ใบงานการทดลองที่ 6

เรื่อง การเขียนโปรแกรมเชิงวัตถุร่วมกับคลาสทางคณิตศาสตร์

1. จุดประสงค์ทั่วไป

1.1. รู้และเข้าใจในการติดต่อกับผู้ใช้และ การติดต่อระหว่างงาน

1.2. รู้และเข้าใจในการสร้างโปรแกรมเชิงวัตถุโดยใช้ภาษาโปรแกรมเชิงวัตถุใหม่ๆ

2. เครื่องมือและอุปกรณ์ เครื่องคอมพิวเตอร์1 เครื่อง ที่ติดตั้งโปรแกรม Eclipse

3. ทฤษฎีการทดลอง

3.1. ก่อนที่จะส่งข้อมูลจากฟอร์ม 1 ไปยังฟอร์ม 2 ควรมีการเตรียมตัวอย่างไร ?

การเตรียมข้อมูลจะเกิดขึ้นหลังผ่านขั้นตอนหลายอย่างที่เริ่มด้วยการเก็บรวบรวมข้อมูลที่ถูกต้อง ตามด้วยการทำความสะอาด การระบุประเภทข้อมูล การสอบทวน และการสร้างเป็นภาพ

3.2. ฟังก์ชันเรียกตัวเองคืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

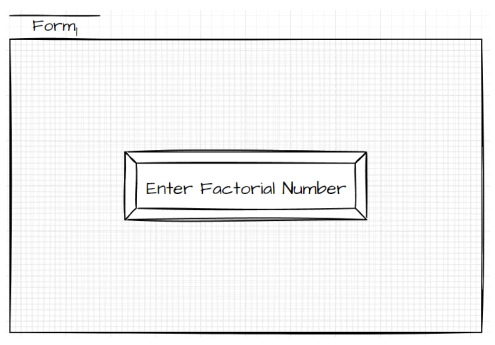
Recursive function (ฟังก์ชันรีเคอร์ชีพ) คือฟังก์ชันที่เรียกใช้ตัวเองเพื่อแก้ปัญหาบางอย่างโดยการแบ่งปัญหาให้เล็กลง จากนั้นรวมผลลัพธ์เข้าด้วยกัน ซึ่งการแก้ปัญหาในรูปแบบนี้เรียกว่า Divide-and-conquer ซึ่งเป็นวิธีการแก้ปัญหาที่พบได้ทั่วไปในชีวิตประวัน เช่น ในการบริหารจัดการองค์กร องค์กรหรือบริษัทนั้นจะถูกแบ่งออกเป็นหน่วยงาน (Division) และแต่ละหน่วยงานจะถูกแบ่งย่อยออกเป็นแผนกต่างๆ (Department) แต่ละแผนกก็จะแบ่งย่อยออกเป็นทีม

4. ลำดับขั้นการปฏิบัติการ

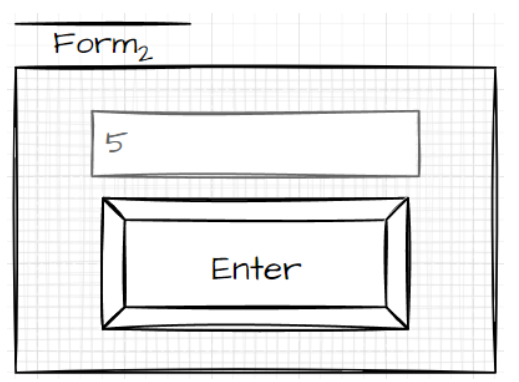
4.1. จงสร้าง Window Builder ในโปรแกรม Eclipse เพื่อสร้างโปรแกรมจำลองการทำงานเพื่อหาค่าของ Factorial ผ่านแบบ

จำลองแบบ Recursion บนโครงสร้างข้อมูลแบบ Stack โดยโปรแกรมจะมีการทำงานอยู่ 2 ฟอร์ม และมลีักษณะการทำงาน

