



python ๑๐๑



ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์
จุฬาลงกรณ์มหาวิทยาลัย

PYTHON ๑๐๑

สิงหาคม ๒๕๖๑

v1.0.2

ผู้อ่านสามารถดาวน์โหลดหนังสือรุ่นล่าสุด
และร่วมแสดงความคิดเห็น/ข้อเสนอแนะเกี่ยวกับหนังสือเล่มนี้ที่

www.cp.eng.chula.ac.th/books/python101

ขอบคุณครับ

กิตติคุณ พละการ, กิตติภพ พละการ, สมชาย ประสิทธิ์จูตระกูล, สุกรี สิ้นธุญโญ

PYTHON ๑๐๑ / กิตติคุณ พละการ, กิตติภพ พละการ, สมชาย ประสิทธิ์จูตระกูล, สุกรี สิ้นธุญโญ

1. การเขียนโปรแกรม (คอมพิวเตอร์)
2. ไพทอน (คอมพิวเตอร์)

005.133

ISBN 978-616-407-189-6

พิมพ์ครั้งที่ 1 จำนวน 1,000 เล่ม พ.ศ. 2560

พิมพ์ครั้งที่ 2 จำนวน 1,000 เล่ม พ.ศ. 2561

สงวนลิขสิทธิ์ตาม พ.ร.บ. ลิขสิทธิ์ พ.ศ. 2537/2540

การผลิตและการลอกเลียนหนังสือเล่มนี้ไม่ว่ารูปแบบใดทั้งสิ้น

ต้องได้รับอนุญาตเป็นลายลักษณ์อักษรจากเจ้าของลิขสิทธิ์

จัดพิมพ์โดย

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

พญาไท กรุงเทพฯ 10330

<https://www.cp.eng.chula.ac.th>

เข้าชมหนังสือเล่มอื่น ๆ ของภาควิชาได้ที่

<https://www.cp.eng.chula.ac.th/books>

ออกแบบปก : กมลพรรณ ลีวประเสริฐ

ออกแบบรูปเล่ม : ภาณุกร สุนทรเวชพงษ์

พิมพ์ที่ โรงพิมพ์แห่งจุฬาลงกรณ์มหาวิทยาลัย โทรศัพท์ 0-2218-3549-50 โทรสาร 0-2218-3551

คำนำ

วิชา ๒๑๑๐๑๐๑ การเขียนโปรแกรมคอมพิวเตอร์ เป็นหนึ่งในวิชาพื้นฐานทางวิศวกรรมศาสตร์ ที่นิสิตชั้นปีที่ ๑ คณะวิศวกรรมศาสตร์ ทุกคนต้องลงทะเบียนเรียน วัตถุประสงค์หลักของวิชานี้คือ ให้นิสิตเข้าใจหลักการในการใช้คำสั่งต่าง ๆ ของภาษาโปรแกรม เพื่อเขียนโปรแกรมคอมพิวเตอร์ให้ตรงตามข้อกำหนดที่ได้รับ การเขียนโปรแกรมเป็นความสามารถที่ต้องลงมือฝึกฝนฝึกปฏิบัติด้วยตนเอง เหมือนกับทักษะอื่นทางวิศวกรรมที่จำเป็นต้องฝึก ๆ ๆ ให้นานาญจึงจะได้ผล ไม่สามารถได้มาด้วยการอ่าน ๆ ๆ จำ ๆ ๆ

หนังสือ PYTHON ๑๐๑ เล่มนี้ถูกจัดทำขึ้น เพื่อให้นิสิตใช้ทบทวนเนื้อหาหลังชมภาพยนตร์บรรยายเนื้อหาด้วยตนเอง ในแต่ละบท ใช้เตรียมตัวก่อนเข้าเรียน และใช้ระหว่างการเขียนโปรแกรมจริงในห้องปฏิบัติการที่จัดขึ้นเป็นกิจกรรมประจำทุกสัปดาห์ โดยมีระบบ Grader ช่วยตรวจสอบความถูกต้องของผลการทำงานของโปรแกรมอย่างอัตโนมัติ นิสิตจะได้ฝึกเขียนโปรแกรมตามโจทย์ ฝึกหาที่ผิด และฝึกตรวจสอบความถูกต้องของโปรแกรมที่พัฒนาขึ้น นอกจากนี้ ยังมีแบบฝึกปฏิบัติเพิ่มเติมอีกมากมายในระบบ Grader ให้นิสิตได้ทำเสริมอีกด้วย

