

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 ?

การเตรียมข้อมูลจะเกิดขึ้นหลังผ่านขั้นตอนหลายอย่างที่เริ่มด้วยการเก็บรวบรวมข้อมูลที่ถูกต้องตามด้วยการทำความสะอาด การระบุประเภทข้อมูล การสอบทวน และการสร้างเป็นภาพ

- 3.2. ฟังก์ชันเรียกตัวเองคืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

Recursive function (ฟังก์ชันรีเคอร์ซีฟ) คือฟังก์ชันที่เรียกใช้ตัวเองเพื่อแก้ปัญหาย่อยๆ โดยการแบ่งปัญหาให้เล็กลง จากนั้นรวมผลลัพธ์เข้าด้วยกัน ซึ่งการแก้ปัญหานี้เรียกว่า Divide-and-conquer ซึ่งเป็นวิธีการแก้ปัญหที่พบได้ทั่วไปในชีวิตประจำวัน เช่น ในการบริหารจัดการองค์กร องค์กรหรือบริษัทนั้นจะถูกแบ่งออกเป็นหน่วยงาน (Division) และแต่ละหน่วยงานจะถูกแบ่งย่อยออกเป็นแผนกต่างๆ (Department) แต่ละแผนกก็จะแบ่งย่อยออกเป็นทีม

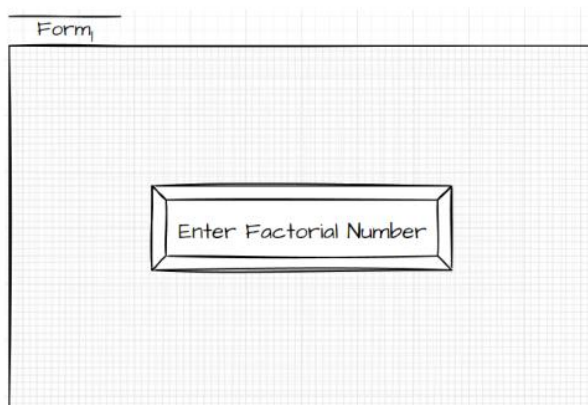
4. ลำดับขั้นการปฏิบัติการ

4.1. จงสร้าง Window Builder ในโปรแกรม Eclipse เพื่อสร้างโปรแกรมจำลองการทำงานเพื่อหาค่าของ Factorial ผ่านแบบ

จำลองแบบ Recursion บนโครงสร้างข้อมูลแบบ Stack โดยโปรแกรมจะมีการทำงานอยู่ 2 ฟอर्म และมีลักษณะการทำงาน

