

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 ?

สร้างตัวแปรของทั้ง 2 ให้เรียบร้อยก่อนส่ง

- 3.2. ฟังก์ชันเรียกตัวเองคืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

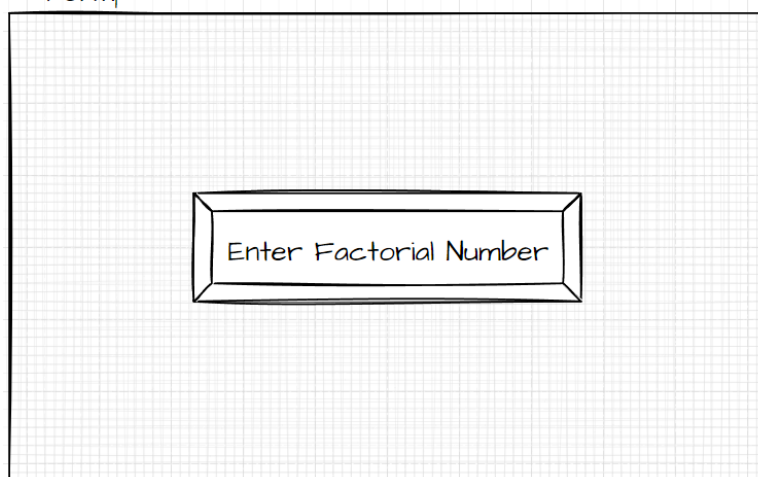
การที่มันเรียกใช้ตัวมันเองในฟังก์ชัน	<pre>int factorial(int N) int x, y ; if(N==0) { printf("\tRoll back point\n"); return(1); } else { x = N-1; printf("%2d! = %2d*%2d\n",N,N,x); y = factorial(x); printf("%2d! = %2d*%3d! = %5d\n",N ,N ,y ,y*N); return(N*y); } }</pre>
--------------------------------------	--

4. ลำดับขั้นการปฏิบัติการ

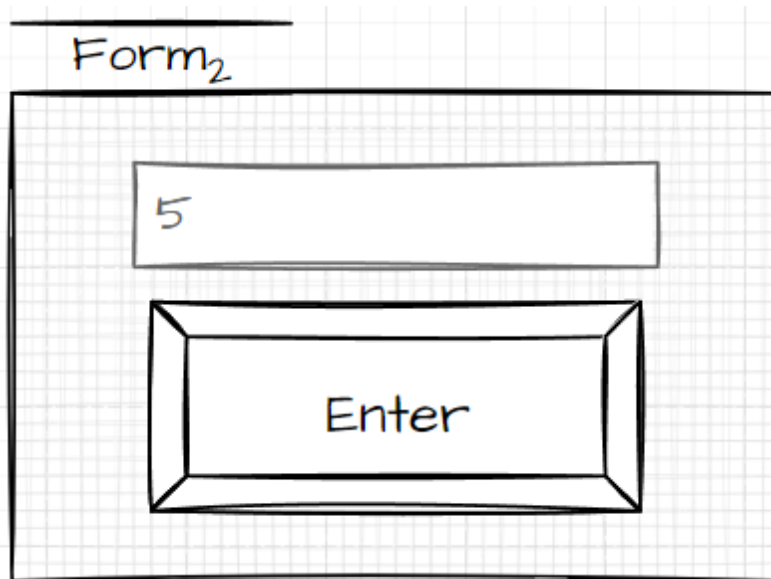
- 4.1. จงสร้าง Window Builder ในโปรแกรม Eclipse เพื่อสร้างโปรแกรมจำลองการทำงานเพื่อหาค่าของ Factorial ผ่านแบบจำลองแบบ Recursion บนโครงสร้างข้อมูลแบบ Stack โดยโปรแกรมจะมีการทำงานอยู่ 2 ฟอร์ม และมีลักษณะการทำงานดังต่อไปนี้