ผู้เขียนต้องขอขอบคุณ ผศ. ดร. นันทิ นิภานันท์ ผู้ปรับปรุงระบบตรวจโปรแกรมอัตโนมัติ Grader เพื่อใช้ประกอบการเรียน การสอน และการสอบวิชาการเขียนโปรแกรมมาตั้งแต่ปีการศึกษา ๒๕๕๗ ขอขอบคุณบุคลากรของศูนย์คอมพิวเตอร์ คณะวิศวกรรมศาสตร์ ที่ให้บริการจัดเตรียมความพร้อมของอุปกรณ์และเครือข่ายในห้องปฏิบัติการ ขอขอบคุณคุณอาจารย์และนิสิตช่วยสอนที่ร่วมกันสร้างโจทย์ปัญหา สอน และปรับปรุงวิชา ๒๑๑๐๑๐๑ ตลอดมา ขอขอบคุณ ภานุกร สุนทรเวชพงษ์ และอดีตคุณ ออไอศูรย์ ที่ช่วยจัดรูปเล่ม และ กมลพรรณ ลีวประเสริฐ ที่ช่วยออกแบบหน้าปก ขอขอบคุณภาควิชาวิศวกรรมคอมพิวเตอร์ ที่ให้การสนับสนุนในสารพัดเรื่อง และท้ายสุดที่ต้องขอบคุณที่สุดก็คือ นิสิตคณะวิศวะฯ กว่าหลายพันคนที่ขยันหมั่นศึกษาและฝ่าฟันอุปสรรคในการเรียนวิชาพื้นฐานบังคับที่ค่อนข้างไม่คุ้นเคยนี้จนสำเร็จ *

กิตติภณ พละการ

กิตติภาพ พละการ

สมชาย ประสิทธิ์จุตระกูล

(ผู้เรียบเรียงและรวบรวมเนื้อหา)

สุกรี สินธุภิญโญ (ที่ปรึกษา)

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

จุฬาลงกรณ์มหาวิทยาลัย

๑๒ สิงหาคม ๒๕๖๐

* ภาควิชาวิศวกรรมคอมพิวเตอร์ขอขอบคุณ บริษัท เอ็กซอนโมบิล จำกัด ที่ให้การสนับสนุนค่าใช้จ่ายสำหรับการจัดพิมพ์ให้กับนิสิตทุกคนที่ลงทะเบียนเรียนวิชา ๒๑๑๐๑๐๑ ใช้ประกอบการเรียนในปีการศึกษา ๒๕๖๐ และ ๒๕๖๑

ขอขอบคุณ

คณาจารย์คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

ผศ. ดร. ชัยเชษฐ์ สายวิจิตร

ศ. ดร. ไพศาล กิตติศุภกร

รศ. ดร. อาณัติ เรืองรัมย์

รศ. ดร. พิสุทธิ เพียรมนกุล

อ. ดร. กรวิก ตันภษรานนท์

อ. ดร. พรรณี แสงแก้ว

รศ. ดร. รัชชา ทวีแสงสกุลไทย

อ. ดร. เชษฐา พันธุ์เครือบุตร

อ. ดร. สุรัฐ ขวัญเมือง

ผศ. ดร. อนุรักษ์ ศรีอริยวัฒน์

ผศ. ดร. จิรวินน์ ชีวรุ่งโรจน์

ผศ. ดร. ณัฐวุฒิ หนูไพโรจน์

ศิษย์เก่าภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

คุณวโรรส โรจนะ

คุณธนวัฒน์ มาลาบุปผา

คุณลิสดา ตรงประกอบ

คุณภัทราวุธ ชื้อสตัยาศิลป์

คุณวิโรจน์ จิรพัฒน์กุล

คุณสุภาชัย สุตันทวิบูลย์

คุณศุภเสฏฐ์ ชูชัยศรี

คุณณัฐชยา ลีละสุภกุล

ที่กรุณาให้ความรู้ถึงความสำคัญของการเขียนโปรแกรมกับงานทางวิศวกรรมในสาขาต่าง ๆ

ขอขอบคุณภาพประกอบจาก

<https://commons.wikimedia.org/wiki/File:WomaBallPython.jpg> (ภาพงู)