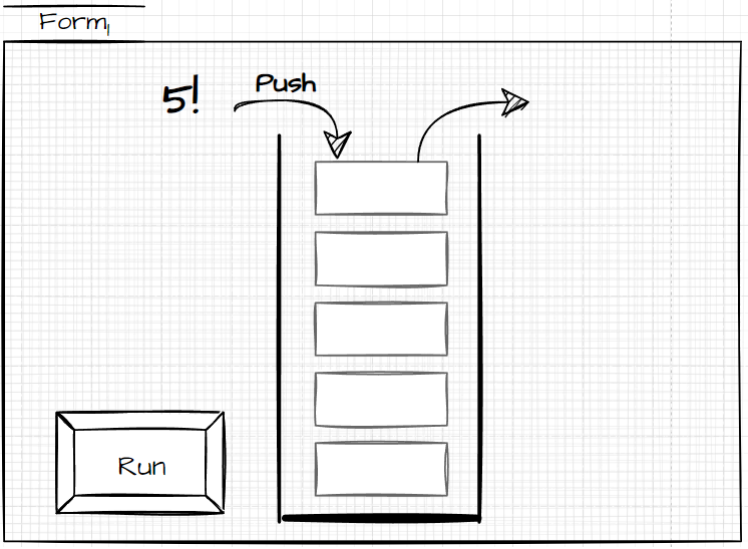
ดังต่อไปนี้

4.1.1. ฟอร์ม 1 โดยจะมีปุ่มเพื่อให้ผู้ใช้กด และเรียกหน้าต่าง ฟอร์ม 2 ขึ้นมา

4.1.2. ฟอร์ม 2 เป็นหน้าต่างใหม่ที่เตรียมให้ผู้ใช้กรอกเลขที่ต้องการหาค่า Factorial ลงไปในช่อง Textbox โดยที่ผู้ใช้จะถูกจำกัดให้กรอกได้เฉพาะเลข 1 ถึง 5 เท่านั้น

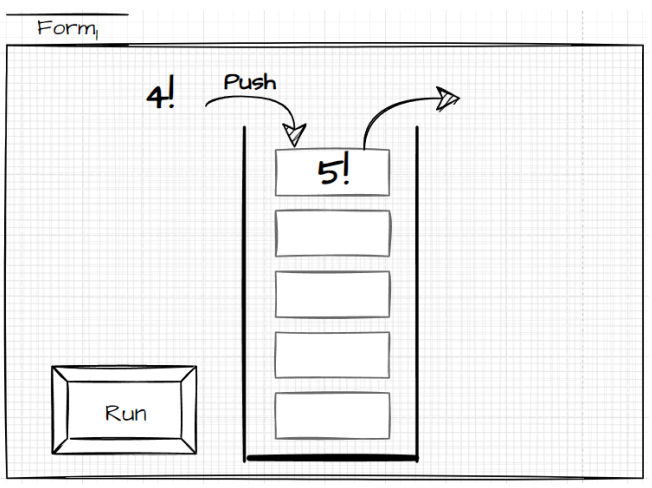


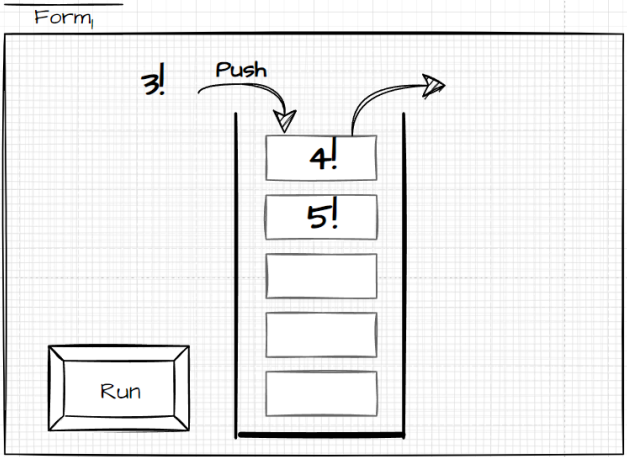
4.1.3. เมื่อกรอกข้อมูลในฟอร์ม 2 เสร็จแล้ว และกดปุ่ม Enter โปรแกรมจะนำเลข 5 ที่ได้จากช่อง Textbox ในฟอร์ม 2 ส่งค่ากลับไปยังฟอร์ม 1 อีกครั้ง และแสดงตัวเลขนั้นในช่องก่อนนำข้อมูล Push เข้าไปใน Stack เมื่อกดปุ่ม Run ทางด้านซ้ายล่าง ให้โปรแกรมทำการ Push ข้อมูล 5! เข้าไปใน Stack

