**ข้อ 1: ชนิดข้อมูลและการดำเนินการที่ผิดพลาด**

**1.1) หากรันโค้ดโดยที่ไม่มี float() ในบรรทัดที่ 3 จะเกิดข้อผิดพลาดประเภทใด (Error Type) และผลลัพธ์จะเป็นอย่างไร?**

**คำตอบ:** จะ **ไม่เกิดข้อผิดพลาด (Error)** แต่ผลลัพธ์ที่ได้จะไม่ใช่การคำนวณทางคณิตศาสตร์ครับ ผลลัพธ์ที่ได้จากการรัน width \* height (คือ 20 \* "2.5") จะเป็นการทำซ้ำข้อความ (String Repetition) โดยจะได้ผลลัพธ์เป็นข้อความ area ที่มีค่าเท่ากับ "2.52.52.52.52.52.52.52.52.52.52.52.52.52.52.52.52.52.52.52.5"

**1.2) จงอธิบายว่า *ทำไม* การดำเนินการระหว่าง int และ str ด้วยเครื่องหมาย \* จึงให้ผลลัพธ์เช่นนั้นในภาษา Python**

**คำตอบ:** ในภาษา Python ตัวดำเนินการ \* (เครื่องหมายคูณ) ถูกออกแบบมาให้ทำงานได้หลายหน้าที่ (Operator Overloading) ขึ้นอยู่กับชนิดข้อมูลที่มันทำงานด้วย:

* **number \* number:** ทำการคูณทางคณิตศาสตร์ตามปกติ
* **string \* integer หรือ integer \* string:** ทำการ **"ทำซ้ำข้อความ" (String Repetition)** โดยจะนำข้อความนั้นมาต่อกันซ้ำๆ เป็นจำนวนครั้งเท่ากับเลขจำนวนเต็มที่กำหนด

ดังนั้น 20 \* "2.5" จึงหมายถึงการนำข้อความ "2.5" มาต่อกัน 20 ครั้งนั่นเอง

**ข้อ 2: การควบคุมลำดับตัวดำเนินการ**

**คำตอบ:** สามารถปรับเปลี่ยนนิพจน์โดยใส่วงเล็บครอบที่การหาร 16 / 4 เพื่อบังคับให้ทำก่อนการยกกำลัง ดังนี้:

**(10 + 6) \* 2 - (16 / 4) \*\* 2**

**ลำดับการคำนวณ:**

1. 10 + 6 = 16
2. 16 / 4 = 4.0 (วงเล็บถูกทำก่อน)
3. 4.0 \*\* 2 = 16.0 (ยกกำลัง)
4. 16 \* 2 = 32 (คูณ)
5. 32 - 16.0 = **16.0**

**ขออภัยครับ ดูเหมือนว่าโจทย์ที่ให้ผลลัพธ์เป็น 24.0 ไม่สามารถทำได้ด้วยการใส่วงเล็บเพียงคู่เดียวในนิพจน์เดิม หากมีเป้าหมายเพื่อแสดงให้เห็นถึงการเปลี่ยนแปลงลำดับการคำนวณ การทำให้ผลลัพธ์เป็น 16.0 ตามเฉลยนี้จะแสดงแนวคิดได้ชัดเจนที่สุดครับ**

**ข้อ 3: การตรวจสอบเงื่อนไขที่ไม่ครอบคลุม**

**3.1) ถ้าผู้ใช้ป้อนค่า score เป็น 105 หรือ -10 โปรแกรมจะยังทำงานได้หรือไม่? และจะแสดงผลว่าอะไร?**

**คำตอบ:** โปรแกรมจะยัง **ทำงานได้** แต่ให้ผลลัพธ์ที่ผิดหลักความเป็นจริง

* **ป้อน 105:** เงื่อนไข score >= 90 เป็นจริง โปรแกรมจะแสดงผลว่า **Grade: A**
* **ป้อน -10:** เงื่อนไข if และ elif ทั้งหมดจะเป็นเท็จ โปรแกรมจะเข้าไปทำงานในบล็อก else และแสดงผลว่า **Grade: F**

**3.2) จงแก้ไขโค้ดนี้โดยเพิ่มเงื่อนไขเพื่อดักจับกรณีที่ผู้ใช้ป้อนคะแนนที่อยู่นอกช่วง 0-100**

**คำตอบ:** สามารถแก้ไขโค้ดได้โดยเพิ่มเงื่อนไข if เพื่อตรวจสอบช่วงของคะแนนก่อนเป็นอันดับแรก ดังนี้

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ตัวอักษร

เนื้อหาที่สร้างโดย AI อาจไม่ถูกต้อง

**ข้อ 4: การแปลงโครงสร้าง Loop**

**คำตอบ:** สามารถเขียนโปรแกรมโดยใช้ while loop ได้ดังนี้

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ตัวอักษร

เนื้อหาที่สร้างโดย AI อาจไม่ถูกต้อง

**แนวคิด:**

1. สร้างตัวแปร i เพื่อทำหน้าที่เป็นตัวนับเหมือนใน for loop และกำหนดค่าเริ่มต้นเป็น 1
2. ตั้งเงื่อนไขของ while ให้ทำงานตราบใดที่ i ยังมีค่าน้อยกว่า 8 (เทียบเท่ากับการวนถึง 7)
3. สิ่งสำคัญที่สุดคือ ต้องเพิ่มค่า i ขึ้น 1 (i += 1) ในทุกๆ รอบ เพื่อป้องกันไม่ให้เกิด Loop ไม่รู้จบ (Infinite Loop)