- 4.1.1. ฟอร์ม 1 โดยจะมีปุ่มเพื่อให้ผู้ใช้กด และเรียกหน้าต่าง ฟอร์ม 2 ขึ้นมา

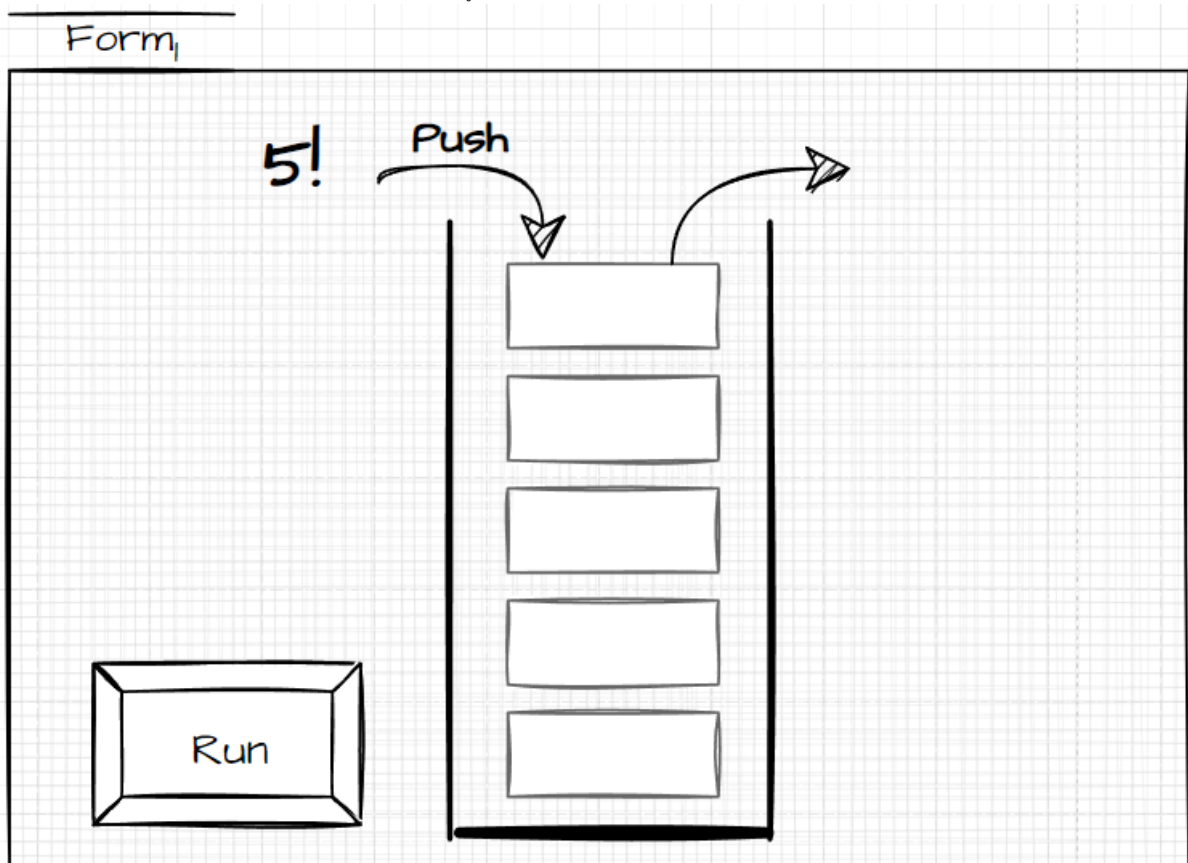
Form₁



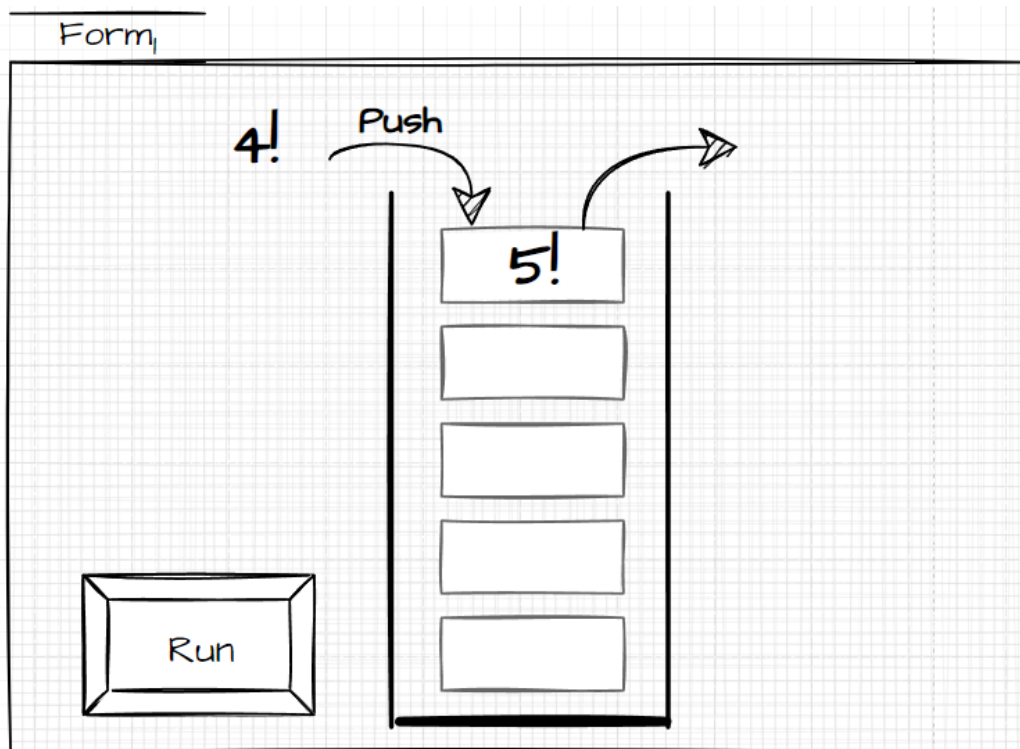
- 4.1.2. ฟอร์ม 2 เป็นหน้าต่างใหม่ที่เตรียมให้ผู้ใช้กรอกเลขที่ต้องการหาค่า Factorial ลงไปในช่อง Textbox โดยที่ผู้ใช้จะถูกจำกัดให้กรอกได้เฉพาะเลข 1 ถึง 5 เท่านั้น