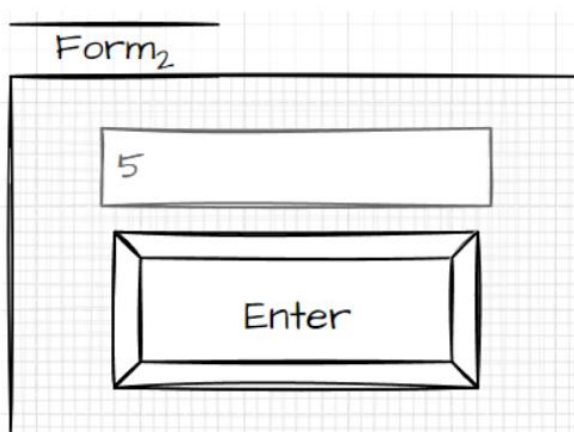
ดังต่อไปนี้

4.1.1. ฟอर्म 1 โดยจะมีปุ่มเพื่อให้ผู้ใช้กด และเรียกหน้าต่าง ฟอर्म 2 ขึ้นมา



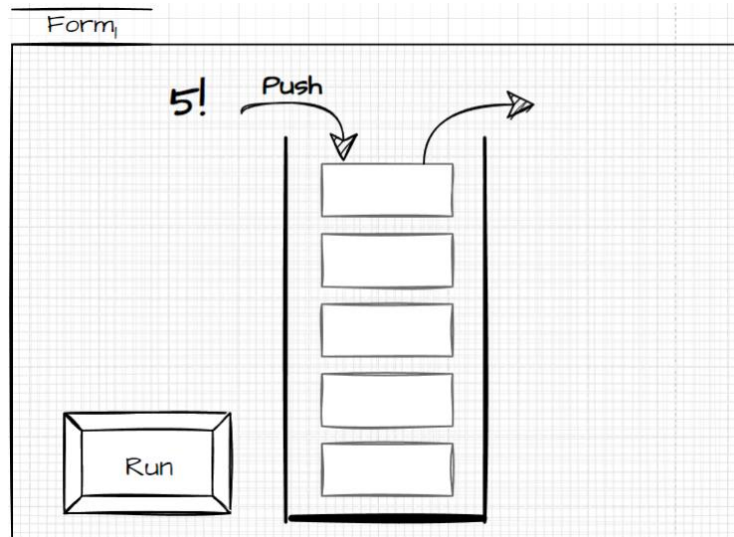
The diagram shows a window titled "Form₁" on a grid background. Inside the window is a rectangular button with a 3D effect, labeled "Enter Factorial Number".

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



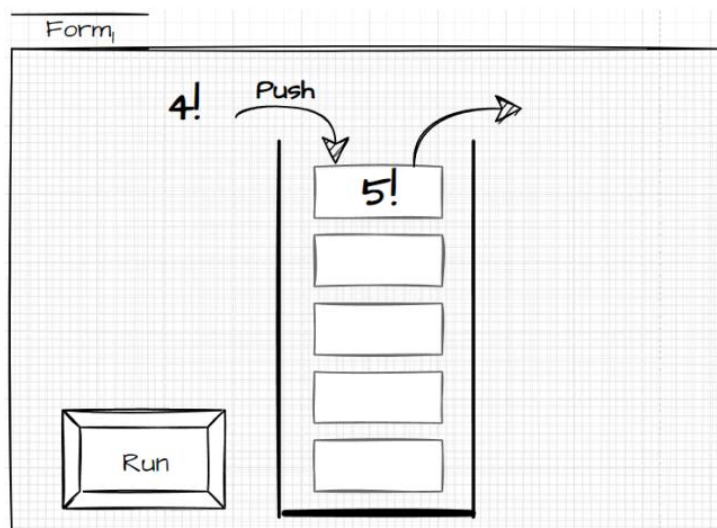
The diagram shows a window titled "Form₂" on a grid background. Inside the window, there is a text box containing the number "5" and a rectangular button with a 3D effect labeled "Enter" positioned below it.

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



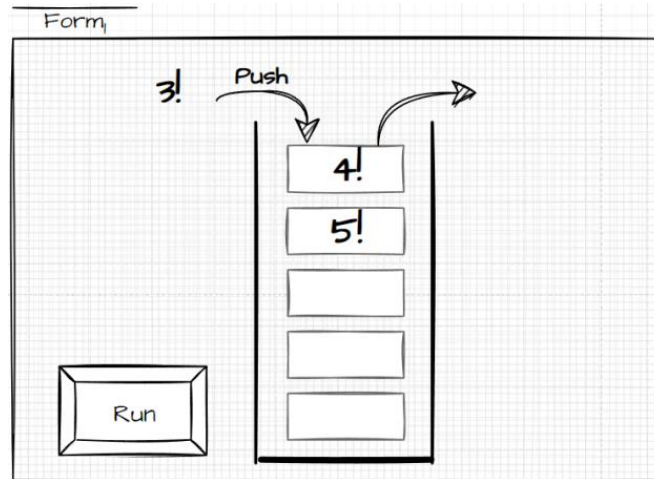
4.1.4. หลังจากกดปุ่ม Run เลข 5! จะเข้าไปอยู่ภายใน Stack และจะมีเลข 4! ที่รี อยู่ตำแหน่งรอ Push เข้าไปใน Stack ดังนั้น