<https://pixabay.com/en/snake-python-tree-python-terrarium-1184810/> (ภาพลายหนังงู)

<https://pixabay.com/photo-1745096/> (ภาพพื้นหลังหน้าปก)

<https://www.python.org/community/logos/> (สัญลักษณ์ Python)

<http://www.pythontutor.com> (ภาพประกอบคำอธิบายเนื้อหา)

สารบัญ

00 : Programming in Engineering.....	1
01 : Data Type, Variable and Expression.....	5
02 : Selection (if-elif-else).....	15
03 : Repetition (while, for).....	29
04 : String.....	43
05 : File.....	55
06.1 : List.....	65
06.2 : Nested List.....	75
06.3 : List Comprehension.....	83
07 : Tuple, Dictionary and Set.....	91
08 : Function and Recursion.....	109
09 : NumPy.....	121
10 : Class.....	133
11 : Solutions to Exercises.....	149
Appendix.....	157

00 : Programming in Engineering

ศาสตร์ของวิศวกรรมไฟฟ้าเป็นศาสตร์ที่ครอบคลุมความหลากหลาย มีการเปลี่ยนแปลงและมีความพลวัตอย่างมากในปัจจุบัน ในหลายโอกาส การสร้างต้นแบบเสมือนจริงอาจมีต้นทุนสูง หรือแทบเป็นไปไม่ได้เลยในช่วงเริ่มต้นของโครงการต่าง ๆ ความสามารถในการใช้ทักษะการ **Programming** เพื่อวัตถุประสงค์ต่าง ๆ เช่น การสร้างแบบจำลองด้วยคอมพิวเตอร์เพื่อการออกแบบ การศึกษาความเป็นไปได้ในการใช้งาน การทดสอบสมรรถนะ หรือแม้กระทั่งการจำลองเพื่อหาเหตุการณ์ที่เกินความคาดหมายของวิศวกร จึงเป็นสิ่งจำเป็นอย่างมากสำหรับวิศวกรไฟฟ้าทุกสาขาในปัจจุบันและอนาคต

ผศ. ดร. ชัยเชษฐ์ สายวิจิตร
ภาควิชาวิศวกรรมไฟฟ้า



Programming เป็นเครื่องมือที่ช่วยในการหาคำตอบเพื่ออธิบายปรากฏการณ์หรือพฤติกรรมพลวัตของหน่วยปฏิบัติการหรือกระบวนการต่าง ๆ นอกจากนี้ยังใช้ประมาณค่าตัวแปรหรือพารามิเตอร์ และพัฒนาไปสู่การทำนายปรากฏการณ์หรือพฤติกรรมพลวัตของหน่วยปฏิบัติการหรือกระบวนการต่อไป

ศ. ดร. ไพศาล กิตติศุภกร
ภาควิชาวิศวกรรมเคมี

ในทางวิศวกรรมโยธา โปรแกรมมีใช้กันอย่างมากเพื่อใช้จำลองพฤติกรรมของโครงสร้างเช่น อาคาร สะพาน ฐานราก ภายใต้การรับแรงซึ่งจะทำให้ทราบแรงที่เกิดขึ้นกับโครงสร้างเพื่อที่วิศวกรโยธาจะทำการออกแบบโครงสร้างให้ปลอดภัยและประหยัด และในการก่อสร้างก็ต้องใช้โปรแกรมเพื่อกำหนดและจัดการการก่อสร้าง นอกจากนั้นก็มีการใช้โปรแกรมในการจำลองระบบขนส่งเพื่อการออกแบบหรือวิเคราะห์ระบบคมนาคมแบบต่าง ๆ

รศ. ดร. อาณัติ เรืองรัมย์
ภาควิชาวิศวกรรมโยธา



การเรียนวิศวกรรมสิ่งแวดล้อม บ่อยครั้ง เราต้องคำนวณและสรุปผลวิเคราะห์ ค่าพารามิเตอร์ทางสิ่งแวดล้อมซึ่งคำนวณไม่ง่าย และไม่ซับซ้อนถึงขนาดต้องเรียน ภาคคอม ฯ การที่นิสิตได้เรียน **Programming** เบื้องต้น จะสามารถช่วยให้นิสิตสามารถ ต่อยอดความรู้ไปสู่งานวิศวกรรมสิ่งแวดล้อมในโลกอนาคตได้อย่างง่ายดาย