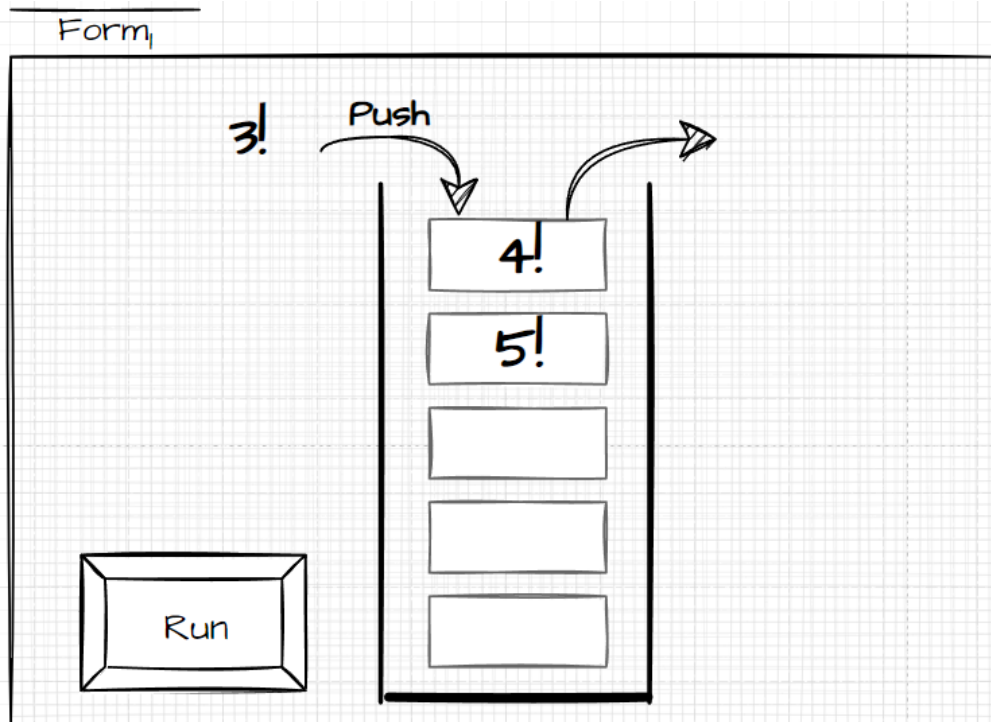
- 4.1.3. เมื่อกรอกข้อมูลในฟอร์ม 2 เสร็จแล้ว และกดปุ่ม Enter โปรแกรมจะนำเลข 5 ที่ได้จากช่อง Textbox ในฟอร์ม 2 ส่งค่ากลับไปยังฟอร์ม 1 อีกครั้ง และแสดงตัวเลขนั้นในช่องก่อนนำข้อมูล Push เข้าไปใน Stack เมื่อกดปุ่ม Run ทางด้านซ้ายล่าง ให้โปรแกรมทำการ Push ข้อมูล 5! เข้าไปใน Stack



