

## ใบงานการทดลองที่ 6

### เรื่อง การเขียนโปรแกรมเชิงวัตถุร่วมกับคลาสทางคณิตศาสตร์

#### 1. จุดประสงค์ทั่วไป

- 1.1. รู้และเข้าใจในการติดต่อกับผู้ใช้ และการติดต่อระหว่างงาน
- 1.2. รู้และเข้าใจในการสร้างโปรแกรมเชิงวัตถุโดยใช้ภาษาโปรแกรมเชิงวัตถุใหม่ๆ

#### 2. เครื่องมือและอุปกรณ์

เครื่องคอมพิวเตอร์ 1 เครื่อง ที่ติดตั้งโปรแกรม Eclipse

#### 3. ทฤษฎีการทดลอง

- 3.1. ก่อนที่จะส่งข้อมูลจากฟอร์ม 1 ไปยังฟอร์ม 2 ควรมีการเตรียมตัวอย่างไร ?

3.1. ก่อนที่จะส่งข้อมูลจากฟอร์ม1 ไปยังฟอร์ม2 ควรมีการเตรียมตัวอย่างไรก่อน?

สร้างหน้าแบบฟอร์มทั้ง2 อันก่อนหลังจากนั้นให้สร้างฟังก์ชันเพื่อเชื่อมโยงไปยังอีกแบบฟอร์มหนึ่ง

- 3.2. ฟังก์ชันเรียกตัวเองคืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

3.2. ฟังก์ชันเรียกตัวเองคืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

การที่function เรียกใช้ตัวมันเองวนไปเรื่อยๆจนกว่าจะถึงbreak case คือหยุดเรียกfunction โดยมันเป็นรูปแบบการloop รูปแบบหนึ่ง

```
Int factorial(n){
```

```
    If(n == 1)
```

```
        Return 1;
```

```
    else
```

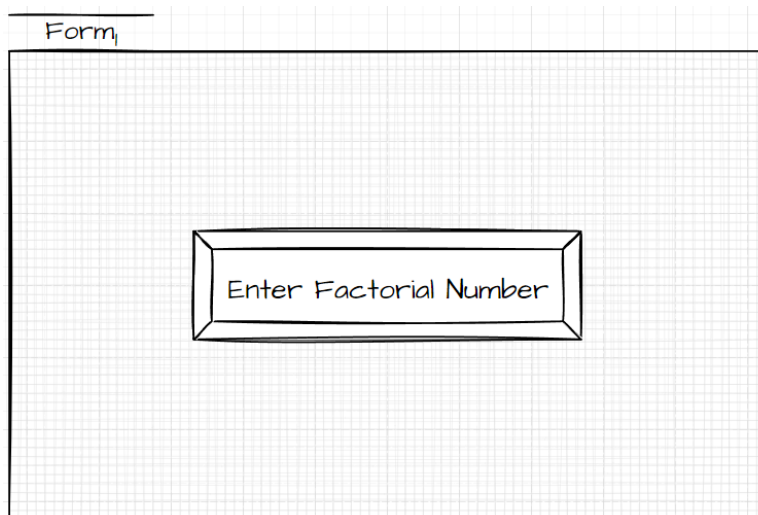
```
        X(factorial(n-1))
```

```
    }
```

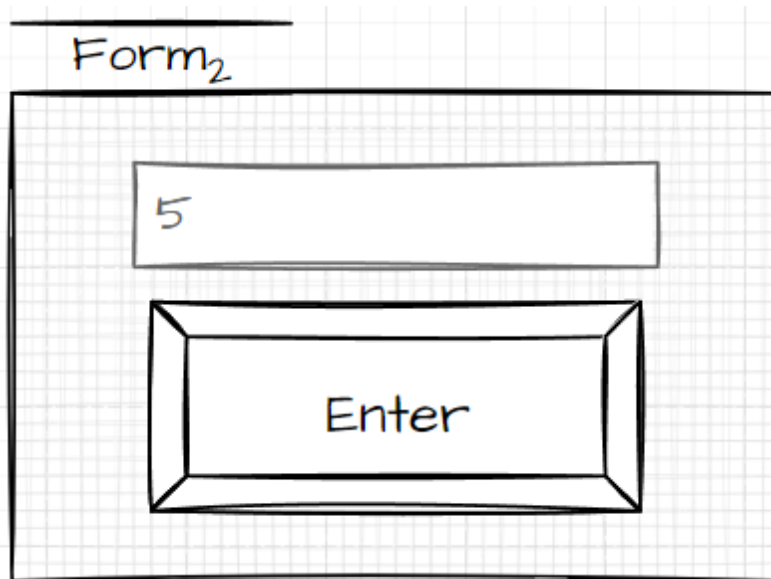
#### 4. ลำดับขั้นตอนการปฏิบัติการ

- 4.1. จงสร้าง Window Builder ในโปรแกรม Eclipse เพื่อสร้างโปรแกรมจำลองการทำงานเพื่อหาค่าของ Factorial ผ่านแบบจำลองแบบ Recursion บนโครงสร้างข้อมูลแบบ Stack โดยโปรแกรมจะมีการทำงานอยู่ 2 ฟอร์ม และมีลักษณะการทำงานดังต่อไปนี้
  - 4.1.1. ฟอร์ม 1 โดยจะมีปุ่มเพื่อให้ผู้ใช้กด และเรียกหน้าต่าง ฟอร์ม 2 ขึ้นมา