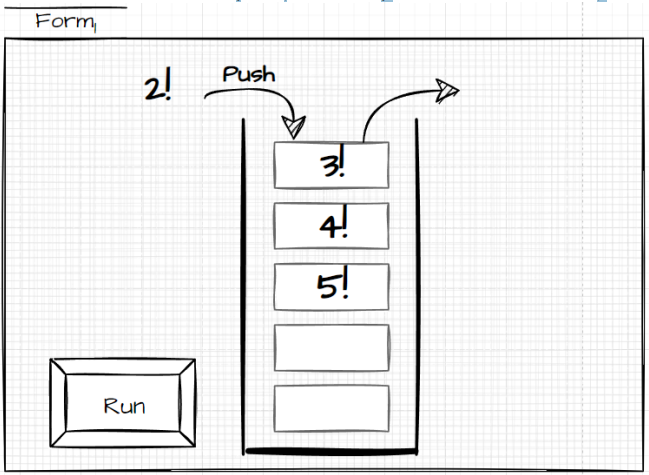


4.1.4. หลังจากกดปุ่ม Run เลข 5! จะเข้าไปอยู่ภายใน Stack และจะมีเลข 4! ทรี่ ออยู่ในตำแหน่งรอ Push เข้าไปใน Stack ดังนั้น

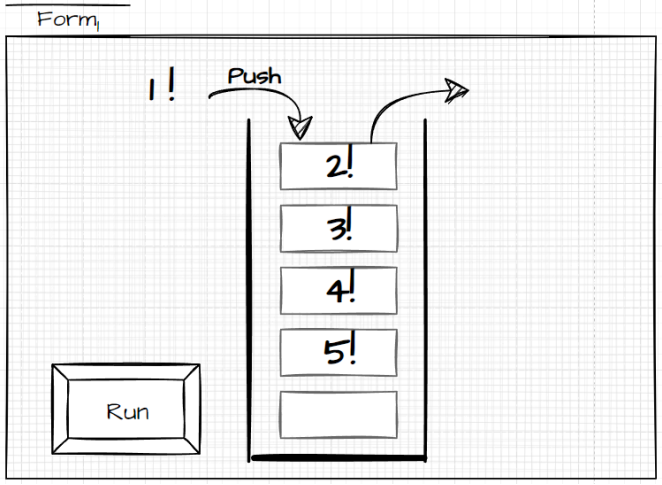
หากด้านบนสุดของ Stack ยังไม่ใช่เลข 1! เมื่อกดปุ่ม Run ระบบก็จะค่อยๆ นำข้อมูลเข้าไปใน Stack เรื่อยๆ



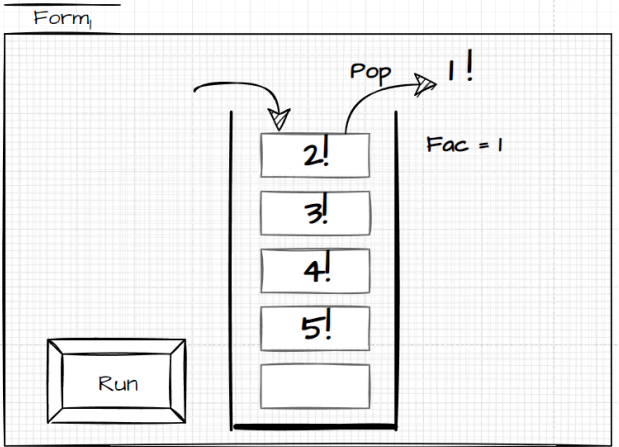
4.1.5. เช่นเดียวกันกับกรณีเมื่อครู่ หลังกดปุ่ม Run เลข 4! ก็จะถูก Push เข้าไปใน Stack ในตำแหน่งด้านบนสุด

4.1.6. เช่นเดียวกันกับกรณีเมื่อครู่ หลังกดปุ่ม Run เลข 3! ก็จะถูก Push เข้าไปใน Stack ในตำแหน่งด้านบนสุด

