**Assignment 1 Recursion Factorial Program**

Source code

import java.util.Scanner;

public class Assignment1 {

public static void main(String[] args) {

//Introduction program

System.out.println("My Recursion Program.");

System.out.println("Program calculate n! by recursion (n <= 15)");

Run('y');

//End program

System.out.println("End Program.");

System.out.println("Program written by 62070501022 Thirathawat Chansarikorn");

}

private static char Run(char check\_end) {

//Declare variable

int n;

long ans;

if (check\_end != 'y') {

//base case --> end function (end program)

return check\_end;

}

else {

//recursive case

n = Read\_Int(0 , 15);

ans = Factorial(n);

System.out.printf("Complete calculate of %d!, answer = %d\n", n, ans);

System.out.print("press [y] to continue, others to exit.");

Scanner in = new Scanner(System.in);

try {

check\_end = in.next().charAt(0);

}

catch (Exception e) {

System.out.println(e.getMessage());

in.nextLine();

}

return Run(check\_end);

}

}

private static long Factorial(int n) {

//Declare variable

long ans, factorial;

if (n == 0) {

//base case

System.out.println("0! is base case return answer of 0! = 1");

System.out.println("Calculate 0! complete.");

return 1;

}

else {

//recursive case

System.out.printf("%d! is recursive case ", n);

System.out.printf("Answer = %d \* recursive of %d!\n", n, n - 1);

System.out.printf("\tRecursion to calculate %d!\n", n - 1);

factorial = Factorial(n - 1);

if (n > 1)

System.out.printf("Calculate %d! complete.\n", n - 1);

System.out.printf("\tReturn answer from %d! = %d ", n - 1, factorial);

ans = n \* factorial;//ans of n!

System.out.printf("to calculate %d! = [%d \* %d!] = %d \* %d = %d\n"

, n, n, n - 1, n, factorial, ans);

return ans;

}

}

private static int Read\_Int(int min, int max) {

//declare local variable

int input;

//get values from keyboard

Scanner in = new Scanner(System.in);

try {

System.out.print("Enter n : ");

input = in.nextInt();

if (input >= min && input <= max)

return input;

else

System.out.printf("Input error, please enter between %d - %d\n", min, max);

}

catch (Exception e) {

System.out.printf("Input error, please enter between %d - %d\n", min, max);

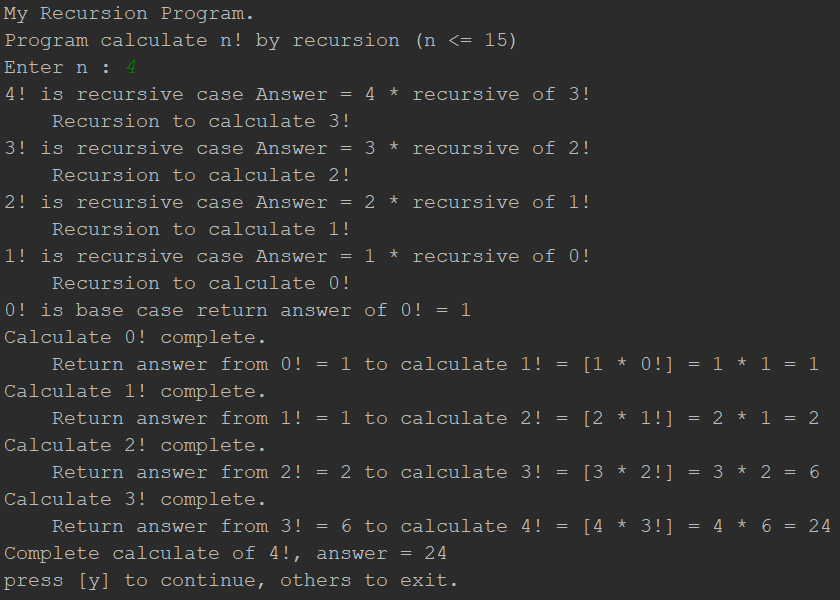
in.nextLine();

}

return Read\_Int(min, max);

}

}

**Test case1: Input = 4**

**อธิบาย:**

ลักษณะการทำงานแบ่งเป็น 2 กรณี

1. Base case เป็นกรณีที่ค่าที่รับเข้ามาเป็น 0 เนื่องจาก 0! = 1 ดังนั้นจึง Return 1 ออกจากฟังก์ชัน
2. Recursive case เป็นกรณีที่ค่าที่รับเข้ามาไม่เป็น 0 การที่จะหาค่า n! ได้ต้องรู้ค่า

(n - 1)! เนื่องจาก n! = n x (n – 1)! จึงต้องมีการเรียกใช้ฟังกันซ้ำเพื่อหาค่า (n – 1)!

