-32)/9 เพราะฉะนั้นถ้าเราจะแปลงจาก F เป็น C ต้องใช้สูตร C = (F-32)/9 * 5

จากความรู้ข้างต้น เราสามารถเขียนผังงานได้ดังนี้



รูปที่ 16 ผังงานการแปลงองศาฟาเรนไฮต์เป็นองศาเซลเซียส

เริ่มต้นด้วยจุดเริ่มต้น (กล่องแคปซูลที่เขียนว่า Start) จากนั้นเรารับอินพุต F มาจากแป้นพิมพ์ (กล่องสี่เหลี่ยมด้านขนานที่เขียนว่า Read F) เพื่อมาทำกระบวนการคำนวณหา C ตามสูตร C = (F-32)/9 * 5 (กล่องสี่เหลี่ยมผืนผ้า) เมื่อได้ C แล้วเราก็แสดงผลลัพธ์ นั่นคือเขียนเอาต์พุต C ออกทางหน้าจอ (กล่องสี่เหลี่ยมด้านขนานที่เขียนว่า Write C) และสุดท้ายจบผังงานด้วยจุดสิ้นสุด (กล่องแคปซูลที่เขียนว่า Stops)

ตัวอย่างที่ 5 การแปลงจากจำนวนวันเป็นจำนวนเดือนและเศษของเดือน

โจทย์ : จงรับจำนวนวันผ่านทางแป้นพิมพ์ จากนั้นให้แสดงออกทางหน้าจอว่าจำนวนวันนั้นคิดเป็นกี่เดือนและกี่วัน โดยกำหนดให้หนึ่งเดือนมี 30 วันเสมอ

ก่อนจะเขียนผังงาน เราควรจะต้องตอบคำถาม 4 ข้อที่กล่าวไปข้างต้นแล้ว

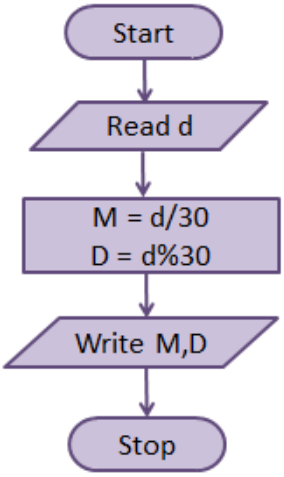
1) เอาต์พุต คือจำนวนเดือน (M) และเศษของเดือนเป็นวัน (D)

2) อินพุต คือจำนวนวัน (d)

3) โจทย์ไม่ได้กำหนดสูตรมาให้ แต่บอกว่า 1 เดือนมี 30 วัน

4) เราทราบว่า 1 เดือนมี 30 วันจากที่โจทย์กำหนดให้ เพราะฉะนั้นเราสามารถคำนวณหาจำนวน เดือนได้โดยนำ 30 ไปหารจำนวนวันที่รับเข้ามา (M = d/30) โดยปัดเศษลง (การใช้สัญลักษณ์ / กับเลขจำนวนเต็มจะทำการปัดเศษลงอยู่แล้วในภาษา C++) ส่วนจำนวนเศษของเดือนที่เป็นวันนั้น เราสามารถหาได้จากจำนวนวัน mod ด้วย 30 (D = d%30) หรือหาจากสมการ D = d – M*30

จากความรู้ข้างต้น เราสามารถเขียนผังงานได้ดังนี้



รูปที่ 17 ผังงานการแปลงจากจำนวนวันเป็นจำนวนเดือนและวัน

เราจะเห็นได้ว่าผังงานลักษณะนี้ไม่ได้ยากในการเขียนหรือการทำความเข้าใจ แต่ความท้าทายอยู่ที่การคิดอัลกอริทึมในการแก้ปัญหา ซึ่งในโจทย์ข้อนี้ความท้าทายอยู่ที่การหาค่าของ M และ D (จริงๆ แล้วการหา M และ D ในตัวอย่างนี้ถือว่าเป็นโจทย์พื้นฐานทางคณิตศาสตร์ และใช้ความรู้ในระดับมัธยมต้นเท่านั้น) โจทย์ปัญหาส่วนมากในทางวิศวกรรมไม่ว่าจะสาขาใดก็ตามจะเกี่ยวข้องกับคณิตศาสตร์ ฟิสิกส์ หรือเคมี ดังนั้นนอกจากจะเขียนผังงานเป็นแล้ว นักศึกษาควรจะต้องมีพื้นฐานวิชาดังกล่าวด้วยในการคิดอัลกอริทึม