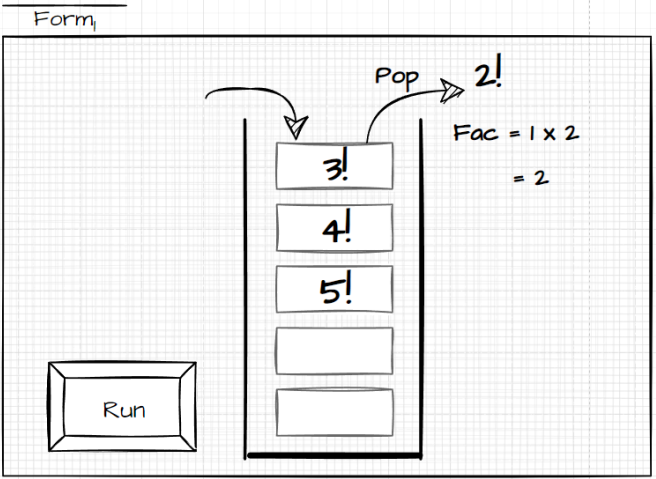
4.1.7. เช่นเดียวกันกับกรณีเมื่อครู่ หลังกดปุ่ม Run เลข 2! ก็จะถูก Push เข้าไปใน Stack ในตำแหน่งด้านบนสุด



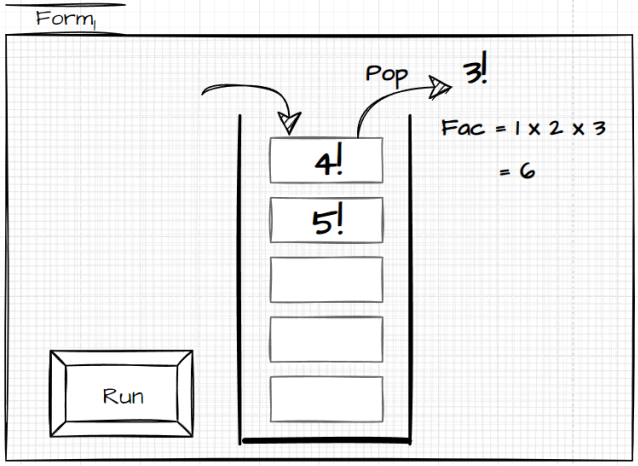
4.1.8. ทีนี้หลังจาก Push เลข 1! เข้าไปในระบบ จากกฎทวี่ ่า 1! = 1 ดังนั้นทำใหเ้ราสามารถหาคำตอบของ 1! ได้และเมื่อได้คำตอบให้ทำการ Pop เลข 1! ออกมา และใส่ไว้ในตัวแปร Fac พร้อมทั้งแสดงออกมาผ่านทาง Label เพื่อให้ผู้ใช้เห็นผลการคูณของชุดตัวเลข



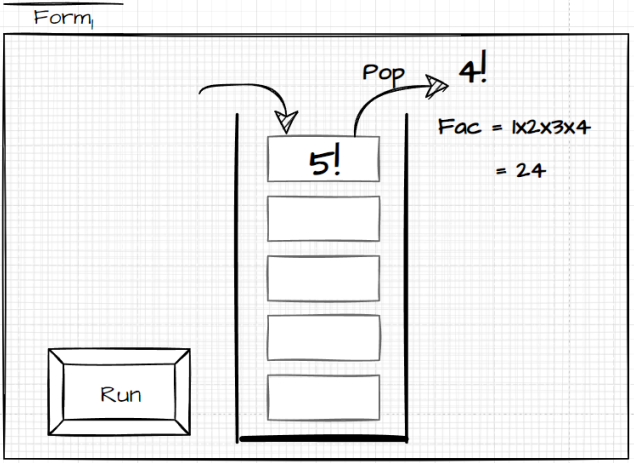
4.1.9. เมื่อกดปุ่ม Run อีกรอบ ระบบก็จะ Pop ตัวเลขบนสุดของ Stack ออกมา แล้วนำไปคูณค่า Fac ให้ผู้ใช้เห็นดังรูป