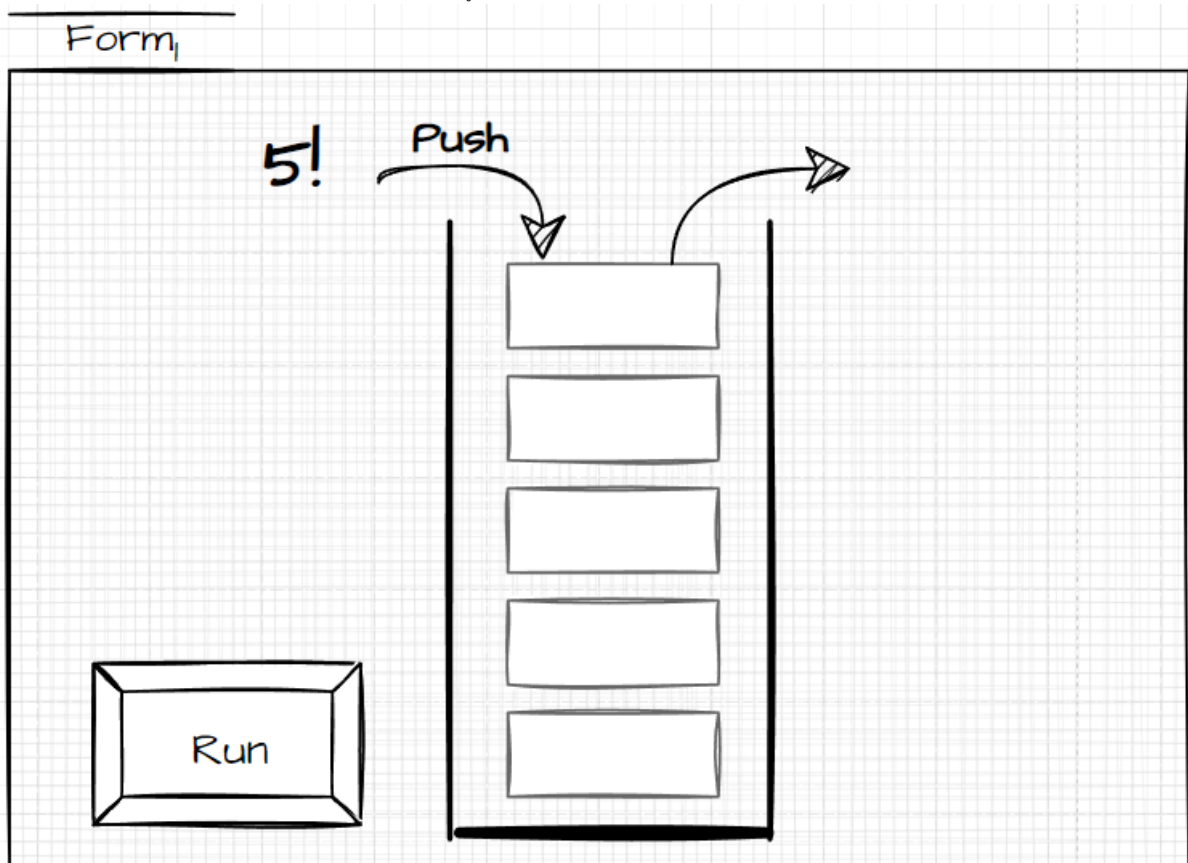
Form<sub>1</sub>



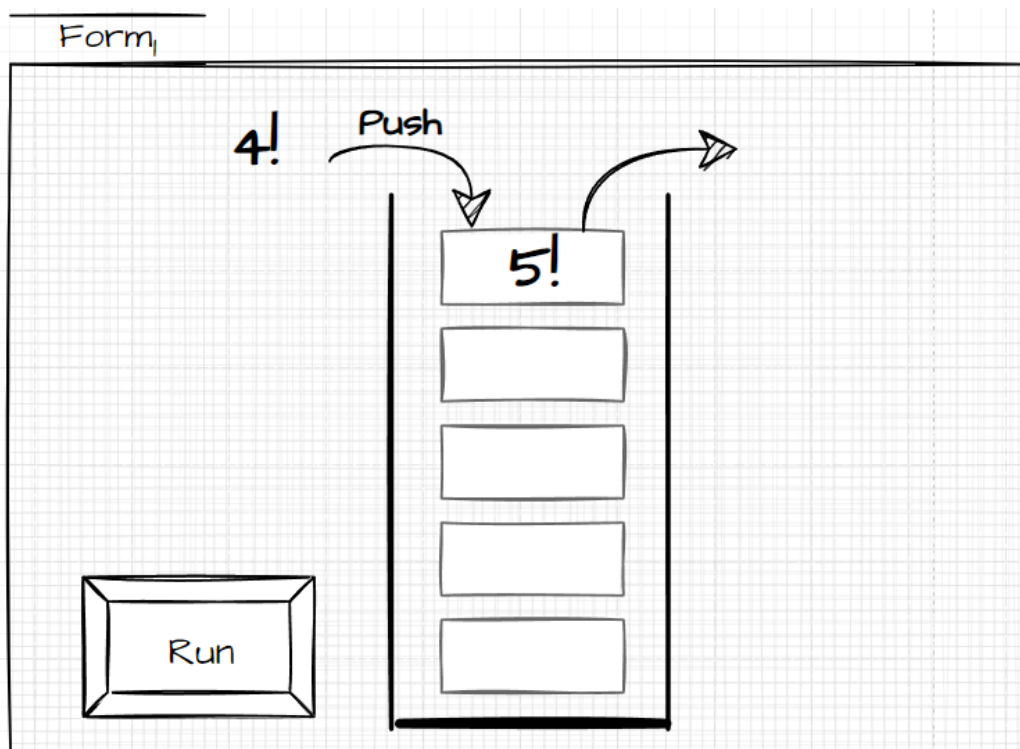
- 4.1.2. ฟอร์ม 2 เป็นหน้าต่างใหม่ที่เตรียมให้ผู้ใช้กรอกเลขที่ต้องการหาค่า Factorial ลงไปในช่อง Textbox โดยที่ผู้ใช้จะถูกจำกัดให้กรอกได้เฉพาะเลข 1 ถึง 5 เท่านั้น



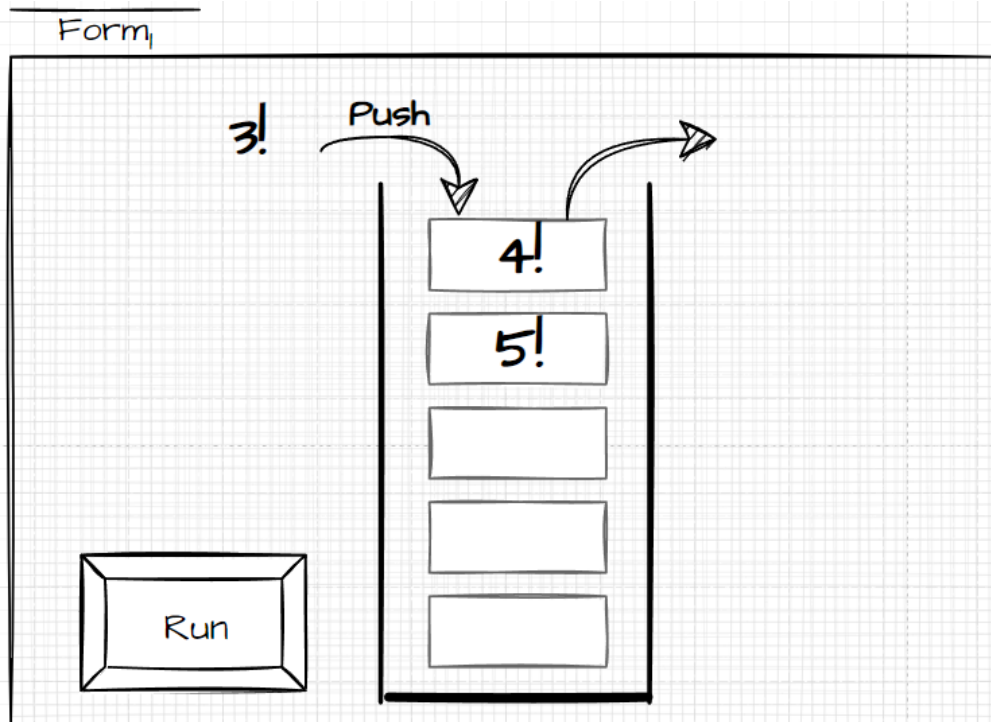
- 4.1.3. เมื่อกรอกข้อมูลในฟอร์ม 2 เสร็จแล้ว และกดปุ่ม Enter โปรแกรมจะนำเลข 5 ที่ได้จากช่อง Textbox ในฟอร์ม 2 ส่งค่ากลับไปยังฟอร์ม 1 อีกครั้ง และแสดงตัวเลขนั้นในช่องก่อนนำข้อมูล Push เข้าไปใน Stack เมื่อกดปุ่ม Run ทางด้านซ้ายล่าง ให้โปรแกรมทำการ Push ข้อมูล 5! เข้าไปใน Stack