ตัวอย่างที่ 6 การสลับค่าของตัวแปร **(swap)**

โจทย์ : จงรับค่า 2 ค่าผ่านทางแป้นพิมพ์มาเก็บไว้ในตัวแปรสองตัว จากนั้นให้ทำการสลับค่าตัวแปรทั้งสองตัวแล้วแสดงผลออกทางหน้าจอ

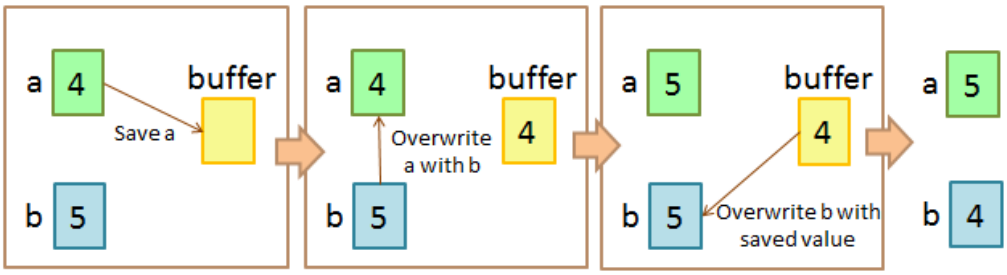
ก่อนจะเขียนผังงาน เราควรจะต้องตอบคำถาม 4 ข้อที่กล่าวไปข้างต้นแล้ว

1) เอาต์พุต คือค่าของตัวแปร a และ b ที่ได้ทำการสลับค่ากันแล้ว

2) อินพุต คือค่าของตัวแปร a และ b

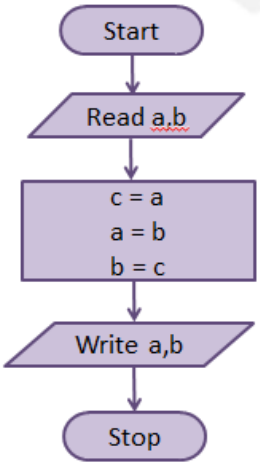
3) โจทย์ไม่ได้กำหนดสูตรมาให้

4) เราจะต้องกำหนดตัวแปรขึ้นมาอีก 1 ตัวคือ c เพื่อที่จะเป็นที่พักค่า (buffer) ของตัวแปรที่เราจะสลับ ดังตัวอย่างในรูปที่ 18



รูปที่ 18 อัลกอริทึมการสลับค่าของตัวแปร

สมมติให้ a และ b ที่รับเข้ามาผ่านทางแป้นพิมพ์มีค่าเท่ากับ 4 และ 5 ตามลำดับ การสลับค่าของตัวแปรจะทำให้ a = 5 และ b = 4 ในการเขียนโปรแกรมนั้น เราต้องทำการเปลี่ยนค่าตัวแปรทีละขั้นตอน นั่นคือ เปลี่ยนค่า a ก่อน แล้วจึงเปลี่ยนค่า b ตาม ดังนั้นเราจึงต้องมีตัวแปร c เพื่อเก็บค่าเดิมของ a ก่อนที่จะเปลี่ยนค่า a เป็นค่าของ b ไม่เช่นนั้น a จะถูกเขียนทับด้วยค่า b ทำให้ค่า a เดิมนั้นหายไป จากนั้นเราจึงเปลี่ยนค่า b โดยเอาค่ามาจากตัวแปร c (ค่า a เดิม) เท่านี้เราก็ได้ทำการสลับค่า a และ b เรียบร้อยแล้ว



รูปที่ 19 ผังงานของการสลับค่าของตัวแปร

รูปที่ 19 แสดงผังงานของการสลับค่าของตัวแปร ซึ่งเราจะเห็นได้ว่าผังงานนี้ไม่ได้สลับซับซ้อนเลย แต่การจะทำกระบวนการสลับตัวแปรให้ถูกต้องนั้นมีความท้าทายระดับหนึ่ง