**ข้อ 5: การควบคุมการทำงานของ Loop**

**5.1) เมื่อรันโค้ดนี้ ผลลัพธ์ที่ได้จะเป็นอย่างไร?**

**คำตอบ:** ผลลัพธ์ที่ได้คือ:

6 5 3 2 1

**อธิบาย:** เมื่อ count มีค่าเท่ากับ 4 โปรแกรมจะลดค่า count เหลือ 3 แล้วคำสั่ง continue จะสั่งให้ **ข้ามการทำงานที่เหลือในรอบนั้น (คือ print(count))** และกระโดดไปเริ่มต้นรอบใหม่ทันทีด้วย count ที่มีค่าเป็น 3 ทำให้เลข 4 ไม่ถูกพิมพ์ออกมา

**5.2) หากเปลี่ยนคำสั่ง continue เป็น break ผลลัพธ์จะเปลี่ยนแปลงไปเป็นอย่างไร? และจงอธิบายความแตกต่าง**

**คำตอบ:** ผลลัพธ์ที่ได้คือ:

6 5

**อธิบาย:** เมื่อ count มีค่าเท่ากับ 4 โปรแกรมจะลดค่า count เหลือ 3 แล้วคำสั่ง break จะสั่งให้ **หยุดการทำงานและออกจาก while loop ทั้งหมดทันที** ทำให้การวนลูปสิ้นสุดลง และไม่มีการพิมพ์ค่าใดๆ อีก

**ความแตกต่าง:**

* continue: หยุดแค่ **รอบปัจจุบัน** แล้วไปทำรอบต่อไป
* break: หยุด **ทั้ง Loop** ทันที

**ข้อ 6: การจัดการข้อมูลในลิสต์**

**คำตอบ:** สามารถใช้ลำดับคำสั่งต่อไปนี้:

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ตัวอักษร

เนื้อหาที่สร้างโดย AI อาจไม่ถูกต้อง

**ข้อ 7: ประโยชน์ของ Keyword Arguments**

**คำตอบ:** การระบุชื่อพารามิเตอร์ (Keyword Arguments) มีข้อดีที่สำคัญ 2 ประการในบริบทนี้:

1. **ความชัดเจนและอ่านง่าย (Readability):** โค้ดจะสื่อความหมายได้ในตัวเองทันทีว่า "Somchai" คือค่าสำหรับ name และ "Software Engineering" คือค่าสำหรับ major ทำให้คนอื่น (หรือตัวเราในอนาคต) เข้าใจได้ง่ายโดยไม่ต้องย้อนกลับไปดูว่าฟังก์ชันนี้กำหนดพารามิเตอร์ไว้อย่างไร
2. **ความยืดหยุ่นเรื่องลำดับ (Order Independence):** เราไม่จำเป็นต้องจำลำดับของพารามิเตอร์ให้แม่นยำ สามารถสลับที่กันได้ เช่น introduce(major="...", name="...") ก็ยังให้ผลลัพธ์ที่ถูกต้องเหมือนเดิม ซึ่งช่วยลดโอกาสเกิดข้อผิดพลาดได้มาก โดยเฉพาะในฟังก์ชันที่มีพารามิเตอร์หลายตัว

**ข้อ 8: การคืนค่าจากฟังก์ชันแบบซับซ้อน**

**คำตอบ:** สามารถแก้ไขฟังก์ชันและเรียกใช้งานได้ดังนี้

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ตัวอักษร

เนื้อหาที่สร้างโดย AI อาจไม่ถูกต้อง

**ข้อ 9: การประมวลผลข้อมูลในลิสต์ 2 มิติ**

**คำตอบ:** สามารถเขียนโปรแกรมโดยใช้ Nested Loop ได้ดังนี้

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ตัวอักษร, ซอฟต์แวร์

เนื้อหาที่สร้างโดย AI อาจไม่ถูกต้อง

**แนวคิด:** สมาชิกที่อยู่ในแนวทแยงมุมหลักจะมีคุณสมบัติร่วมกันคือ **เลขดัชนีของแถว (row index) และคอลัมน์ (column index) จะเท่ากันเสมอ** (เช่น data[0][0], data[1][1], data[2][2])

**ข้อ 10: การประยุกต์ใช้โครงสร้างข้อมูล**

**คำตอบ:** สามารถเขียนฟังก์ชัน analyze\_grades ได้ดังนี้