หากด้านบนสุดของ Stack ยังไม่ใช่เลข 1! เมื่อกดปุ่ม Run ระบบก็จะค่อยๆ นำข้อมูลเข้าไปใน Stackเรื่อยๆ



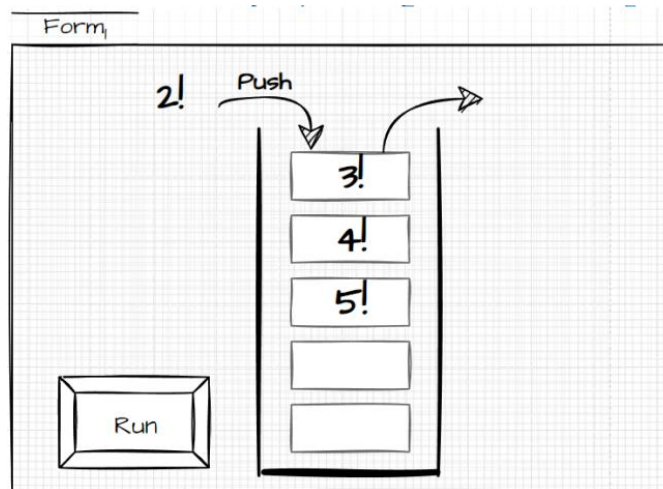
4.1.5. เช่นเดียวกันกับกรณีเมื่อครู หลังกดปุ่ม Run เลข 4! ก็จะถูก Push เข้าไปใน Stack ในตำแหน่งด้าน

บนสุด

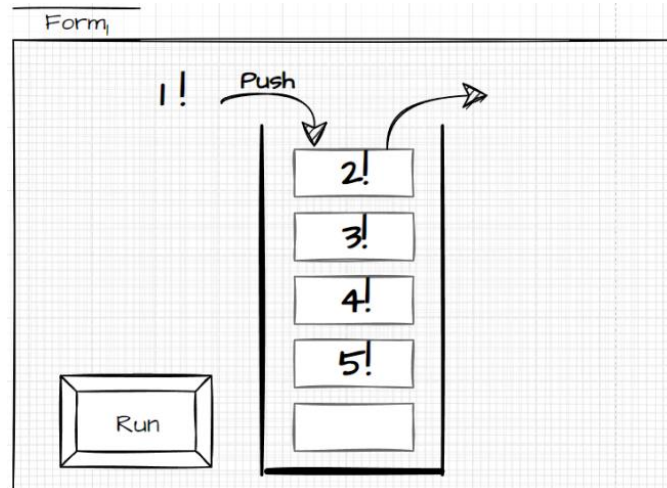


4.1.6. เช่นเดียวกันกับกรณีเมื่อครู หลังกดปุ่ม Run เลข 3! ก็จะถูก Push เข้าไปใน Stack ในตำแหน่งด้าน

