

แผนการสอนปฏิบัติ 7

- ✚ **วัตถุประสงค์** สร้างตารางอาร์เรย์แสดงค่าตัวแปรต่างๆที่อยู่ในวงรอบ
- ✚ **ผลลัพธ์การเรียนรู้** เข้าใจการทำงานวงรอบ และการสร้างเงื่อนไขการวนรอบ และโครงสร้างแบบอาร์เรย์
- ✚ **โจทย์ปัญหา** เขียนโปรแกรมวนรอบ สร้างตารางเพื่อแสดงค่าตัวแปรในแต่ละรอบ และสรุปคำตอบ
 1. แสดงตารางค่าตัวแปร และผลรวม **n** เทอมแรกของ $\frac{2}{1} + \frac{2+4}{1*2} + \frac{2+4+6}{1*2*3} + \frac{2+4+6+8}{1*2*3*4} + \dots$
ทดสอบที่ **n** = 10 ได้คำตอบเป็น 8.154842
 2. แสดงตารางค่า fibonacci จำนวน **n** เทอม เฉพาะคำตอบที่เป็นเลขคู่ > 0 (**n** และ **fn**)
ทดสอบที่ **n** = 10 ได้คำตอบเป็น f3 = 2, f6=8, f9=34, f12=144, ..., f30 = 832040
 3. แสดงตารางของจำนวนเฉพาะที่อยู่ระหว่าง **min** ถึง **max** แล้วสรุป จำนวนเทอม และ ผลบวกของค่าในตาราง
ทดสอบที่ min = 1 , max = 30 ได้คำตอบเป็น count =10 , sum = 126 // 2, 3, 5, 7, 11, 13, 17, 19, 23, 29
 4. แสดงตารางลำดับ ค่า และผลรวมของอนุกรม (1×40)+(3×38)+(5×36)+ ... + (39×2) ตั้งแต่เทอมแรกจนกระทั่งผลรวมไม่เกินค่า **max** และสรุปจำนวนเทอม และผลรวมต่ำสุดก่อนถึงค่า **max**
ทดสอบที่ max = 5000 ได้คำตอบ n =14, sum=4730
 5. แสดงตารางค่าของพจน์ที่เกิดขึ้นใน **n** พจน์แรกของลำดับเลขคณิต 1, 4, 7, 10, 13, 16 ... ที่มีค่าซ้ำอยู่ในลำดับ 1, 6, 11, 16, 21, 26, 31 ... (ต้องหาสูตรลำดับที่ทำให้เกิดค่าซ้ำรวมกันมาเปรียบเทียบ)
ทดสอบที่ n = 15 // 1, 16, 31, 46, 61, 76, 91, 106, 121, 136, 151, 166, 181, 196, 211
- ✚ **ขั้นตอน**
 - สร้างฟังก์ชันสำหรับเติมค่าลงในอาร์เรย์
 - สร้างฟังก์ชันสำหรับแสดงผลข้อมูลในอาร์เรย์
 - สร้างฟังก์ชันสำหรับแสดงเมนูและรีเทิร์นค่าเป็นตัวเลขที่ผู้เลือก
 - รับค่าทดสอบ (มีการป้องกันการป้อนผิด) วนรอบเรียกใช้ฟังก์ชัน เมื่อรับค่ากลับมาแล้วให้ใช้คำสั่ง printf เพื่อแสดงค่าตัวแปรที่เกี่ยวข้องในแต่ละวนรอบของฟังก์ชัน ในรูปแบบตาราง
 - แสดงค่าตัวแปรที่กำหนดหลังจากหลุดจากการวนรอบ
- ✚ **การส่งงาน**
 - โค้ดของโปรแกรมที่ปรับปรุงแก้ไขแล้ว พร้อมอธิบายคำสั่งโดยการเขียนคอมเมนต์แทรกลงในโค้ด
 - รายงาน ประกอบด้วยหน้าจอสรุปลผลการรันในกรณีต่างๆ

ตัวอย่างตารางคำตอบ

Test #1

```
*****
* i * fi * sum *
*****
* 1 * 2.000000 * 2.000000 *
* 2 * 3.000000 * 5.000000 *
* 3 * 2.000000 * 7.000000 *
* 4 * 0.833333 * 7.833333 *
* 5 * 0.250000 * 8.083333 *
* 6 * 0.058333 * 8.141667 *
* 7 * 0.011111 * 8.152778 *
* 8 * 0.001786 * 8.154563 *
* 9 * 0.000248 * 8.154812 *
* 10 * 0.000030 * 8.154842 *
*****
```

ans = 8.154842

Test #2

```
*****
*count* i * fi *
*****
* 1 * 3 * 2 *
* 2 * 6 * 8 *
* 3 * 9 * 34 *
* 4 * 12 * 144 *
* 5 * 15 * 610 *
* 6 * 18 * 2584 *
* 7 * 21 * 10946 *
* 8 * 24 * 46368 *
* 9 * 27 * 196418 *
* 10 * 30 * 832040 *
*****
```

count = 10, ans = 832040

Test #3

```
*****
* no * i * sum *
*****
* 1 * 2 * 2 *
* 2 * 3 * 5 *
* 3 * 5 * 10 *
* 4 * 7 * 17 *
* 5 * 11 * 28 *
* 6 * 13 * 41 *
* 7 * 17 * 58 *
* 8 * 19 * 77 *
* 9 * 23 * 100 *
* 10 * 29 * 129 *
*****
```

count = 10, sum = 129

Test #4

```
*****
* i * fi * sum *
*****
* 1 * 40 * 40 *
* 2 * 114 * 154 *
* 3 * 180 * 334 *
* 4 * 238 * 572 *
* 5 * 288 * 860 *
* 6 * 330 * 1190 *
* 7 * 364 * 1554 *
* 8 * 390 * 1944 *
* 9 * 408 * 2352 *
* 10 * 418 * 2770 *
* 11 * 420 * 3190 *
* 12 * 414 * 3604 *
* 13 * 400 * 4004 *
* 14 * 378 * 4382 *
* 15 * 348 * 4730 *
* 16 * 310 * 5040 *
*****
```

n = 16, ans = 5040

Test #5

```
*****
* no * term *
*****
* 1 * 1 *
* 2 * 16 *
* 3 * 31 *
* 4 * 46 *
* 5 * 61 *
* 6 * 76 *
* 7 * 91 *
* 8 * 106 *
* 9 * 121 *
* 10 * 136 *
* 11 * 151 *
* 12 * 166 *
* 13 * 181 *
* 14 * 196 *
* 15 * 211 *
* 16 * 226 *
* 17 * 241 *
* 18 * 256 *
* 19 * 271 *
* 20 * 286 *
*****
```

count = 20