รศ. ดร. พิสุกรี เพ็ชรมนกุล
ภาควิชาวิศวกรรมสิ่งแวดล้อม



การเขียนโปรแกรมมีความสำคัญกับงานทุกด้านของวิศวกรรมสำรวจ โดยเฉพาะอย่างยิ่งงานด้าน GIS ที่ต้องการ**การเขียนโปรแกรม**ในการจัดการ ประมวลผล วิเคราะห์ และแสดงผลข้อมูล โดยเฉพาะข้อมูลขนาดใหญ่ที่ซอฟต์แวร์พื้นฐาน โดยทั่วไปเช่น MS Excel ไม่สามารถรองรับได้ นอกจากนี้ ยังมีความสำคัญต่อการ พัฒนาซอฟต์แวร์ด้านแผนที่ โดยเฉพาะระบบแผนที่บนเว็บที่ต้องอาศัยการพัฒนาด้วย **การเขียนโปรแกรม** และการเขียนโปรแกรมสำคัญมากที่สุดกับการศึกษาในระดับสูง เนื่องจากจะต้องพัฒนาซอฟต์แวร์ใหม่ขึ้นมาใหม่เอง

อ. ดร. กรวิก ตนกษรานนท์
ภาควิชาวิศวกรรมสำรวจ

Programming มีความสำคัญต่อการเรียนในสาขาวิชาวิศวกรรมนิวเคลียร์ค่อนข้างมาก การประมวลผลการตรวจวัดรังสีแบบที่เป็นจำนวนนับรังสี จำนวนข้อมูลการนับรังสี มีจำนวนมากต่อการวัด เครื่องตรวจนับรังสีต้องใช้**โปรแกรมคอมพิวเตอร์**ในการรวบรวม และประมวลผลข้อมูลมาสร้างเป็นแถบสเปกตรัมข้อมูล สำหรับการประมวลผลที่ได้จากภาพ การฉายรังสีแบบภาพคอนทราสต์ขาว-ดำ การถ่ายภาพรังสีแบบทะลุผ่านและมีสแต็ปการถ่ายภาพ หมุนรอบวัตถุหนึ่งที่เรียกว่า CT-scan เมื่อรวบรวมผลของแต่ละสแต็ปมารวมกันจะได้ ภาพตัดขวางของวัตถุหรือถ้ามีการวัดในแนวตั้งด้วยทำให้ได้ภาพสุดท้ายเป็นภาพ 3D ซึ่งจะได้ เป็นโมเดลที่มีรายละเอียดภายในด้วย โดยเทคนิคนี้ทำให้ได้ข้อมูลภายในของวัตถุหรือแม้แต่ ภายในของร่างกายมนุษย์ได้ และยังออกแบบ**โปรแกรม**ในการควบคุมอุปกรณ์การตรวจวัดรังสีให้ทำงานได้อย่างอัตโนมัติ และ ควบคุมได้จากระยะไกล ทำให้ผู้ใช้รับปริมาณรังสีน้อยลง สำหรับระบบการควบคุมการทำงานทั้งหมดของโรงไฟฟ้านิวเคลียร์นั้น ทำโดย**โปรแกรมคอมพิวเตอร์**ทั้งหมด



อ. ดร. พรรณี แสงแก้ว
ภาควิชาวิศวกรรมนิวเคลียร์

บทบาทของวิศวกรรมอุตสาหการ คือ การออกแบบ ดำเนินการ ปรับปรุงและสร้างสรรค์นวัตกรรมระบบ ทั้งการผลิต บริการ และ ธุรกิจ

ทักษะ **Programming** ช่วยให้มีตรรกะการคิดอย่างมีเหตุผล อีกทั้งยังเป็นเครื่องมือในการวิเคราะห์และประมวลผลอย่างรวดเร็วและแม่นยำตามตัวแปรและข้อจำกัดในบริบทหนึ่ง ๆ ซึ่งมีความซับซ้อนมากขึ้นในอนาคต