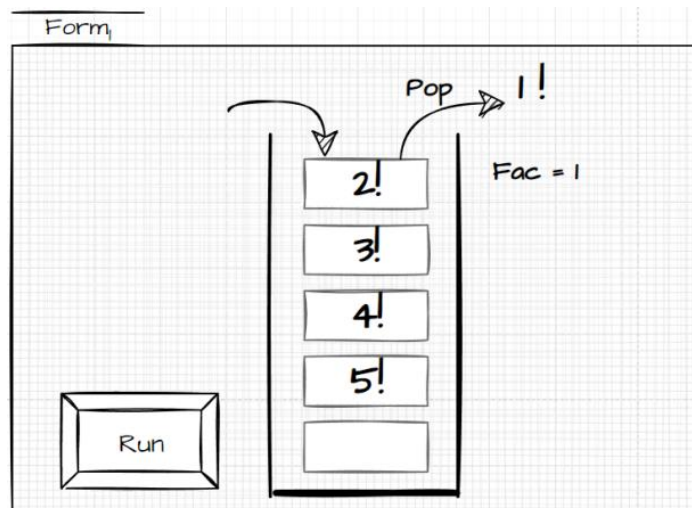
บนสุด



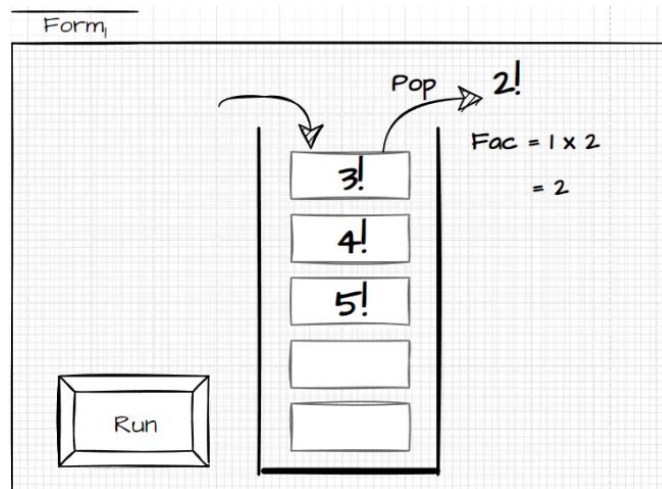
4.1.7. เช่นเดียวกันกับกรณีเมื่อครั้ง หลังกดปุ่ม Run เลข 2! ก็จะถูก Push เข้าไปใน Stack ในตำแหน่งด้านบนสุด



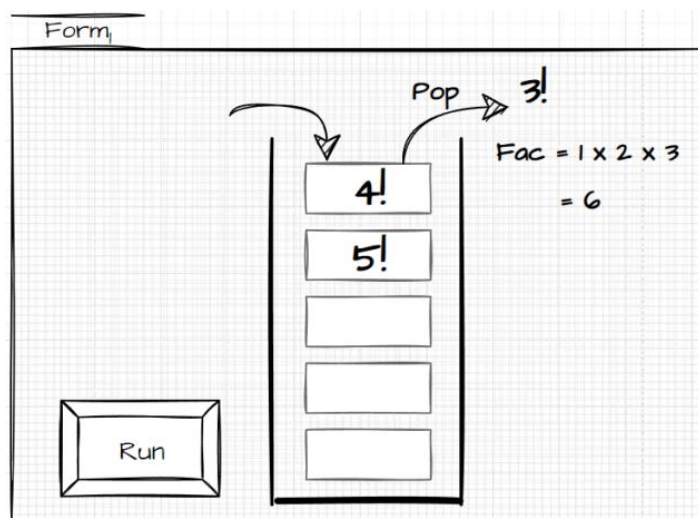
4.1.8. ทีนี้หลังจาก Push เลข 1! เข้าไปในระบบ จากกฎที่ว่า $1! = 1$ ดังนั้นทำให้เราสามารถหาคำตอบของ 1! ได้และเมื่อได้คำตอบให้ทำการ Pop เลข 1! ออกมา และใส่ไว้ในตัวแปร Fac พร้อมทั้งแสดงออกมาผ่านทาง Label เพื่อให้ผู้ใช้เห็นผลการคูณของชุดตัวเลข



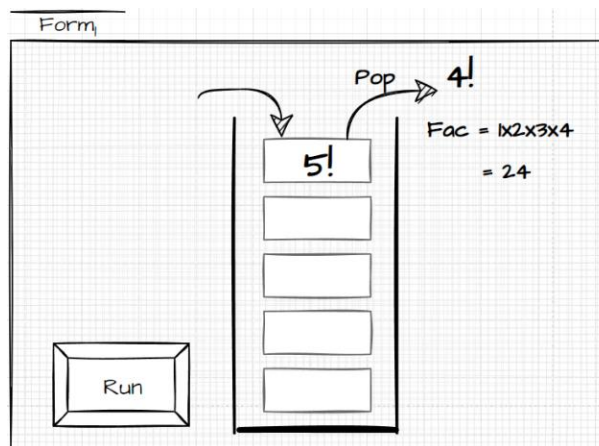
4.1.9. เมื่อกดปุ่ม Run อีกรอบ ระบบก็จะ Pop ตัวเลขบนสุดของ Stack ออกมา แล้วนำไปคูณค่า Fac ให้ผู้ใช้
เห็นดังรูป