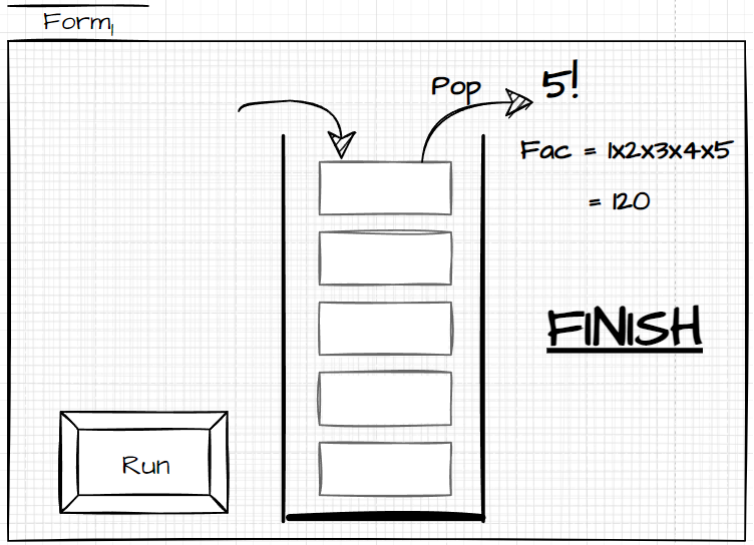
4.1.10. เมื่อกดปุ่ม Run อีกรอบ ระบบก็จะ Pop ตัวเลขบนสุดของ Stack ออกมา แล้วนำไปคูณค่า Fac ให้ผู้ใช้เห็นดังรูป



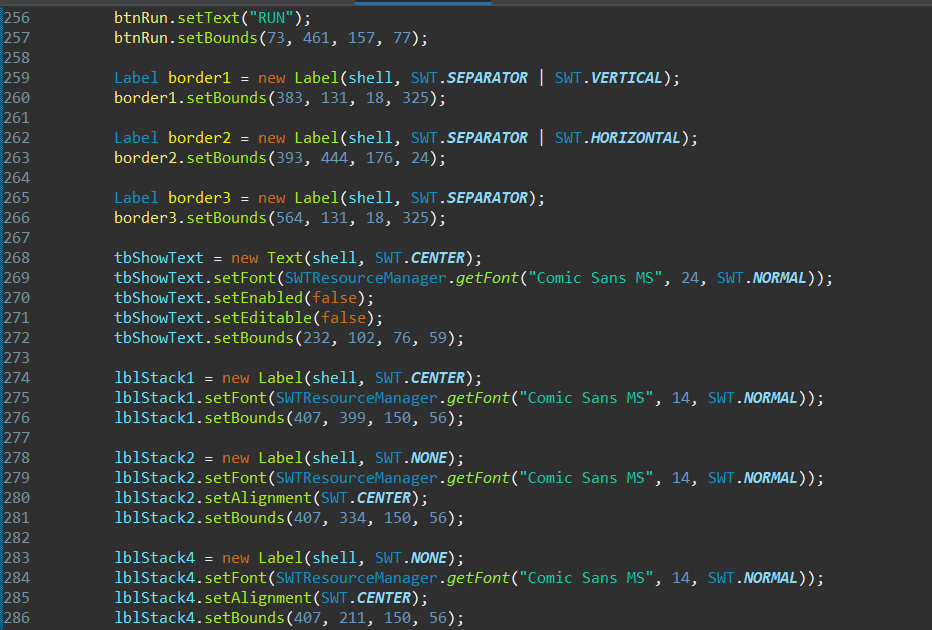
4.1.11. เมื่อกดปุ่ม Run อีกรอบ ระบบก็จะ Pop ตัวเลขบนสุดของ Stack ออกมา แล้วนำไปคูณค่า Fac ให้ผู้ใช้เห็นดังรูป



4.1.12. เมื่อกดปุ่ม Run อีกรอบ ระบบก็จะ Pop ตัวเลขบนสุดของ Stack ออกมา แล้วนำไปคูณค่า Fac ให้ผู้ใช้เห็นดังรูป และเมื่อถึงค่าสุดท้าย จะต้องปรากฏคำว่า “Finish” ขึ้นดังรูปด้วยเช่นกัน