เมื่อได้ค่ามาแล้วจึงคำนวณค่า n! x (n – 1)! แล้ว Return คำตอบ

ใน Test case นี้ 4! เป็น Recursive case จะหาค่าได้ต้องรู้ค่าของ 3! 2! 1! และ 0! การทำงานเป็นไปตามกรอบด้านล่าง

Factorial(4) = 4 \* Factorial(3) //status of Factorial(4)

= 4 \* (3 \* Factorial(2)) //status of Factorial(3)

= 4 \* (3 \* (2 \* Factorial(1))) //status of Factorial(2)

= 4 \* (3 \* (2 \* (1 \* Factorial(0)))) //status of Factorial(1)

= status of Factorial(0) return 1 //base case

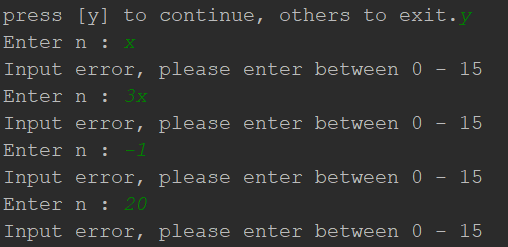
= 4 \* (3 \* (2 \* (1 \* 1))) //status of Factorial(1)

= 4 \* (3 \* (2 \* 1)) //status of Factorial(2)

= 4 \* (3 \* 2) //status of Factorial(3)

= 4 \* 6 = 24 //status of Factorial(4)

**Test case2: Input = x, 3x, -1, 20**

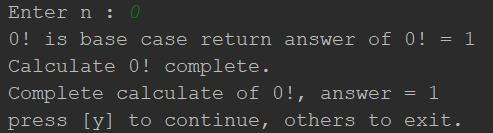


**อธิบาย:**

เนื่องจากโปรแกรมถูกกำหนดไว้ว่าจะคำนวณค่า 0! ถึง 15! จึงต้องมีการจัดการกับข้อมูลที่รับเข้ามาโดยแบ่งเป็นกรณี ดังนี้

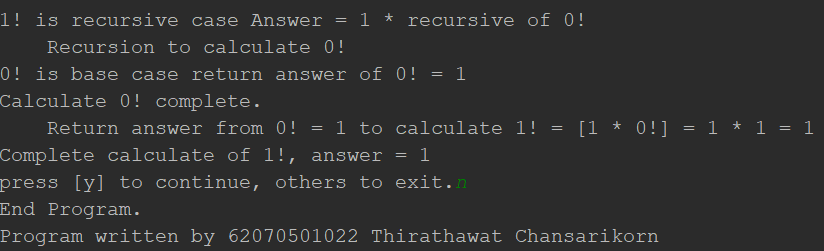
1. เป็นค่าจำนวนเต็มที่ไม่ได้อยู่ระหว่าง 0 – 15 (ดักจับด้วยเงื่อนไข if-else)
2. ค่าที่กรอกเข้ามาไม่ใช่ตัวเลขจำนวนเต็ม (ดักจับ Exception)

**Test case3: Input = 0**



**อธิบาย:** 0 เป็น Base case ฟังก์ชันจึง Return 1 เป็นคำตอบ

**Test case3: Input = 1**



**อธิบาย:** 1 เป็น Recursive case ต้องรู้ค่า 0! เพื่อใช้ในการคำนวณซึ่งการทำงานเป็นไปตามกรอบด้านล่าง

Factorial(1) = 1 \* Factorial(0) //status of Factorial(1)

= status of Factorial(0) return 1 //base case

= 1 \* 1 = 1 //status of Factorial(1)