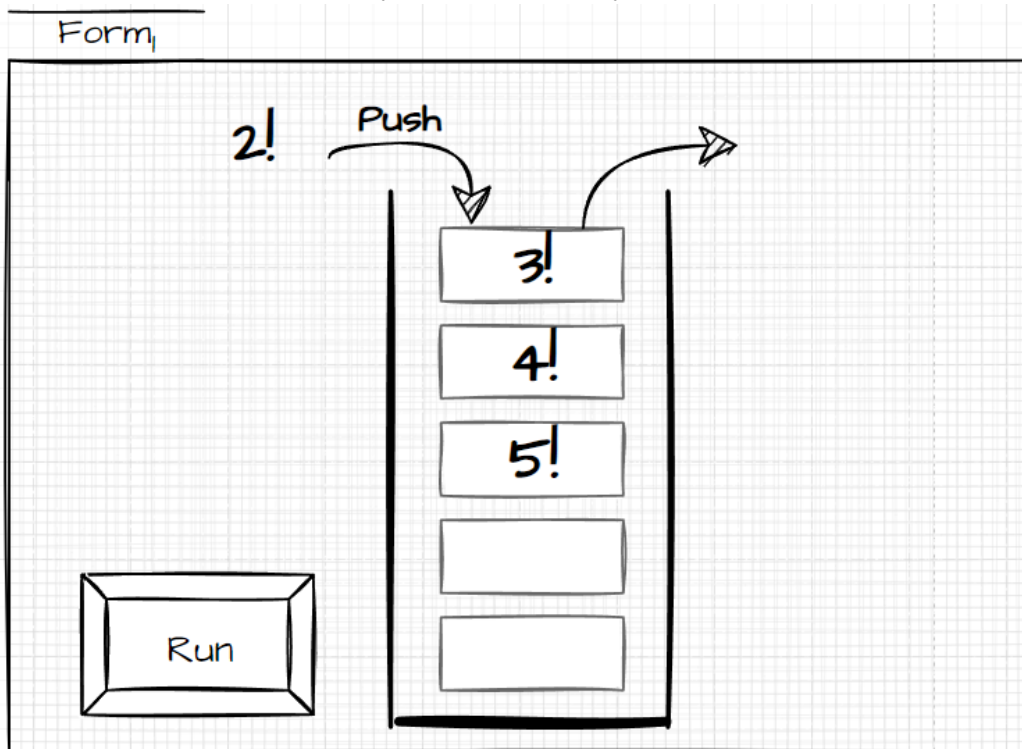
- 4.1.4. หลังจากกดปุ่ม Run เลข 5! จะเข้าไปอยู่ภายใน Stack และจะมีเลข 4! ที่รออยู่ในตำแหน่งรอ Push เข้าไปใน Stack ดังนั้นหากด้านบนสุดของ Stack ยังไม่ใช่เลข 1! เมื่อกดปุ่ม Run ระบบก็จะค่อยๆ นำข้อมูลเข้าไปใน Stack เรื่อยๆ



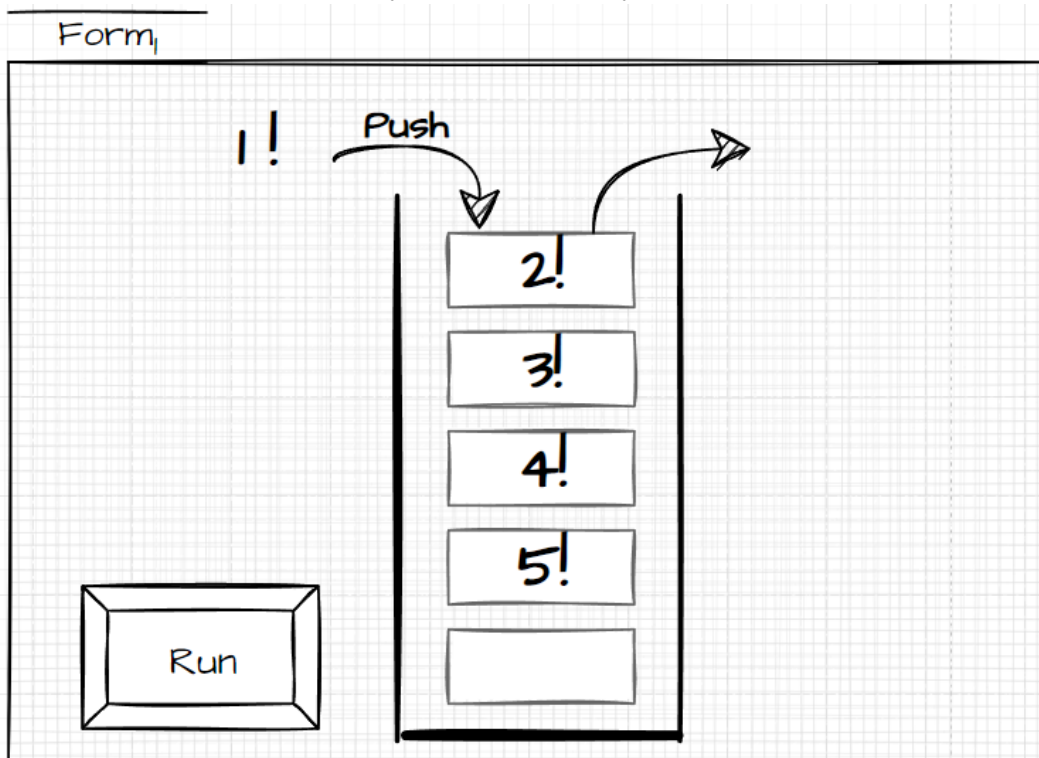
4.1.5. เช่นเดียวกันกับกรณีเมื่อครู่ หลังกดปุ่ม Run เลข 4! ก็จะถูก Push เข้าไปใน Stack ในตำแหน่งด้านบนสุด



