กำหนดส่ง

Lab	
HW	
Until	

กระบวนวิชา 204111

การบ้านปฏิบัติการ 4

Conditionals (20 คะแนน)

ข้อกำหนด

i. การเรียกใช้ฟังก์ชันเพื่อการทดสอบ ต้องอยู่ภายใต้เงื่อนไข `if __name__ == '__main__' :` เพื่อให้สามารถ `import` ไปเรียกใช้งานจาก Script อื่น ๆ ได้อย่างมีมาตรฐาน

ii. ไม่อนุญาตให้ใช้การทำซ้ำเช่น **`for`**, **`while`** (Iterations), Recursions, หรือ Data Type อื่น ๆ ที่ยังไม่สอนในบทเรียน เช่น **`range`**, **`list`** หรือ **`map`** ในการแก้ปัญหา

iii. นักศึกษาสามารถสร้างฟังก์ชันย่อยต่าง ๆ เพิ่มเติมได้ตามความเหมาะสม

iv. ในข้อที่ระบุว่ามี **[Attachments]** ให้ Download ไฟล์ Template จาก Grader ลงมา implement

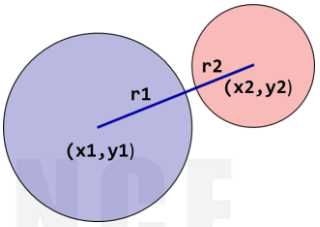
Hint: ควรสร้างฟังก์ชันทดสอบเพื่อทดสอบกับกรณีทดสอบหลายๆ ชุดโดยอัตโนมัติ โดยใช้ Statement **`assert`**

1) **4 คะแนน** (`Lab04_1_6XXXXXXXX.py`) **[Attachments]** ให้เขียนฟังก์ชัน

```
circle_intersect(x1: float, y1: float, r1: float,
                 x2: float, y2: float, r2: float,
                 epsilon=10**-6) -> int
```

เพื่อคำนวณว่าวงกลมสองวง ที่มีจุดศูนย์กลางที่ Coordinate (x_1,y_1) และ (x_2,y_2) และมีรัศมี r_1 และ r_2 ตามลำดับ สัมผัสกัน (Touching) ตัดกัน (Intersecting) หรือ ไม่ตัดกัน (Non-intersecting) โดยหากส่วนที่ใกล้ที่สุดของเส้นรอบวงของวงกลมทั้งสอง ห่างกันไม่เกินค่า *epsilon* ให้ถือว่าวงกลมทั้งสองสัมผัสกัน ทั้งนี้ระยะห่างระหว่างสองจุดใด ๆ (Distance) สามารถหาได้จากสูตร

$$distance = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$



โดยฟังก์ชันจะมีการคืนค่าดังนี้

- `1` กรณีวงกลมสองวงตัดกัน (intersecting)
- `0` กรณีวงกลมสองวงสัมผัสกัน (touching)
- `-1` กรณีวงกลมสองวงไม่ตัดและไม่สัมผัสกัน (non-intersecting)

Hint: พิจารณาศึกษาฟังก์ชัน `almost_equal()` หรือ `math.isclose()` จาก slide เรื่อง Conditionals Part I

Input	Output
2 3 5 5 7 1	1
0 0 2.5 3 4 2.5	0

2) **4 คะแนน** ให้เขียนฟังก์ชัน *my_min_mid_max*(*a*: int, *b*: int, *c*: int) -> None เพื่อแสดงผลค่าน้อยที่สุด (min) ค่าที่อยู่ตรงกลาง (mid) และค่ามากที่สุด (max) จากจำนวนเต็ม a, b และ c โดยการแสดงผลจะอยู่ในรูปแบบดังแสดงด้านล่าง ทั้งนี้ไม่อนุญาตให้ใช้ฟังก์ชัน built-in **max()**, **min()** ในการแก้ปัญหา