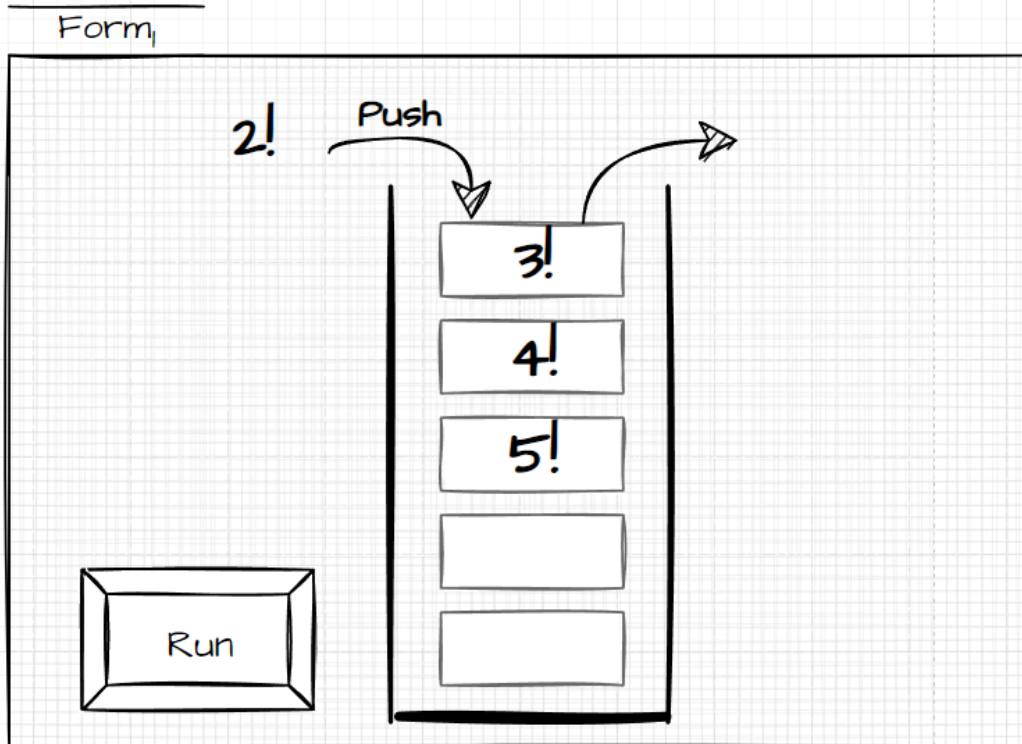
- 4.1.4. หลังจากกดปุ่ม Run เลข 5! จะเข้าไปอยู่ภายใน Stack และจะมีเลข 4! ที่รออยู่ในตำแหน่งรอ Push เข้าไปใน Stack ดังนั้น หากด้านบนสุดของ Stack ยังไม่ใช่เลข 1! เมื่อกดปุ่ม Run ระบบก็จะค่อยๆ นำข้อมูลเข้าไปใน Stack เรื่อยๆ



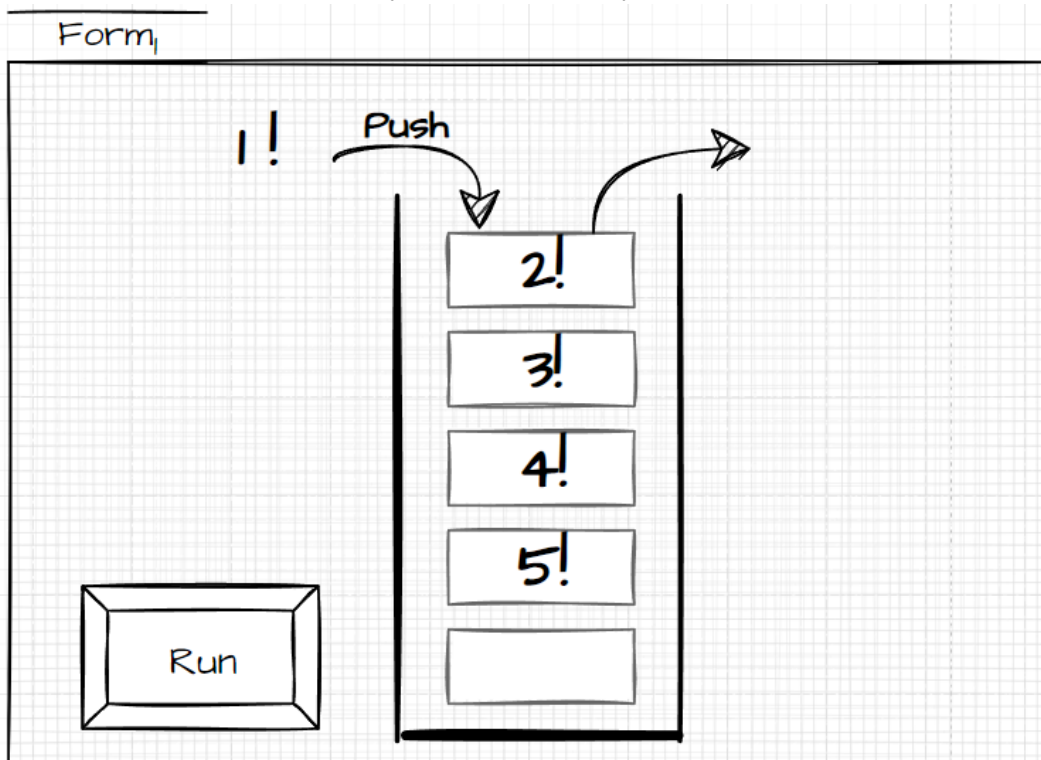
4.1.5. เช่นเดียวกันกับกรณีเมื่อครู่ หลังกดปุ่ม Run เลข 4! ก็จะถูก Push เข้าไปใน Stack ในตำแหน่งด้านบนสุด