รศ. ดร. ณัฐชา กวีแสงสกุลไทย
ภาควิชาวิศวกรรมอุตสาหการ



วิทยาการทางด้านวิศวกรรมโลหการและวัสดุ เป็นความรู้ที่เกี่ยวข้องกับหลักการพื้นฐานของวัสดุต่าง ๆ กระบวนการแปรรูปและขึ้นรูปโลหะ สมบัติของวัสดุ และการเลือกและออกแบบวัสดุที่เหมาะสมกับงานที่หลากหลาย ดังนั้น เพื่อให้สามารถพัฒนาวัสดุใหม่ให้มีสมบัติต่าง ๆ ที่ดีขึ้น **Computer Programming** จึงเข้ามามีบทบาทสำคัญในการสร้างแบบจำลองต่าง ๆ เพื่อเชื่อมโยงความสัมพันธ์ระหว่าง กระบวนการผลิต - โครงสร้างของวัสดุ - สมบัติของวัสดุ - ความสามารถในการใช้งาน ในปัจจุบันมีการนำ **Computer Programming** เข้ามาใช้อย่างแพร่หลายมากขึ้น อาทิเช่น การทำนายโครงสร้างจุลภาคของวัสดุภายหลังการขึ้นรูปด้วยกรรมวิธีต่าง ๆ เช่น Casting, 3D printing, Metal Forming จนสามารถทำให้ปรับปรุงให้วัสดุมีความแข็งแรงที่สูงขึ้นได้ หรือมีการนำมาใช้เพื่อทำนายการพังเสียหายของวัสดุจากการเกิด Fatigue และ Corrosion ทำให้สามารถวางแผนการซ่อมบำรุงได้อย่างเหมาะสมมากขึ้น นอกจากนี้ยังมีการนำ Artificial Intelligence เข้ามาใช้เพื่อหาส่วนผสมของวัสดุใหม่ ๆ ที่ยังไม่เคยมีการค้นพบอีกด้วย

อ. ดร. เชษฐา พันธุ์ศรีบุญตร
ภาควิชาวิศวกรรมโลหการ

Programming เป็นเครื่องมือที่ใช้ในการสร้าง คำนวณและวิเคราะห์ผลของแบบจำลองทางวิศวกรรมของระบบทางกล ความร้อน ของแข็งและของไหล นอกจากแบบจำลองแล้ว ยังใช้ในการควบคุมระบบทางกลต่าง ๆ เช่น ระบบอัตโนมัติหุ่นยนต์ เป็นต้น

อ. ดร. สุวัจ ขวัญเมือง
ภาควิชาวิศวกรรมเครื่องกล





งานทางด้านวิศวกรรมแหล่งน้ำมีความจำเป็นที่จะต้องยุ่งเกี่ยวกับข้อมูลจำนวนมากทั้งข้อมูลน้ำฝน ข้อมูลน้ำท่า ข้อมูลสภาพพื้นที่ ข้อมูลความต้องการการใช้น้ำ **Programming** จึงเป็นเครื่องมือสำคัญในการช่วยจัดการ วิเคราะห์ และประมวลผล ข้อมูลต่าง ๆ อย่างมีประสิทธิภาพ

ผศ. ดร. อนุรักษ์ ศรีอริยวัฒน์
ภาควิชาวิศวกรรมแหล่งน้ำ

การใช้แบบจำลองแหล่งกักเก็บปิโตรเลียมเป็นสิ่งจำเป็นในงานวิศวกรรมปิโตรเลียม เพื่อทำความเข้าใจกับพฤติกรรมการผลิตในอดีตและปัจจุบัน และยังใช้ทำนายการผลิตในอนาคตอีกด้วย นอกจากนี้ปริมาณข้อมูลการผลิตที่ถูกจัดเก็บอย่างต่อเนื่อง จำเป็นต้องมีการบริหารจัดการที่ดี เพื่อให้วิศวกรสามารถนำไปใช้ประโยชน์ได้หลากหลายรูปแบบ ความสำคัญของการเข้าใจและสามารถเขียนโปรแกรมเพื่อใช้งานเป็นการเฉพาะเป็นสิ่งจำเป็นสำหรับวิศวกรปิโตรเลียม