ตัวอย่างที่ 7 การคำนวณเกรด

โจทย์ : จงรับคะแนนสอบของนักศึกษาคนหนึ่งในการสอบ 4 ครั้งมาจากแป้นพิมพ์ จากนั้นให้รวมคะแนนแล้วคิดเกรดจากเงื่อนไขที่กำหนดให้ตามตารางต่อไปนี้

คะแนน	เกรด

มากกว่า 800	A
601-800	B
401-600	C
201-400	D
200 ลงมา	F

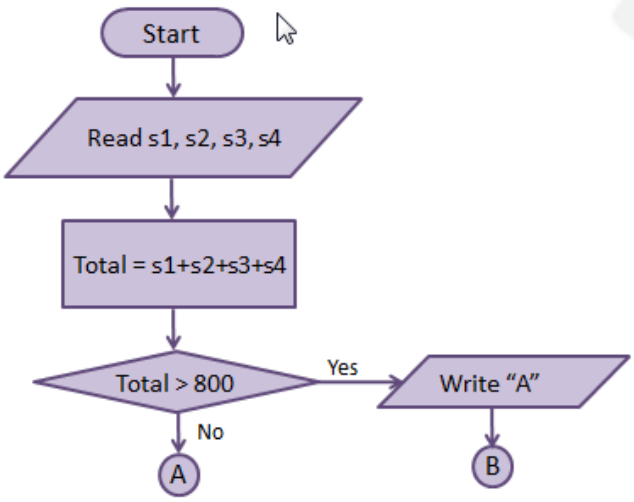
ก่อนจะเขียนผังงาน เราควรจะต้องตอบคำถาม 4 ข้อที่กล่าวไปข้างต้นแล้ว

1) เอาต์พุต คือเกรด A, B, C, D, หรือ F
2) อินพุต คือ คะแนนสอบของนักศึกษาคนหนึ่งในการสอบ 4 ครั้ง นั่นคือเราต้องเราอินพุตเข้ามา 4 ตัว
3) ให้รวมคะแนนทั้ง 4 ครั้ง และโจทย์ได้กำหนดเงื่อนไขของเกรดมาให้ตามตาราง
4) ไม่มีข้อมูลเพิ่มเติม

การแก้โจทย์ปัญหาข้อนี้คือการดูว่าคะแนนสอบของนักศึกษาตกอยู่ในช่วงคะแนนใด และให้เกรดตามช่วงคะแนนนั้น ซึ่งในการดูนี้เราจะต้องมีการเปรียบเทียบคะแนนในช่วงต่างๆ แล้วดูว่าคะแนนสอบนั้นอยู่ในช่วงที่เรากำลังเปรียบเทียบอยู่หรือไม่ ถ้าใช่ เราก็กำหนดเกรดให้และจบการทำโปรแกรม แต่ถ้าไม่ได้อยู่ในช่วงที่เรากำลังเปรียบเทียบ เราก็นำคะแนนสอบนั้นไปเปรียบเทียบคะแนนในช่วงอื่นต่อไป ดังแสดงในรูปที่ 20

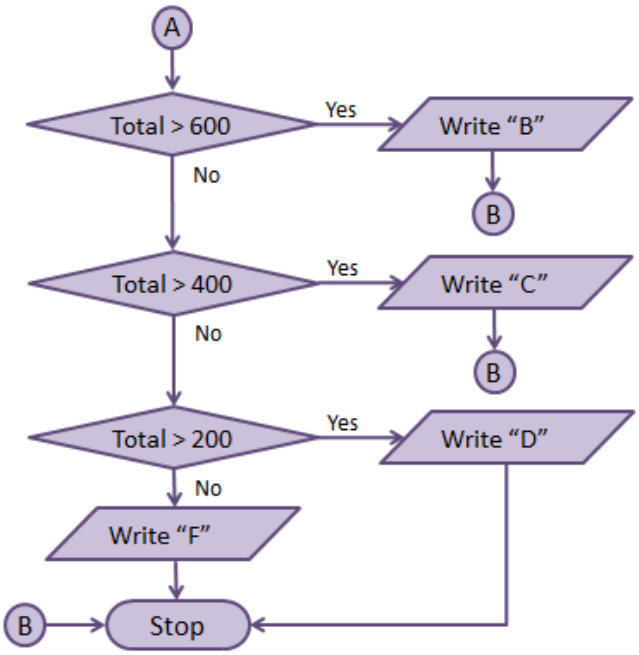
	Total Marks	Grade
598? →	> 800	A
598? →	601 – 800	B
598? →	401 – 600	C
598? →	201 – 400	D