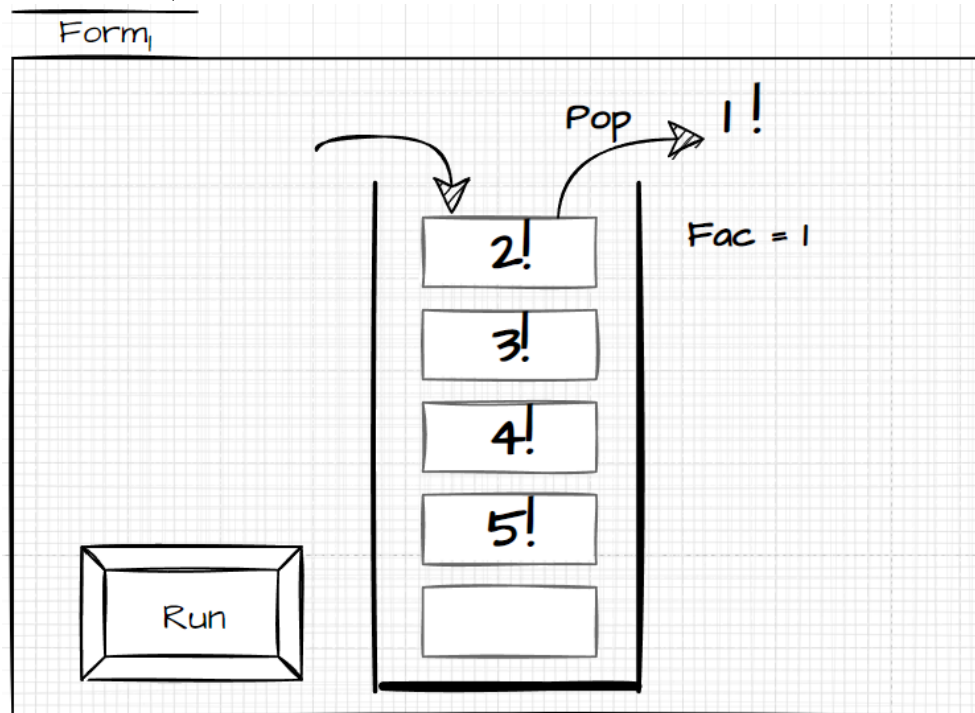
4.1.6. เช่นเดียวกันกับกรณีเมื่อครู! หลังกดปุ่ม Run เลข 3! ก็จะถูก Push เข้าไปใน Stack ในตำแหน่งด้านบนสุด