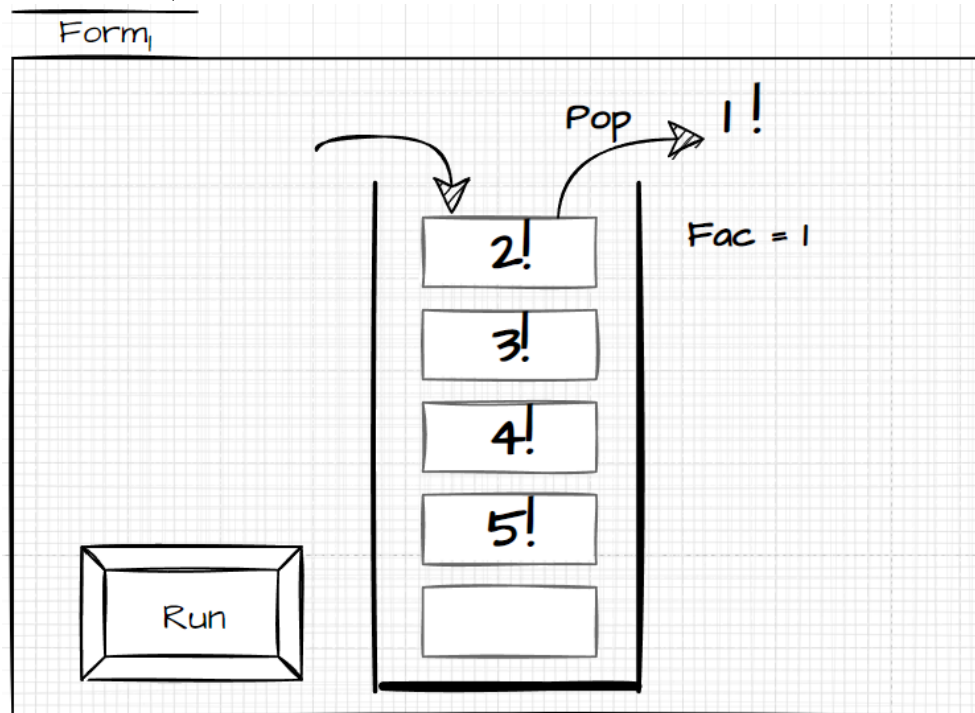
4.1.6. เช่นเดียวกันกับกรณีเมื่อครู! หลังกดปุ่ม Run เลข 3! ก็จะถูก Push เข้าไปใน Stack ในตำแหน่งด้านบนสุด



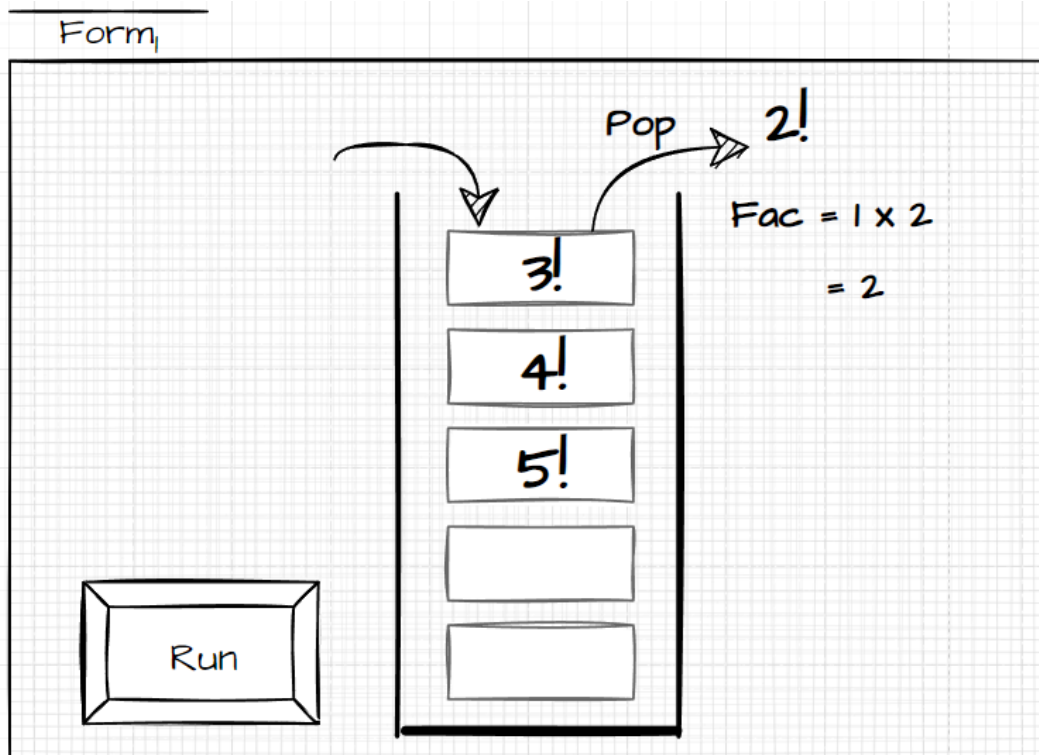
4.1.7. เช่นเดียวกันกับกรณีเมื่อครู! หลังกดปุ่ม Run เลข 2! ก็จะถูก Push เข้าไปใน Stack ในตำแหน่งด้านบนสุด