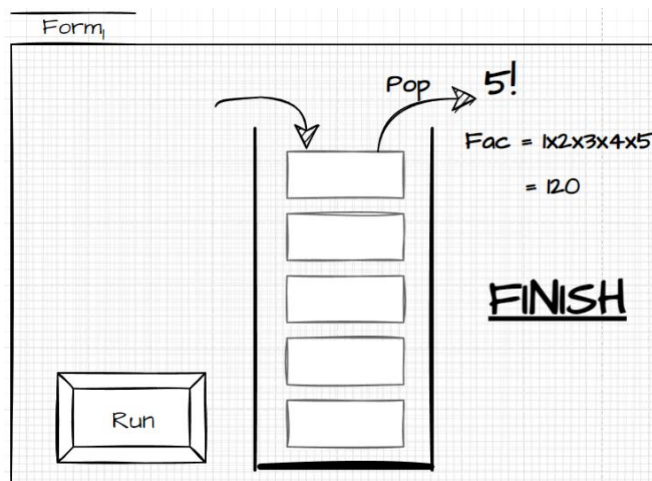
4.1.10. เมื่อกดปุ่ม Run อีกรอบ ระบบก็จะ Pop ตัวเลขบนสุดของ Stack ออกมา แล้วนำไปคูณค่า Fac ให้ผู้ใช้
เห็นดังรูป