สำหรับงานวิศวกรรมเหมืองแร่มีการนำ **Programming** ไปใช้ในการออกแบบการทำเหมืองแร่ เช่น การเปิดหน้าดินเพื่อนำแร่ออกมา การคำนวณเสถียรภาพของชั้นดิน การขนส่งวัตถุดิบจากบริเวณหน้าเหมือง การใช้ธรณีสถิติเพื่อประกอบการคำนวณปริมาณสำรอง เป็นต้น



ผศ. ดร. จิรวัดน์ ชูรุ่งโรจน์
ภาควิชาวิศวกรรมเหมืองแร่และปิโตรเลียม



การที่มนุษย์เราเป็นสัตว์สังคมที่อยู่ร่วมกัน ทำให้สิ่งที่เราจะเรียนรู้ตั้งแต่เด็ก ๆ คือภาษาที่เราใช้ในการสื่อสารระหว่างกัน เพื่อให้มนุษย์เราสามารถเข้าใจกันได้เป็นอย่างดี เปรียบเช่นเดียวกัน การที่จะเข้าใจสิ่งต่าง ๆ ในโลกปัจจุบัน ที่เป็นโลกดิจิทัล จำเป็นอย่างยิ่งที่เราจะต้องเรียนรู้ภาษาที่คอมพิวเตอร์เข้าใจ ดังเช่นภาษา Python ซึ่งจะช่วยให้เราได้เข้าใจการทำงานของคอมพิวเตอร์มากยิ่งขึ้น ดังนั้น นิสิตที่ศึกษาในภาควิชาวิศวกรรมคอมพิวเตอร์ มีความจำเป็นอย่างยิ่งที่จะต้องเรียนรู้การเขียนโปรแกรม เพราะไม่เพียงแต่จะต้องเข้าใจกลไกและกระบวนการทำงานของคอมพิวเตอร์เท่านั้น แต่ยังจะต้องสามารถเขียนโปรแกรมเพื่อควบคุมการทำงานของคอมพิวเตอร์ได้อย่างเหมาะสมและมีประสิทธิภาพ

ผศ. ดร. ณัฐวุฒิ หนูไพโรจน์
ภาควิชาวิศวกรรมคอมพิวเตอร์

01 : Data Type, Variable and Expression

สรุปเนื้อหา

ตัวแปรเป็นที่เก็บข้อมูลในโปรแกรม ต้องมีชื่อกำกับ

- ชื่อตัวแปรประกอบด้วยตัวอักษร ตัวเลข หรือเครื่องหมายขีดเส้นใต้ _ ตัวอังกฤษใหญ่ไม่เหมือนตัวเล็ก ห้ามขึ้นต้นชื่อด้วยตัวเลข
- อย่าตั้งชื่อตัวแปรซ้ำกับชื่อฟังก์ชันใน Python เช่น `int`, `str`, `max`, `sum`, `abs`, ... (ไม่ห้ามถ้าจะตั้งซ้ำ แต่ไม่ควรทำอย่างยิ่ง)

ข้อมูลใน Python ที่นำมาประมวลผลมีหลายประเภท ที่เราจะศึกษาในบทนี้มีดังต่อไปนี้

<code>int</code>	จำนวนเต็ม	-10 5000011 (Python ห้ามไม่ให้เขียน 0 นำหน้าจำนวนเต็ม เช่น 020)
<code>float</code>	จำนวนจริง	10.0 1.23e59 มีค่าเท่ากับ 1.23×10^{59}
<code>str</code>	ข้อความ	'Programming is easy' "Let's go shopping"

การให้ค่ากับตัวแปร

- `a = b = c = 0.0` ให้ตัวแปร `a` `b` และ `c` เก็บจำนวนจริง 0.0
- `a = 5; b = 6; a, b = b, a` ตัวแปร `a` กับ ตัวแปร `b` สลับค่ากันได้ `a` เก็บ 6 และ `b` เก็บ 5
- `a = x` ถ้า `x` ไม่เคยมีการให้ค่ามาก่อน คำสั่งนี้จะผิด เพราะไม่รู้ `x` มีค่าเท่าใด