รูปที่ 20 ตารางการให้เกรดตามช่วงคะแนน



รูปที่ 21 ผังงานการคำนวณเกรดส่วนที่หนึ่ง

จากผังงานส่วนแรกในรูปที่ 21 เราจะเห็นว่ามีการอ่านอินพุตเข้ามา 4 ตัวและทำการรวมผลอินพุตนั้นเก็บไว้ในตัวแปร Total จากนั้นก็เริ่มการเปรียบเทียบคะแนนในช่วงคะแนนต่างๆ ตามที่โจทย์กำหนดให้ โดยขั้นตอนแรกจะเอาคะแนนมาเปรียบเทียบกับ 800 ก่อนว่าคะแนนมากกว่า 800 หรือเปล่า ถ้ามากกว่า ให้แสดงทางหน้าจอว่าได้เกรด A สมมติว่าหน้ากระดาษเราหมดแค่นี้ (เพื่อแสดงตัวอย่างการใช้ตัวเชื่อมต่อ) เราจึงต้องทำขั้นตอนต่อไปด้วยการเชื่อมต่อผังงานไปที่หน้าอื่น โดยหลังจากแสดงเกรด A แล้วให้ไปต่อที่ขั้นตอน B แต่ถ้าเปรียบเทียบกับ 800 แล้วคะแนนไม่ได้มากกว่า (น้อยกว่าหรือเท่ากับ) ให้ไปทำที่ขั้นตอน A ในรูปที่ 22



รูปที่ 22 ผังงานการคำนวณเกรดส่วนที่สอง

ในรูปที่ 22 ผังงานเริ่มต้นด้วยตัวเชื่อม A นั่นคือถ้าเราทำการตรวจสอบเงื่อนไขในรูปที่ 21 และพบว่าเงื่อนไขที่ Total มากกว่า 800 นั้นเป็นเท็จแล้ว เราจะมาทำต่อในผังงานในรูปที่ 22 โดยเริ่มที่ตัวเชื่อม A คือการตรวจสอบว่า Total มากกว่า 600 หรือไม่ นั่นคือการเปรียบเทียบคะแนนรวม (Total) กับช่วงคะแนนต่างๆ หากพบว่า Total นั้นอยู่ในช่วงคะแนนใด ก็ให้แสดงเกรดของช่วงคะแนนนั้นออกทางหน้าจอ และไปทำขั้นตอนของตัวเชื่อม B นั่นคือการสิ้นสุดโปรแกรมต่อไป

ตัวอย่างนี้แสดงให้เห็นถึงการเขียนผังงานที่มีการตรวจสอบเงื่อนไขต่อกันหลายเงื่อนไขและมีการใช้ตัวเชื่อมสำหรับผังงานที่ซับซ้อนขึ้นและใหญ่กว่าพื้นที่ที่กำหนด

ตัวอย่างที่ 8 การหาแฟกทอเรียลแบบที่ 1

โจทย์ : จงรับค่าจำนวนเต็ม n มาจากแป้นพิมพ์ จากนั้นให้คำนวณหาค่า n!