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



4.2. จงเขียนโค้ดโปรแกรมที่อยู่ภายในปุ่ม Run

โค้ดโปรแกรมภายในปุ่ม Run

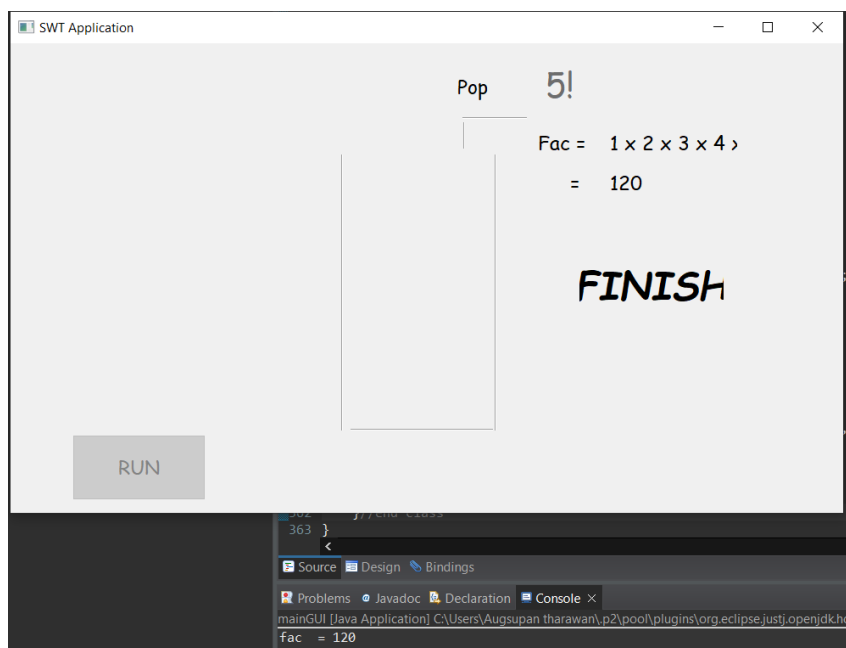
```
256     btnRun.setText("RUN");
257     btnRun.setBounds(73, 461, 157, 77);
258
259     Label border1 = new Label(shell, SWT.SEPARATOR | SWT.VERTICAL);
260     border1.setBounds(383, 131, 18, 325);
261
262     Label border2 = new Label(shell, SWT.SEPARATOR | SWT.HORIZONTAL);
263     border2.setBounds(393, 444, 176, 24);
264
265     Label border3 = new Label(shell, SWT.SEPARATOR);
266     border3.setBounds(564, 131, 18, 325);
267
268     tbShowText = new Text(shell, SWT.CENTER);
269     tbShowText.setFont(SWTResourceManager.getFont("Comic Sans MS", 24, SWT.NORMAL));
270     tbShowText.setEnabled(false);
271     tbShowText.setEditable(false);
272     tbShowText.setBounds(232, 102, 76, 59);
273
274     lblStack1 = new Label(shell, SWT.CENTER);
275     lblStack1.setFont(SWTResourceManager.getFont("Comic Sans MS", 14, SWT.NORMAL));
276     lblStack1.setBounds(407, 399, 150, 56);
277
278     lblStack2 = new Label(shell, SWT.NONE);
279     lblStack2.setFont(SWTResourceManager.getFont("Comic Sans MS", 14, SWT.NORMAL));
280     lblStack2.setAlignment(SWT.CENTER);
281     lblStack2.setBounds(407, 334, 150, 56);
282
283     lblStack4 = new Label(shell, SWT.NONE);
284     lblStack4.setFont(SWTResourceManager.getFont("Comic Sans MS", 14, SWT.NORMAL));
285     lblStack4.setAlignment(SWT.CENTER);
286     lblStack4.setBounds(407, 211, 150, 56);
287
288     lblStack3 = new Label(shell, SWT.NONE);
289     lblStack3.setFont(SWTResourceManager.getFont("Comic Sans MS", 14, SWT.NORMAL));
290     lblStack3.setAlignment(SWT.CENTER);
291     lblStack3.setBounds(407, 272, 150, 56);
292
293     lblStack5 = new Label(shell, SWT.NONE);
294     lblStack5.setFont(SWTResourceManager.getFont("Comic Sans MS", 14, SWT.NORMAL));
295     lblStack5.setAlignment(SWT.CENTER);
296     lblStack5.setBounds(407, 149, 150, 56);
297
298     lblPush = new Label(shell, SWT.NONE);
299     lblPush.setFont(SWTResourceManager.getFont("Comic Sans MS", 14, SWT.NORMAL));
300     lblPush.setBounds(347, 36, 54, 24);
301     lblPush.setText("Push");
302     lblPush.setVisible(false);
303
304     lblPop = new Label(shell, SWT.NONE);
305     lblPop.setFont(SWTResourceManager.getFont("Comic Sans MS", 14, SWT.NORMAL));
306     lblPop.setText("Pop");
307     lblPop.setBounds(527, 36, 41, 31);
308     lblPop.setVisible(false);
309
310     lblFac = new Label(shell, SWT.NONE);
311     lblFac.setFont(SWTResourceManager.getFont("Comic Sans MS", 14, SWT.NORMAL));
312     lblFac.setText("Fac = ");
313     lblFac.setBounds(622, 102, 55, 31);
314     lblFac.setVisible(false);
```