ตัวดำเนินการ ลำดับการทำงาน และการแปลงประเภทข้อมูล

- ตัวดำเนินการ บวก (+), ลบ (-), คูณ (*), ยกกำลัง (**),หาร (/),หารปัดเศษ (/), เศษจากการหาร (%)
- การดำเนินการระหว่างจำนวนเต็มกับจำนวนจริงจะได้ผลเป็นจำนวนจริง (เช่น `2 + 1.0` ได้ `3.0`)
- // กับจำนวนลบ : `1//2` ได้ 0, `(-1)//2` ได้ -1, `11//10` ได้ 1, `(-11)//10` เหมือน `11// -10` ได้ -2
- `a = 3+97//2**3%8` a มีค่า `3+97//8%8 = 3+12%8 = 3+4 = 7`
- `a = 12//3/2+2**3**2` a มีค่า `12//3/2+2**9 = 12//3/2+512 = 4/2+512 = 2.0+512 = 514.0`
- `a += 2` ก็คือ `a = a + 2`, `a /= 2` ก็คือ `a = a / 2`, `a *= -1` ก็คือ `a = a * -1`
- ถ้า `import math` จะมีค่าคงตัวและฟังก์ชันทางคณิตศาสตร์ให้ใช้มากมาย
 - `math.pi`, `math.e`, `math.sin(x)`, `math.cos(x)`, `math.sqrt(x)`, `math.log(x,b)`, ...
- นำสตริงบวกกัน คือนำสตริงมาต่อกัน เช่น `'12'+'23'` คือ `'1223'`
- สตริงคูณกับจำนวนเต็ม คือนำสตริงนั้นมาต่อกันเป็นจำนวนครั้งเท่ากับค่าของจำนวนเต็มนั้น เช่น `'12'*3` คือ `'121212'`
- ฟังก์ชัน `int`, `float` และ `str` มีไว้เปลี่ยนประเภทข้อมูล เช่น
 - `int('12')` ได้ 12, `float('1.2')` ได้ 1.2, `str(12//2)` ได้ '6', `str('Aa')` ได้ 'Aa'
- ข้อควรระวัง :** รู้ความแตกต่างของ / กับ // และศึกษาลำดับการทำงานของ operator ให้ดี ถ้าไม่มั่นใจ ใส่วงเล็บ เช่น `a/2*b` เท่ากับ `(a/2)*b` แต่ `a/(2*b)` เท่ากับ `a/2/b`

คำสั่งการแสดงผลข้อมูลทางจอภาพ

- `print(a,b,c)` นำค่าในตัวแปร a b และ c มาแสดงต่อกันคั่นด้วยช่องว่างบนบรรทัดเดียวกัน
- `print(str(a)+str(b)+str(c))` นำค่าในตัวแปร a b และ c มาเปลี่ยนเป็นสตริงต่อกัน แล้วแสดงบนบรรทัดเดียวกัน

คำสั่งการอ่านข้อมูลจากแป้นพิมพ์

- `a = input()` อ่านข้อความจากแป้นพิมพ์หนึ่งบรรทัด เก็บใส่ตัวแปร a (เป็นสตริง)
 - `a = input().strip()` อ่านข้อความจากแป้นพิมพ์หนึ่งบรรทัด ตัดช่องว่างทางซ้ายและขวาออก เก็บใส่ตัวแปร a
 - `a = int(input())` อ่านจำนวนเต็มหนึ่งจำนวนจากแป้นพิมพ์ เก็บใส่ตัวแปร a
 - `a = float(input())` อ่านจำนวนจริงหนึ่งจำนวนจากแป้นพิมพ์ เก็บใส่ตัวแปร a
 - ถ้าต้องการอ่านข้อมูลหลาย ๆ ตัวที่ผู้ใช้ป้อนเข้ามาในบรรทัดเดียวกัน โดยข้อมูลแต่ละตัวคั่นด้วยช่องว่าง
 - `a,b,c = [e for e in input().split()]` หรือ อ่านสตริง 3 ตัว
 - `a,b,c = input().split()` อ่านจำนวนเต็ม 2 จำนวน
 - `a,b,c = [float(e) for e in input().split()]` อ่านจำนวนจริง 3 จำนวน
 - หากจะอ่านจำนวนจริงตามด้วยจำนวนเต็ม ก็อ่านเป็นสตริงก่อน โดยใช้คำสั่ง `f,n = input().split()` แล้วจึงค่อยแปลงเป็นจำนวนจริงกับจำนวนเต็ม โดยใช้คำสั่ง `f = float(f); n = int(n)`
- *** ถ้าโจทย์บอกว่าข้อมูลที่รับมาคั่นด้วยช่องว่างในบรรทัดเดียวกัน อย่าใช้ `input().split(' ')` แต่ควรใช้ `input().split()` แทน