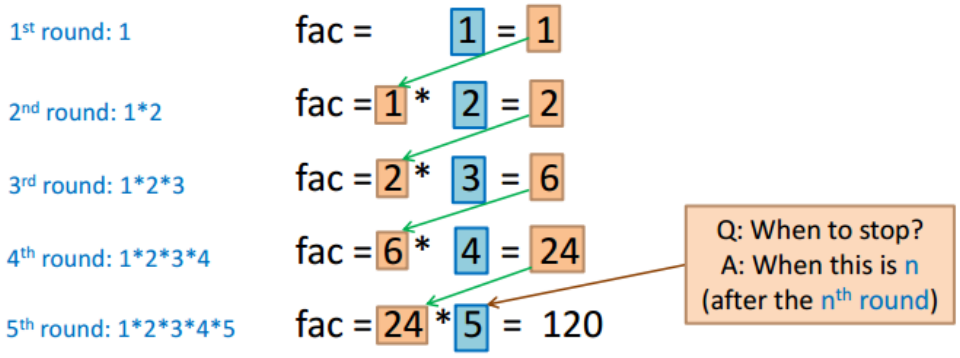
เช่นเดิม ก่อนจะเขียนผังงาน เราควรจะต้องตอบคำถาม 4 ข้อที่กล่าวไปข้างต้นแล้ว

1) เอาต์พุต คือค่า n! หรือค่าแฟกทอเรียลของ n

2) อินพุต คือจำนวนเต็ม n

3) โจทย์ไม่ได้กำหนดอะไรให้นอกเหนือจากนี้

4) เรารู้ว่า n! = n * (n-1) * (n-2) * ... * 1 และเรารู้อัลกอริทึมการหาค่า n! แบบเป็นขั้นตอนในตัวอย่างที่ 3 แต่ในตัวอย่างนี้เราจะลองเขียนผังงานโดยใช้นิยามที่ว่า n! = 1 * 2 * ... * (n-1) * n ซึ่งให้ค่าเท่ากันและขั้นตอนคล้ายกัน



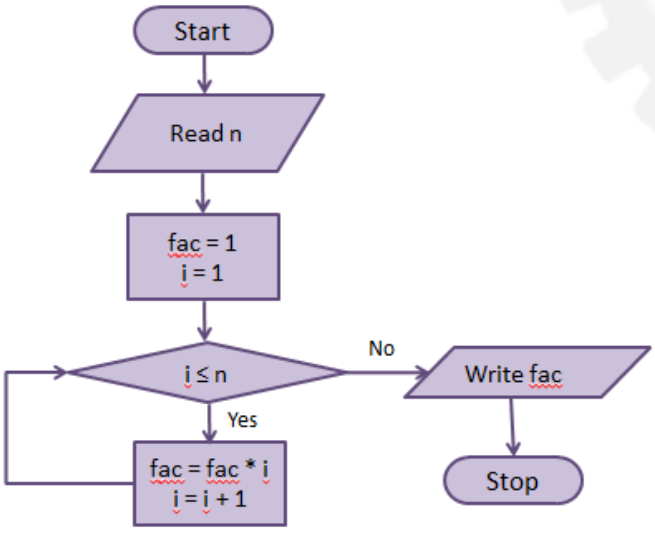
รูปที่ 23 ขั้นตอนการหา **n!** แบบที่ 1

จากรูปที่ 23 เราเห็นว่าการหาแฟกทอเรียลจะต้องมีการวนซ้ำโดย

1) มีการใช้ค่าจากรอบที่แล้วในการคำนวณค่าปัจจุบัน เราจึงต้องเก็บค่าล่าสุดไว้ในตัวแปรหนึ่งเสมอ

2) เราใช้ค่าจากรอบที่แล้วมาคูณกับจำนวนรอบที่วนซ้ำไปแล้วในปัจจุบัน และเก็บค่าใหม่เข้าไปที่ตัวแปรเดิม

3) มีการวนซ้ำเป็นจำนวน n รอบ นั่นคือหยุดการวนซ้ำหลังจากทำไปแล้ว n รอบ จึงต้องมีตัวแปรเพื่อนับว่าเราทำการวนซ้ำไปแล้วกี่รอบ และมีเงื่อนไขการหยุดวนซ้ำว่าถ้าจำนวนรอบเกิน n ให้หยุดทำการวนซ้ำ