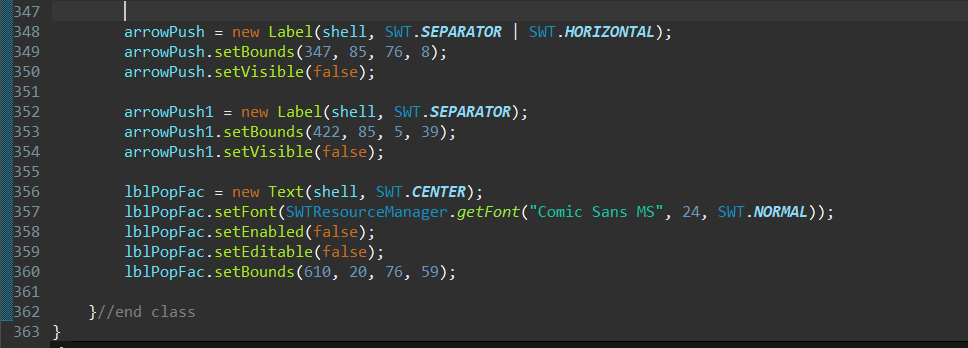




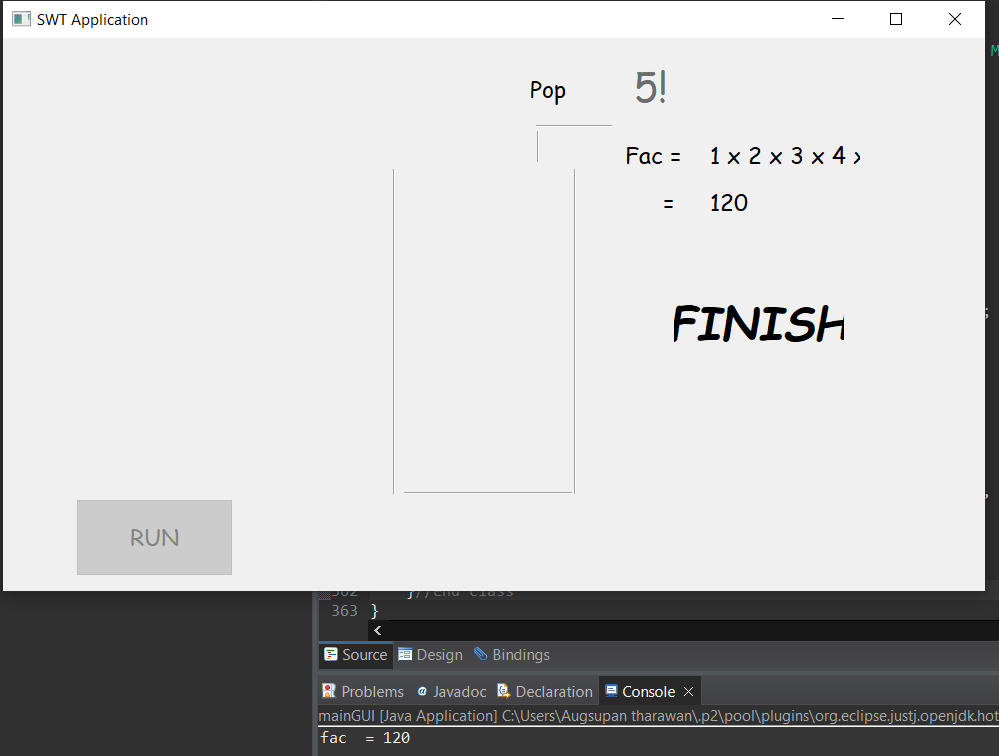








5. สรุปผลการปฏิบัติการ



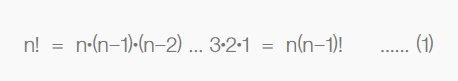
จากการทดสอบป้อนค่าเลขเข้าไปแล้วทำการกดปุ่มคำสั่ง RUN เพื่อให้ตัวแอพพลิเคชั่นทำงานคำนวณFactorial คำตอบที่ได้คือ 120