Hint: สามารถใช้ 3 **if** statement ในการแก้ปัญหา และควรอ่าน slide min max and more (w04) ก่อนทำการบ้าน

a. ให้เขียน Flowchart แสดง Algorithm ในการแก้ปัญหา (Flowgorithm, Lucidchart, etc) และส่งไฟล์ออนไลน์ผ่านระบบ Mango ของรายวิชา

b. (Lab04_2_6XXXXXXXX.py) เขียนฟังก์ชันในภาษา python ตาม Algorithm ที่ออกแบบไว้

Input	Output
1 2 3	min = 1 mid = 2 max = 3

3) **4 คะแนน** (HW04_1_6XXXXXXXX.py) ในเกม Pokémon Go ผู้เล่นจะได้ค่าประสบการณ์ (exp) จากการพัฒนาร่าง (Evolve) จากร่าง 1 เป็นร่าง 2 ในแต่ละครั้งเท่ากับ 1000 exp และต้องเสียลูกอม (Candy) จำนวนหนึ่ง เช่น Pidgey (ร่าง 1) จะใช้ ลูกอมจำนวน 12 ลูก เพื่อพัฒนาเป็น Pidgeotto (ร่าง 2) และรางวัลจากการพัฒนาร่างเป็นพลังเพิ่ม 1000 exp และลูกอม 1 ลูก ดังรูป

ให้เขียนฟังก์ชัน *calculate_exp*(*p*: int, *c*: int) -> int เพื่อคำนวณและคืนค่า exp ที่มากที่สุดที่เป็นไปได้เฉพาะจากการพัฒนา Pidgey เป็น Pidgeotto เมื่อมี Pidgey จำนวน p ตัว และ ลูกอมจำนวน c ลูก โดยกำหนดให้นกทุกตัว (Pidgey และ Pidgeotto) สามารถแลกเปลี่ยนเป็นลูกอมได้ 1 ลูก และจำนวนลูกอมที่ใช้ในการพัฒนาร่างเท่ากับ 12 (ค่าคงที่)

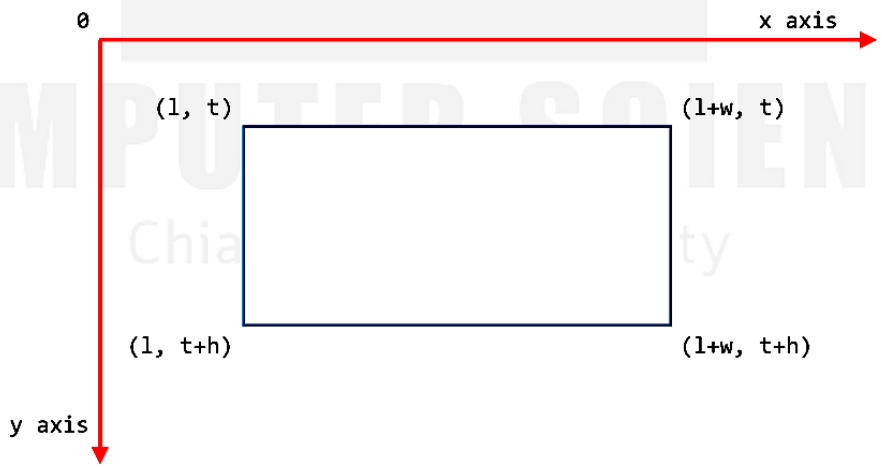
Input	Output	คำอธิบาย
1 12	1000	# มี candy เพียงพอในการ evolve 1 ครั้ง
2 12	1000	# มี candy เพียงพอในการ evolve 1 ครั้ง
2 22	2000	# evolve รอบแรกและนำ Pidgeotto ไปแลกเป็นแคนดี้ เพื่อให้เพียงพอในการ evolve ตัวที่สอง

4) **4 คะแนน** (HW04_2_6XXXXXXXX.py) **[Attachments]** ให้เขียนฟังก์ชัน
`minute_diff(h1: int, m1: int, p1: str, h2: int, m2: int, p2: str) -> int:`
เพื่อคืนค่าระยะห่างเป็นนาทีของเวลาที่ระบุด้วยจำนวนเต็ม $h1$, $m1$ และ $h2$, $m2$ ($1 \leq hx \leq 12$ และ $0 \leq mx \leq 59$) โดย hx และ mx จะแทนเวลาเป็นชั่วโมงตามเข็มนาฬิกา และนาทีตามลำดับ และตัวแปร px เป็น string ระบุช่วงเวลาก่อนหรือหลังเที่ยงในรูป `'AM'` และ `'PM'` ทั้งนี้ให้ถือว่าเวลาที่ระบุเป็นเวลาที่อยู่ในวันเดียวกันเสมอ และไม่อนุญาตให้ใช้ module **datetime** ในการแก้ปัญหา