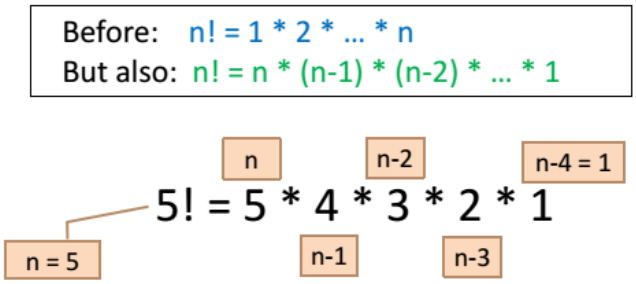


รูปที่ 24 ผังงานแสดงการหา **n!** แบบที่ 1

รูปที่ 24 แสดงผังงานของการหา n! แบบแรก โดยหลังจากอ่านอินพุตเข้ามาแล้ว เราต้องตั้งค่าตัวแปรสองตัวที่ได้กล่าวไปก่อนหน้า คือ fac และ i ให้เท่ากับ 1 โดย fac เป็นตัวแปรที่ใช้เก็บค่าการคูณในแต่ละรอบ ส่วน i เป็นตัวนับจำนวนรอบ โดย i จะถูกเพิ่มขึ้นทีละหนึ่งในแต่ละรอบตามที่ได้อธิบายไปก่อนหน้านี้

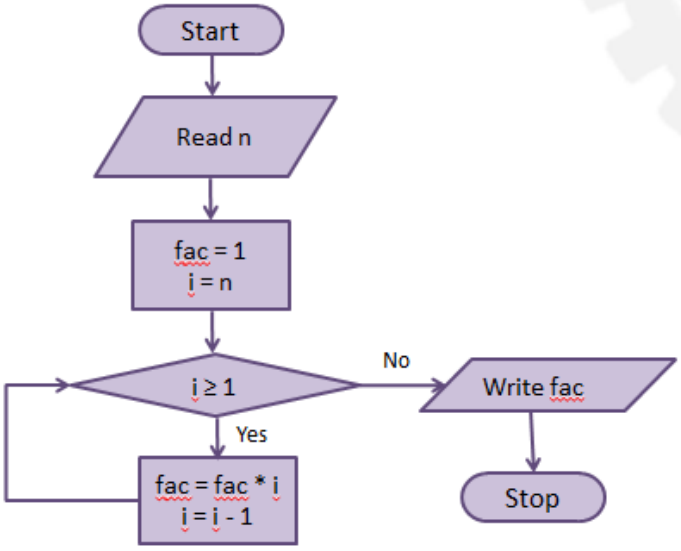
ตัวอย่างที่ 9 การหาแฟกทอเรียลแบบที่ 2

ในตัวอย่างนี้ โจทย์เหมือนกับตัวอย่างที่ 8 แต่ต่างกันตรงที่วิธีการหาแฟกทอเรียลเราจะใช้นิยามว่า n! = n * (n-1) * (n-2) * ... * 1 ตามรูปที่ 25



รูปที่ 25 ขั้นตอนการหา **n!** แบบที่ 2

จากผังงานในรูปที่ 26 เราเห็นได้ว่าขั้นตอนการแก้โจทย์การหาแฟกทอเรียลทั้งสองวิธีนั้นแทบจะเหมือนกัน ยกเว้นการให้ค่าเริ่มต้นแก่ i, เงื่อนไขในการทำซ้ำ และการปรับค่า i ในแต่ละรอบ เนื่องจากในตัวอย่างที่ 8 ค่าของ i ซึ่งเป็นทั้งจำนวนรอบและตัวคูณจะเพิ่มจาก 1 ไปจนถึง n แต่ในตัวอย่างนี้ i จะลดลงจาก n ไปจนถึง 1