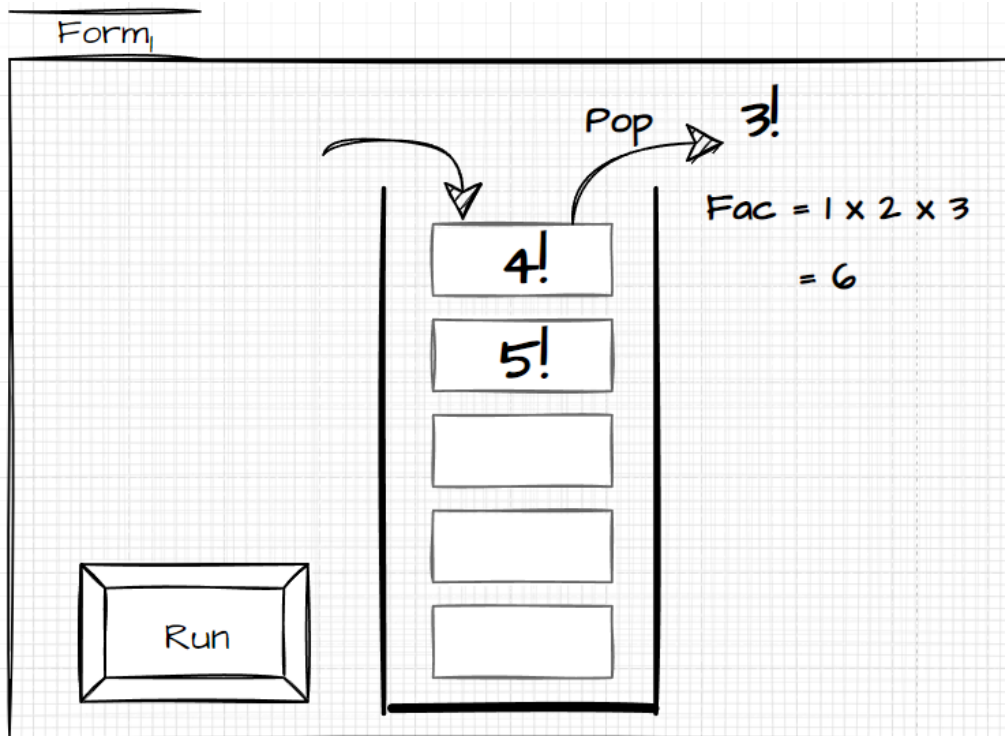
- 4.1.8. ที่นี้หลังจาก Push เลข 1! เข้าไปในระบบ จากกฎที่ว่า $1! = 1$ ดังนั้นทำให้เราสามารถหาคำตอบของ 1! ได้ และเมื่อได้คำตอบให้ทำการ Pop เลข 1! ออกมา และใส่ไว้ในตัวแปร Fac พร้อมทั้งแสดงออกมาผ่านทาง Label เพื่อให้ผู้ใช้เห็นผลการคูณของชุดตัวเลข