Hint: นักศึกษาสามารถศึกษาการระบุช่วงเวลาด้วย AM และ PM ได้จาก https://en.wikipedia.org/wiki/12-hour_clock

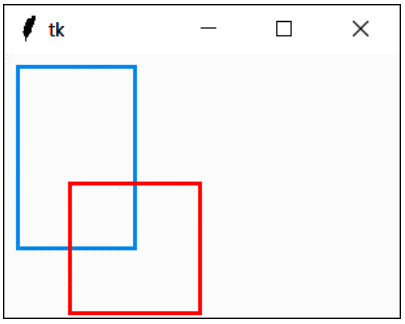
Function Call	Output
`minute_diff(8, 23, 'AM', 8, 24, 'AM')`	1
`minute_diff(8, 23, 'AM', 1, 24, 'PM')`	301
`minute_diff(1, 24, 'PM', 8, 23, 'AM')`	301

5) **4 คะแนน** (HW04_3_6XXXXXXXX.py) ให้เขียนฟังก์ชัน Boolean (ฟังก์ชันที่คืนค่า **True** หรือ **False** เท่านั้น)
`is_overlapped(l1: float, t1: float, w1: float, h1: float,`
`l2: float, t2: float, w2: float, h2: float) -> bool`
เพื่อตรวจสอบว่าสี่เหลี่ยมมุมฉากสองรูปมีส่วนทับ (Overlap) กันหรือไม่ โดยที่เราสามารถนิยามสี่เหลี่ยมมุมฉากดังนี้



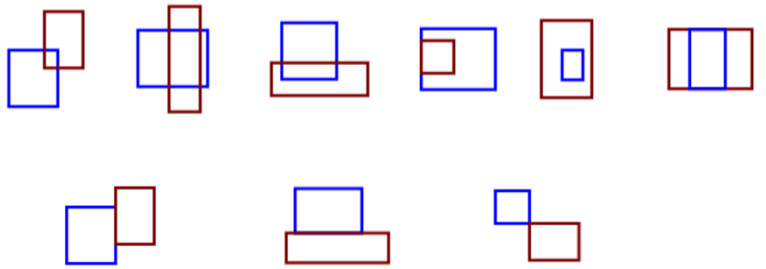
โดย t คือ top, l คือ left, w คือ width และ h คือ height ของรูปสี่เหลี่ยม

ดังนั้น `is_overlapped(10, 10, 100, 150, 50, 100, 150, 200)` จะคืนค่าเป็น **`True`** ดังรูป

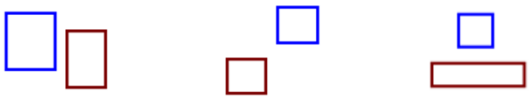


Hint: พิจารณาเงื่อนไขกรณีสี่เหลี่ยมที่**ไม่**ทับกันจะแก้ปัญหาได้ง่ายกว่า

`True`



`False`



การส่งงาน

1. ลักษณะ/ลำดับข้อความของการรับค่า/แสดงผล จะ**ต้องเป็นไปตามที่ระบุ**ในตัวอย่างการ run
2. ไฟล์งานที่ส่ง จะต้องมีการแทรก comment ที่ต้นไฟล์ตามข้อกำหนดใน canvas รายวิชา
3. ไฟล์งานโปรแกรมที่ส่ง จะต้องมีการแทรก pseudocode เป็น comment ในแต่ละขั้นตอน
4. Upload ไฟล์ source code ตามที่ระบุในแต่ละข้อ ไปยังระบบตรวจให้คะแนนอัตโนมัติ https://cmu.to/gdr111