รูปที่ 26 ผังงานแสดงการหา **n!** แบบที่ 2

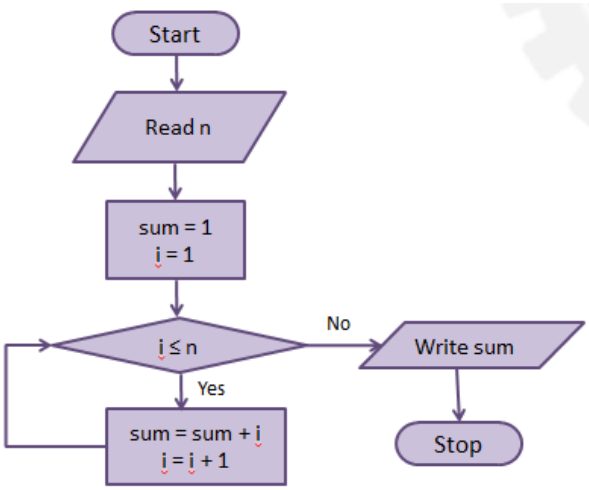
ตัวอย่างที่ 10 การหาผลรวม

โจทย์ : จงรับค่าจำนวนเต็ม n มาจากแป้นพิมพ์ จากนั้นให้หาผลรวมจาก 1 ถึง n โดยการวนซ้ำ

เช่นเดิม ก่อนจะเขียนผังงาน เราควรจะต้องตอบคำถาม 4 ข้อที่กล่าวไปข้างต้นแล้ว

1) เอาต์พุต คือค่าผลรวมจาก 1 ถึง n
2) อินพุต คือจำนวนเต็ม n
3) โจทย์ไม่ได้กำหนดอะไรให้นอกเหนือจากนี้
4) ไม่มีข้อมูลเพิ่มเติม

จากโจทย์ข้อนี้ เรารู้ว่าเราจะต้องบวกเลขจาก 1 ถึง n เป็นการวนทำซ้ำคล้ายๆ การหาแฟกทอเรียลคือเราจะต้องบวกเลขไปทีละตัวในแต่ละรอบ เพราะฉะนั้นจะต้องมีตัวแปรหนึ่งสำหรับเก็บค่าผลรวมในแต่ละรอบ และอีกตัวแปรหนึ่งสำหรับนับรอบและเป็นตัวบวกในแต่ละรอบ ดังแสดงในรูปที่ 27



รูปที่ 27 ผังงานแสดงการหาผลรวมของเลข n ตัว

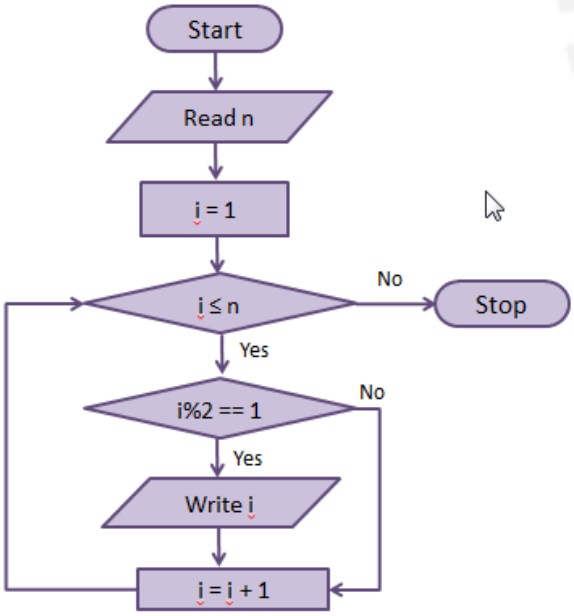
ตัวอย่างที่ 11 การพิมพ์เลขคี่

โจทย์ : จงรับค่าจำนวนเต็ม n มาจากแป้นพิมพ์ จากนั้นให้พิมพ์เลขคี่ที่อยู่ภายในช่วง 1 ถึง n ออกทางหน้าจอ

เช่นเดิม ก่อนจะเขียนผังงาน เราควรจะต้องตอบคำถาม 4 ข้อที่กล่าวไปข้างต้นแล้ว

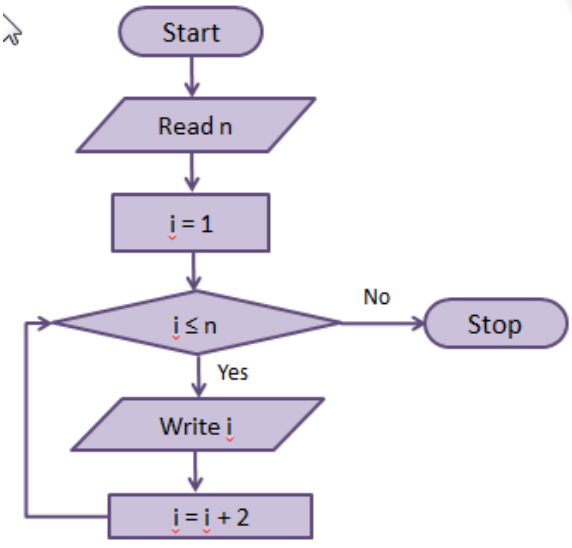
1) เอาต์พุต คือเลขคี่ที่อยู่ภายในช่วง 1 ถึง n
2) อินพุต คือจำนวนเต็ม n
3) โจทย์ไม่ได้กำหนดอะไรให้นอกเหนือจากนี้
4) เรารู้ว่าเลขคี่คือเลขที่หาร 2 ไม่ลงตัว