4.2. จงเขียนโค้ดโปรแกรมที่อยู่ภายในปุ่ม Run

โค้ดโปรแกรมภายในปุ่ม Run

```
316 lblFac_1 = new Label(shell, SWT.NONE);
317 lblFac_1.setFont(SWTResourceManager.getFont("Comic Sans MS", 14, SWT.NORMAL));
318 lblFac_1.setAlignment(SWT.CENTER);
319 lblFac_1.setText("=");
320 lblFac_1.setBounds(646, 149, 41, 31);
321 lblFac_1.setVisible(false);
322
323 lblShowCal = new Label(shell, SWT.NONE);
324 lblShowCal.setFont(SWTResourceManager.getFont("Comic Sans MS", 14, SWT.NORMAL));
325 lblShowCal.setBounds(707, 102, 150, 34);
326 lblShowCal.setVisible(false);
327
328 lblShowSumFac = new Label(shell, SWT.NONE);
329 lblShowSumFac.setFont(SWTResourceManager.getFont("Comic Sans MS", 14, SWT.NORMAL));
330 lblShowSumFac.setBounds(707, 149, 150, 34);
331 lblShowSumFac.setVisible(false);
332
333 lblShowFinish = new Label(shell, SWT.CENTER);
334 lblShowFinish.setAlignment(SWT.CENTER);
335 lblShowFinish.setFont(SWTResourceManager.getFont("Comic Sans MS", 27, SWT.BOLD | SWT.ITALIC));
336 lblShowFinish.setText("FINISH");
337 lblShowFinish.setBounds(671, 252, 170, 59);
338 lblShowFinish.setVisible(false);
339
340 arrowPop = new Label(shell, SWT.SEPARATOR | SWT.HORIZONTAL);
341 arrowPop.setBounds(533, 85, 76, 8);
342 arrowPop.setVisible(false);
343
344 arrowPop2 = new Label(shell, SWT.SEPARATOR);
345 arrowPop2.setBounds(534, 85, 5, 39);
346 arrowPop2.setVisible(false);
347
348 arrowPush = new Label(shell, SWT.SEPARATOR | SWT.HORIZONTAL);
349 arrowPush.setBounds(347, 85, 76, 8);
350 arrowPush.setVisible(false);
351
352 arrowPush1 = new Label(shell, SWT.SEPARATOR);
353 arrowPush1.setBounds(422, 85, 5, 39);
354 arrowPush1.setVisible(false);
355
356 lblPopFac = new Text(shell, SWT.CENTER);
357 lblPopFac.setFont(SWTResourceManager.getFont("Comic Sans MS", 24, SWT.NORMAL));
358 lblPopFac.setEnabled(false);
359 lblPopFac.setEditable(false);
360 lblPopFac.setBounds(610, 20, 76, 59);
361
362 } //end class
363 }
```

5. สรุปผลการปฏิบัติการ



จากการทดสอบป้อนค่าเลขเข้าไปแล้วทำการกดปุ่มคำสั่ง RUN เพื่อให้ตัวแอปพลิเคชันทำงาน
คำนวณ Factorial คำตอบที่ได้คือ 120

6. คำถามท้ายการทดลอง

6.1. ฟังก์ชันการทำงานใน Stack ควรมีอะไรบ้าง?

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

6.2. การคำนวณ Factorial มีสูตรว่าอย่างไร ?

$$n! = n \cdot (n-1) \cdot (n-2) \dots 3 \cdot 2 \cdot 1 = n(n-1)! \dots (1)$$

และเพื่อให้คุณสมบัตินี้เป็นจริงสำหรับทุกจำนวนเต็มบวก n จึงต้องกำหนดค่า $0!$ เพิ่มเติม โดยการแทน $n=1$ ใน (1) จะได้

$$1! = 1(1-1)!$$

$$1! = 1 \cdot 0!$$

เพราะว่า $1! = 1$ และ 1 เป็นเอกลักษณ์ของการคูณ จะได้ว่า

$$1 = 0!$$

6.3. หลักการสร้าง Recursion คืออะไร?

Recursion คือวิธีการแก้ปัญหาแบบหนึ่งที่เกี่ยวข้องกับการแตก ปัญหาเป็นปัญหาที่เล็กลงๆ จนกระทั่งปัญหาเล็กพอที่เราจะแก้ มันได้โดยง่าย โดยทั่วไปแล้ว recursion เกี่ยวข้องกับฟังก์ชันที่เรียกตัวเอง 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