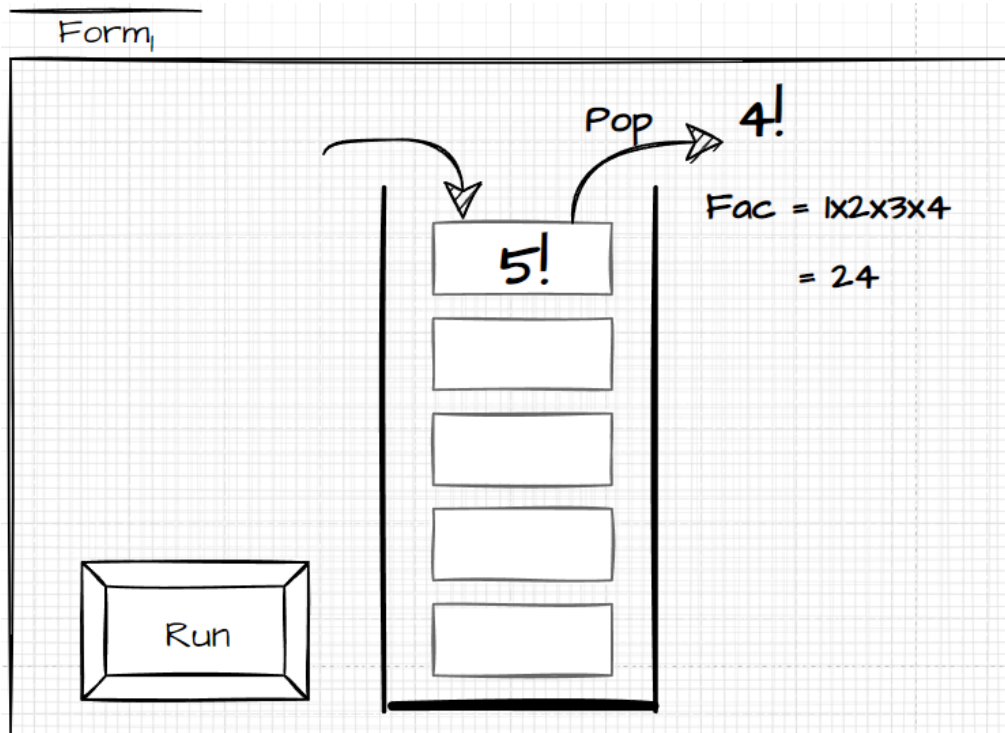
- 4.1.9. เมื่อกดปุ่ม Run อีกรอบ ระบบก็จะ Pop ตัวเลขบนสุดของ Stack ออกมา แล้วนำไปคูณค่า Fac ให้ผู้ใช้เห็นดังรูป



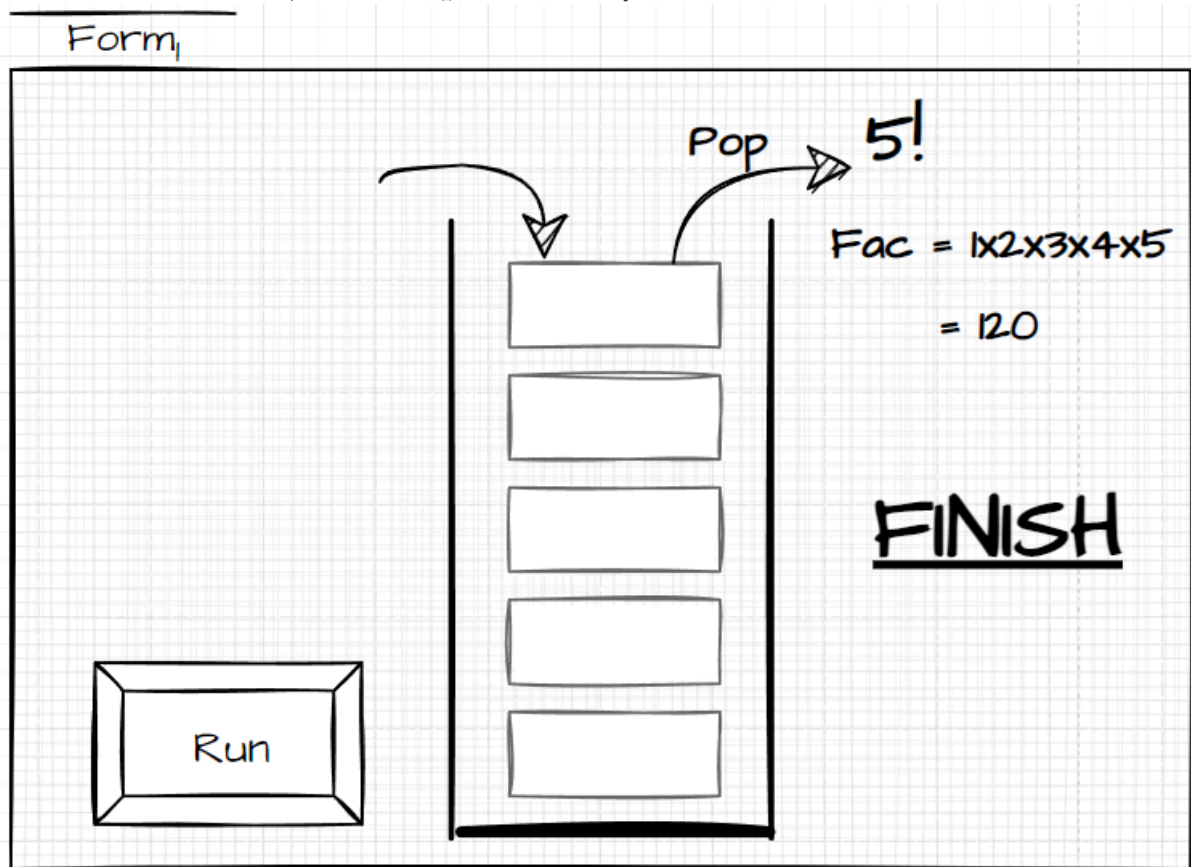
4.1.10. เมื่อกดปุ่ม Run อีกรอบ ระบบก็จะ Pop ตัวเลขบนสุดของ Stack ออกมา แล้วนำไปคูณค่า Fac ให้ผู้ใช้เห็นดังรูป