เราสามารถทำผังงานได้ 2 แบบ แบบแรกแสดงในรูปที่ 28 เราจะทำการวนซ้ำจาก 1 ถึง n ไปทีละตัว ในการวนแต่ละรอบเราต้องทำการตรวจสอบว่าค่า i เป็นเลขคี่หรือไม่ ถ้าเป็นเลขคี่ก็ให้พิมพ์ออก ถ้าไม่ใช่ก็ให้ข้ามไปรอบถัดไปเลย



รูปที่ 28 ผังงานแสดงการพิมพ์เลขคี่ แบบที่ 1

สำหรับผังงานแบบที่ 2 แสดงในรูปที่ 29 เราไม่ได้ทำการวนซ้ำไปทีละตัว แต่เราจะวนซ้ำตัวเว้นตัว โดยเริ่มจาก 1, 3, 5, ไปตามลำดับจนถึง n โดยสังเกตจากการปรับค่า i แทนที่เราจะเพิ่มค่า i ไปทีละหนึ่ง เราจะเพิ่มค่าไปทีละ 2 การทำแบบนี้ทำให้เราไม่ต้องตรวจสอบค่า i ในแต่ละรอบเพราะ i ในแต่ละรอบเป็นเลขคี่อยู่แล้ว



รูปที่ 29 ผังงานแสดงการพิมพ์เลขคี่ แบบที่ 2

3.4 สรุปใจความสำคัญของบทเรียน

1) เราเขียนผังงานเพื่อแสดงอัลกอริทึม หรือกระบวนการแก้ปัญหา เพื่อแสดงถึงการแก้ปัญหาแบบเป็นระบบ เป็นขั้นตอนที่ไม่ซับซ้อน ทำให้ง่ายต่อการเขียนโค้ดเพื่อแก้ปัญหาที่กำหนดให้
2) ผังงานจะมีจุดเริ่มต้น 1 ตำแหน่งและจุดสิ้นสุด 1 ตำแหน่งเท่านั้น
3) ควรเขียนผังงานให้เรียบร้อย และอย่าลืมเขียนข้อความกำกับตรงเงื่อนไขต่างๆ
4) เมื่อต้องการแก้ปัญหา เขียนผังงาน หรือเขียนโปรแกรม ให้ถามตัวเองว่า
    a. อะไรคือเอาต์พุต
    b. อะไรคืออินพุต
    c. โจทย์กำหนดอะไรมาให้บ้าง เช่นเงื่อนไข สูตรทางคณิตศาสตร์
    d. การแก้ปัญหานี้ต้องใช้ความรู้อื่นนอกเหนือจากที่โจทย์กำหนดให้ไหม
    e. อะไรคือกระบวนการแก้ปัญหา มีเงื่อนไขหรือไม่ ต้องทำซ้ำหรือไม่
    f. ควรจะเรียงลำดับในแต่ละขั้นตอนอย่างไรเพื่อจะแก้ปัญหานั้น
5) วิธีการแก้ปัญหาหนึ่งๆ สามารถมีได้หลายวิธี ดังนั้นเราสามารถมีหลายอัลกอริทึมในการแก้ปัญหาที่กำหนดให้ได้ (ไม่จำเป็นต้องทำเหมือนอาจารย์ในทุกขั้นตอน ถ้าผลลัพธ์นั้นถูกต้องตามที่โจทย์ต้องการ)

3.5 แบบฝึกหัดท้ายบท

จะเพิ่มเติมทีหลัง