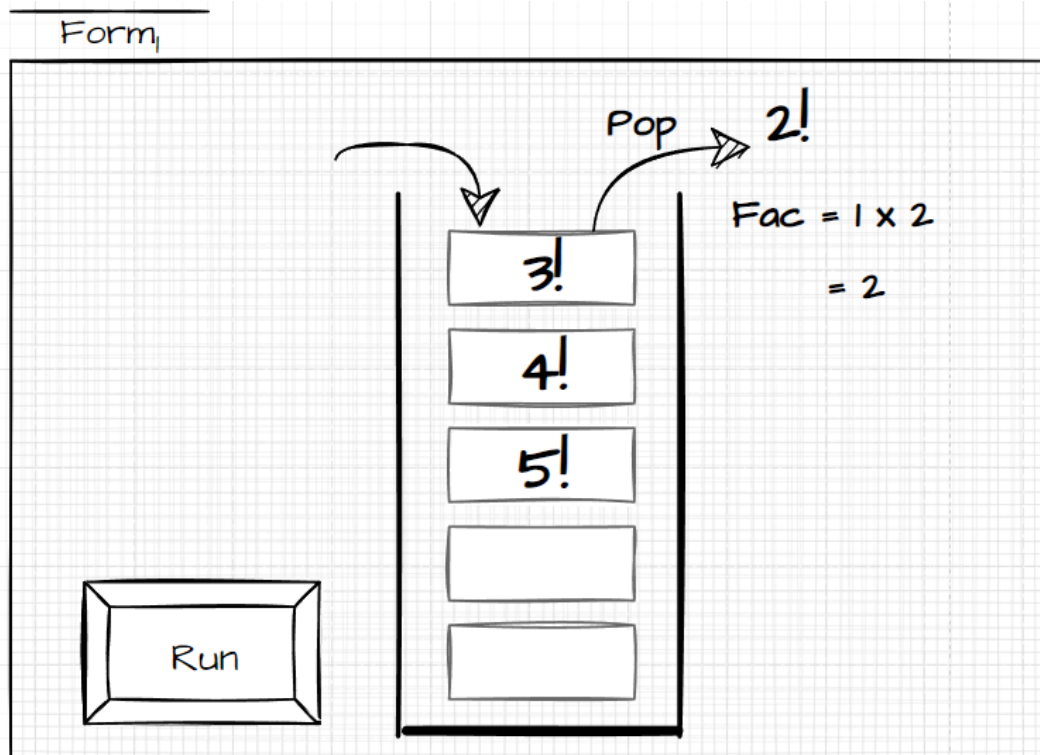
4.1.7. เช่นเดียวกันกับกรณีเมื่อครู! หลังกดปุ่ม Run เลข 2! ก็จะถูก Push เข้าไปใน Stack ในตำแหน่งด้านบนสุด



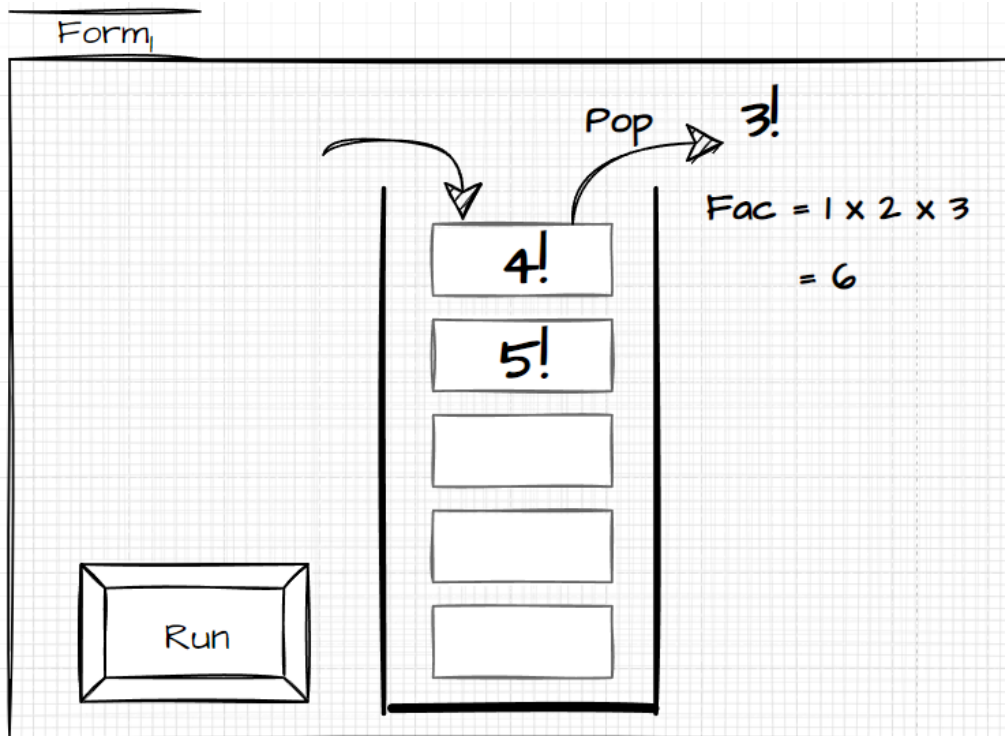
- 4.1.8. ที่นี้หลังจาก Push เลข 1! เข้าไปในระบบ จากกฎที่ว่า  $1! = 1$  ดังนั้นทำให้เราสามารถหาคำตอบของ 1! ได้ และเมื่อได้คำตอบให้ทำการ Pop เลข 1! ออกมา และใส่ไว้ในตัวแปร Fac พร้อมทั้งแสดงออกมาผ่านทาง Label เพื่อให้ผู้ใช้เห็นผลการคูณของชุดตัวเลข