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

```
110 Button btnRun = new Button(fm1, SWT.NONE);
111 btnRun.addSelectionListener(new SelectionAdapter() {
112     @Override
113     public void widgetSelected(SelectionEvent e) {
114         if(backward) {
115             if(i > 1) {
116                 lbPush.setText((i-1) + "! Push");
117             } else {
118                 lbPush.setText("");
119             }
120             lbPop.setText("");
121         } else {
122             lbPush.setText("");
123             lbPop.setText("Pop " + (i-1) + "!");
124             switch(i-1) {
125                 case 1:
126                     facAns.setText("Fac\t= 1\n\t= " + (ans = fac(i-1)));
127                     break;
128                 case 2:
129                     facAns.setText("Fac\t= 1x2\n\t= " + (ans = fac(i-1)));
130                     break;
131                 case 3:
132                     facAns.setText("Fac\t= 1x2x3\n\t= " + (ans = fac(i-1)));
133                     break;
134                 case 4:
135                     facAns.setText("Fac\t= 1x2x3x4\n\t= " + (ans = fac(i-1)));
136                     break;
137                 case 5:
138                     facAns.setText("Fac\t= 1x2x3x4x5\n\t= " + (ans = fac(i-1)));
139                     break;
140             }
141         }
142     }
143     switch(num) {
144         case 1:
```

```

145     switch(i) {
146     case 1:
147         lbStack5.setText("1");
148         lbStack4.setText("");
149         lbStack3.setText("");
150         lbStack2.setText("");
151         lbStack1.setText("");
152         break;
153     case 2:
154         lbStack5.setText("");
155         lbStack4.setText("");
156         lbStack3.setText("");
157         lbStack2.setText("");
158         lbStack1.setText("");
159         break;
160     case 3:
161         fm1.close();
162         break;
163     }
164     break;
165 case 2:
166     switch(i) {
167     case 1:
168         lbStack5.setText("1");
169         lbStack4.setText("2");
170         lbStack3.setText("");
171         lbStack2.setText("");
172         lbStack1.setText("");
173         break;
174     case 2:
175         lbStack5.setText("2");
176         lbStack4.setText("");
177         lbStack3.setText("");
178         lbStack2.setText("");
179         lbStack1.setText("");
180         break;
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

5. สรุปผลการปฏิบัติการ

ส่วนใหญ่โปรแกรมจะใช้คำสั่ง switch เพื่อให้ทำตามที่ได้ มีการติดปัญหาเรื่องการกดปุ่ม

แล้ววันที่เดียวหมดเลยไม่สามารถกดให้มันออกมาที่เลขได้

6. คำถามท้ายการทดลอง

6.1. ฟังก์ชันการทำงานใน Stack ควรมีอะไรบ้าง?

1. ตัวชี้สแต็ก หรือ Stack Pointer

2. ส่วนสมาชิกของสแต็ก

6.2. การคำนวณ Factorial มีสูตรว่าอย่างไร ?

...n, n-1, n-2, ..., 1

$9! = 9 \cdot 8 \cdot 7 \cdot 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 = 362,880$

6.3. หลักการสร้าง Recursion คืออะไร?

การเขียนฟังก์ชันมาตัวหนึ่ง ถ้ายังหาคำตอบไม่ได้ก็ให้เรียกตัวเองซ้ำไปเรื่อย ๆ จนกว่าจะเจอ

คำตอบ

6.4. ข้อควรระวังในการส่งข้อมูลข้ามฟอร์มคืออะไร ?

ต้องเช็คดูก่อนว่าสร้างตัวแปรไว้พร้อมทั้ง 2 ฟังก์ชันแล้ว แล้วก็เช็คการทำงานในโปรแกรมที่ทำการ

เคลื่อนย้ายข้อมูลด้วย