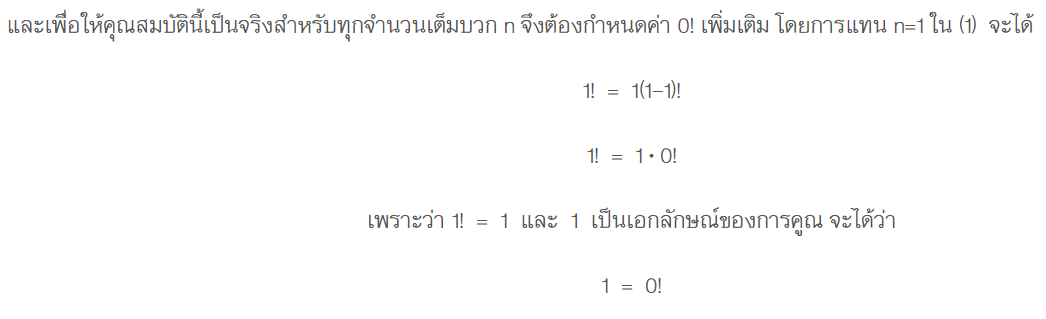
6. คำถามท้ายการทดลอง

6.1. ฟังก์ชันการทำงานใน Stack ควรมีอะไรบ้าง?

Stack เป็นโครงสร้างข้อมูลที่ถูกกล่าวถึงมากโครงสร้างหนึ่ง ซึ่งมักจะใช้เป็นประโยชน์ในการอินเตอร์รัพต์ การกระโดดไปมาระหว่างโปรแกรมย่อย การเขียนโปรแกรมแบบเรียกใช้ตัวเอง (recursive) นอกจากนั้นแล้วโครงสร้างข้อมูลชนิดนี้มักจะใช้ช่วยในการเข้าไปในโครงสร้างแบบพิเศษ เช่น เครือข่าย หรือต้นไม้ โดยจะช่วยในการจำเส้นทาง และงานที่เรานำโครงสร้างแบบ Stack แล้วเราพบเห็นบ่อยๆ คือ การยกเลิกคำสั่ง (Undo) ในไมโครซอฟท์เวิร์ด

6.2. การคำนวณ Factorial มีสูตรว่าอย่างไร ?





6.3. หลักการสร้าง Recursion คืออะไร?

Recursion คือวิธีการแก้ปัญหาแบบหนึ่งที่เกี่ยวกับการแตก ปัญหาเป็นปัญหาที่เล็กลงๆ จนกระทั่งปัญหาเล็กพอที่เราจะแก้ มันได้โดยง่าย โดยทั่วไปแล้ว recursion เกี่ยวกับฟังก์ชันที่เรียกตัวเอง แม้ว่ามันจะมองออกยาก แต่ recursion ทําให้เราสามารถเขียน คําตอบในรูปที่สวยงามของปัญหาได้ แม้ว่าจะเขียนโปรแกรม ยากก็ตาม

6.4. ข้อควรระวังในการส่งข้อมูลข้ามฟอร์มคืออะไร ?

จํากัดของแบบฟอร์มจะรวมสิ่งที่อยู่ในช่องเก็บรอบใหม่ด้วย (ตัวอย่างเช่น ฟอร์มที่ใช้งานอยู่ 350 ฟอร์มในพอร์ทัล Microsoft Forms + 50 ฟอร์มในถังรีไซเคิล = 400 ฟอร์ม) เมื่อผู้ตอบกรอกแบบฟอร์มและส่งแบบฟอร์มจะนับเป็นการตอบกลับหนึ่งครั้ง (ไม่ว่าจะมีคําถามกี่ข้อในแบบฟอร์มก็ตาม) ตัวอย่างเช่น ถ้าฟอร์มมีคําถาม 100 ข้อ และคําถามทั้งหมดถูกตอบโดยผู้ตอบ 12 คนจริงๆ แบบฟอร์มจะนับว่าได้รับคําตอบ 12 ครั้ง

สิ่งสำคัญ: แนวทาง "การตอบสนองหนึ่งครั้งต่อคน" จะบังคับใช้เฉพาะภายในชุดการตอบกลับ 50,000 ครั้งอย่างต่อเนื่องและไม่ได้รับการรับประกันสําหรับชุดข้อมูลแบบเต็มเมื่อมีการตอบกลับมากกว่า 50,000 ครั้ง