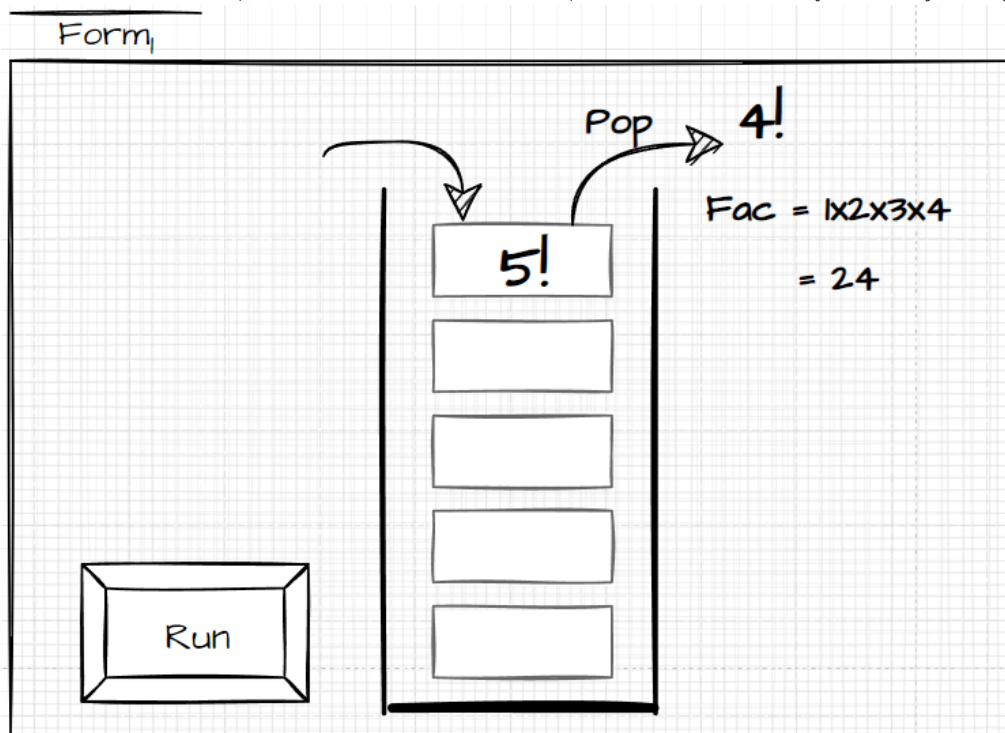
- 4.1.9. เมื่อกดปุ่ม Run อีกรอบ ระบบก็จะ Pop ตัวเลขบนสุดของ Stack ออกมา แล้วนำไปคูณค่า Fac ให้ผู้ใช้เห็นดังรูป



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

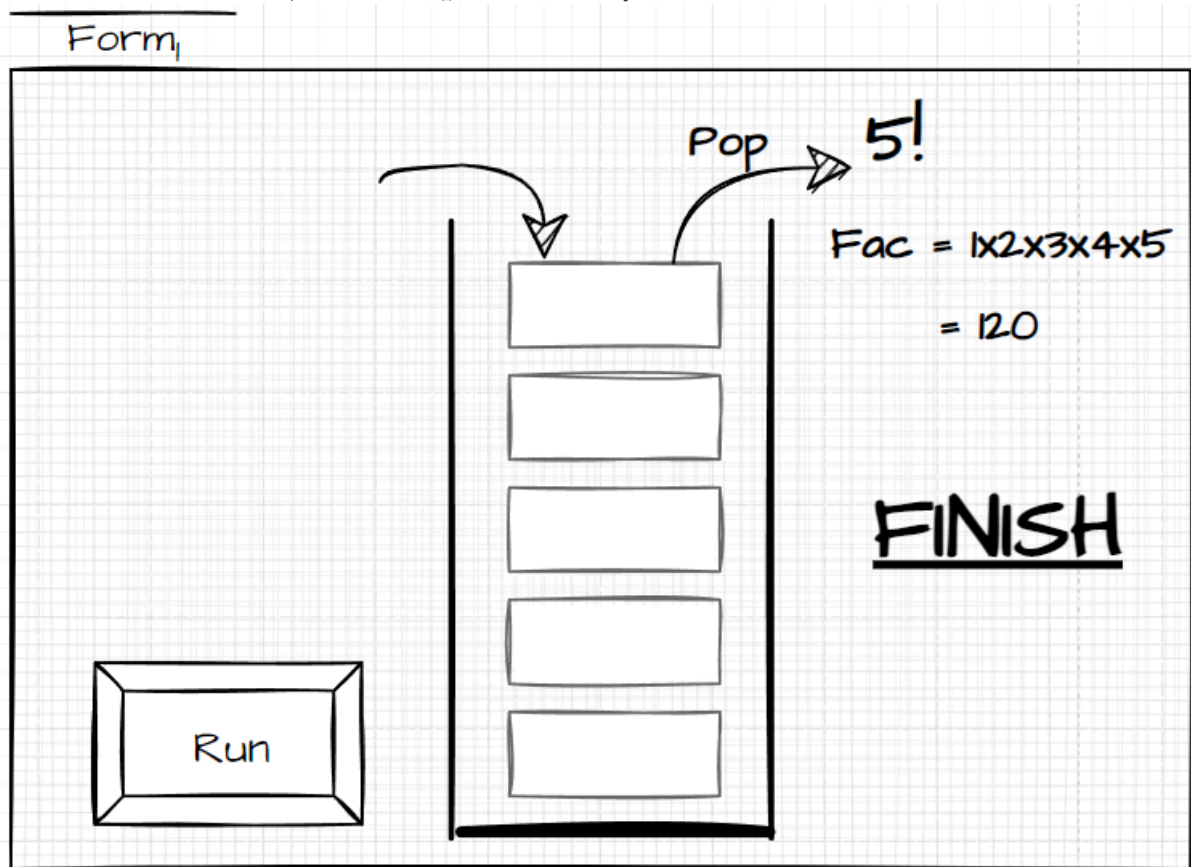


4.1.11. เมื่อกดปุ่ม Run อีกรอบ ระบบก็จะ Pop ตัวเลขบนสุดของ Stack ออกมา แล้วนำไปคูณค่า Fac ให้ผู้ใช้เห็นดังรูป





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



- 4.2. จงเขียนโค้ดโปรแกรมที่อยู่ภายในปุ่ม Run

โค้ดโปรแกรมภายในปุ่ม Run

```
Button btnRun = new Button(fm1, SWT.NONE);
btnRun.addSelectionListener(new SelectionAdapter() {
    @Override
    public void widgetSelected(SelectionEvent e) {
        if(backward) {
            if(i > 1) {
                lbPush.setText((i-1) + "! Push");
            } else {
                lbPush.setText("");
            }
            lbPop.setText("");
        } else {
            lbPush.setText("");
            lbPop.setText("Pop " + (i-1) + "!");
            switch(i-1) {
                case 1:
                    facAns.setText("Fac\t= 1\n\t= " + (ans = fac(i-1)));
                    break;
                case 2:
                    facAns.setText("Fac\t= 1x2\n\t= " + (ans = fac(i-1)));
```

```

        break;
    case 3:
        facAns.setText("Fac\t= 1x2x3\n\t= " + (ans = fac(i-1)));
        break;
    case 4:
        facAns.setText("Fac\t= 1x2x3x4\n\t= " + (ans = fac(i-1)));
        break;
    case 5:
        facAns.setText("Fac\t= 1x2x3x4x5\n\t= " + (ans = fac(i-1)));
        break;
    }
}

```

```

btnRun.setBounds(54, 213, 75, 25);
btnRun.setText("Run");

```

```

label_line1.setVisible(false);
label_line2.setVisible(false);
label_line3.setVisible(false);
lbStack1.setVisible(false);
lbStack2.setVisible(false);
lbStack3.setVisible(false);
lbStack4.setVisible(false);
lbStack5.setVisible(false);
lbPush.setVisible(false);
lbPop.setVisible(false);
facAns.setVisible(false);
btnRun.setVisible(false);

```

## 5. สรุปผลการปฏิบัติการ

สามารถสร้าง Window Builder ในโปรแกรม Eclipse และสร้างโปรแกรมจำลองการทำงานเพื่อหาค่าของ Factorial ผ่านแบบจำลองแบบ Recursion บนโครงสร้างข้อมูลแบบ Stack โดยโปรแกรมจะมีการทำงานอยู่ 2 ฟอรัม ได้

## 6. คำถามท้ายการทดลอง



6.1. ฟังก์ชันการทำงานใน Stack ควรมีอะไรบ้าง?

Stack Pointer และ Stack Element

6.2. การคำนวณ Factorial มีสูตรอย่างไร ?

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

6.3. หลักการสร้าง Recursion คืออะไร?

คือวิธีการ (1) – เพียงแค่เพิ่มทีละตัว

$$f(n) = 1 + 2 + 3 + \dots + n$$

วิธีการ (2) – การเพิ่มแบบเรียกซ้ำ

$$f(n) = 1 \quad n=1$$

$$f(n) = n + f(n-1) \quad n>1$$

6.4. ข้อควรระวังในการส่งข้อมูล ลามปอรัมคือ อะไร ?

ควรใช้ฟังก์ชันหรือเมธอดที่เหมือนกัน