เรื่องพิศบอย

รับข้อมูลจากแป้นพิมพ์แล้วลึ้มแปลงเป็นจำนวน ก่อนนำไปคำนวณ	<code>x = input()</code> <code>y = x**2 + 7</code> ผิดเพราะ x เป็นสตริง
ลำดับการทำงานของตัวดำเนินการ + - * / // % ** ผิด (** ทำก่อน * / // % ทำก่อน + -)	<code>y = x / 2*a</code> จะได้ $y = (x/2)*a$ ถ้าต้องการคำนวณ $y = \frac{x}{2a}$ ต้องเขียน <code>y = x/(2*a)</code> <code>y = x**1/3</code> จะได้ $(x**1)/3$ ถ้าต้องการหารากที่สามของ x ต้องเขียน <code>y = x**(1/3)</code>
ลึ้มใส่ * สำหรับการคูณ	<code>y = 2x + 1</code> ต้องเขียน <code>y = 2*x + 1</code>
10e7 มีค่าไม่เท่ากับ 10^7	10e7 มีค่าเท่ากับ 10×10^7 อยากได้ 10^7 ต้องเขียน 1e7
1e3 ไม่ใช่จำนวนเต็ม 1000	1e3 มีค่าเท่ากับ 1000.0 ดังนั้น 2345 % 1e2 ได้ 45.0

สำหรับผู้ที่เคยเรียนภาษา C อย่าเผลอเขียนคำสั่ง ++k หรือ --k	++k คือการติดบวกค่าใน k สองครั้ง จึงมีค่าเท่ากับ k ค่าใน k ไม่เปลี่ยน --k คือการติดลบค่าใน k สองครั้ง จึงมีค่าเท่ากับ k ค่าใน k ไม่เปลี่ยน
ลืม import math เมื่อใช้ฟังก์ชันของ math	$y = (-b + \text{math.sqrt}(b*b - 4*a*c)) / (2*a)$ จะฟ้องว่าไม่รู้จัก math
ใส่วงเล็บเปิดกับปิดไม่ครบ	import math $y = 2 + (x * \text{abs}(y - z / 2))$ วงเล็บปิดมีน้อยไป $y = -b + \text{math.sqrt}(b*b - 4*a*c) / (2*a)$ ขาดวงเล็บเปิด
สะกดชื่อตัวแปรผิด หรือผิดเรื่องการใช้ ตัวอังกฤษเล็กกับใหญ่	count = 0 Count = count + 1 Count กับ count เป็นคนละตัว
ตั้งชื่อตัวแปรซ้ำกับชื่อฟังก์ชันมาตรฐาน ใน IDLE ตัวแปรที่ถูกต้องมีสีดำ เป็นสีอื่นจะ สร้างปัญหา	int = 27 print(int) ได้ 27 แต่หลังจากนี้ ใช้คำสั่ง a = int(input()) เพื่ออ่านข้อมูลจากแป้นพิมพ์ แล้วแปลงเป็นจำนวนเต็มไม่ได้แล้ว (IDLE แสดง int ด้วยสีม่วง)
นำข้อมูลที่ไม่ใช่สตริงมาบวกกับสตริง	import math a = math.pi * r**2 print('area = '+a) ผิด print('area = '+str(a)) แปลง a เป็นสตริงก่อน print('area = ',a) แบบนี้ print แปลง a เป็นสตริงให้



Problem	Code
<p><u>Input:</u> รับจำนวนเต็ม 3 จำนวนจากแป้นพิมพ์ (บรรทัดละจำนวน) เก็บในตัวแปร h, m และ s ซึ่งแทนจำนวน ชั่วโมง นาที และ วินาที</p> <p><u>Process:</u> คำนวณจำนวนวินาทีรวมที่คิดจาก h, m และ s</p> <p><u>Output:</u> จำนวนวินาทีรวมทั้งหมดที่คำนวณได้</p>	
<p><u>Input:</u> รับจำนวนจริง 1 จำนวนจากแป้นพิมพ์ เก็บใน x</p> <p><u>Process:</u> คำนวณ $y = 2 - x + \frac{3}{7}x^2 - \frac{5}{11}x^3 + \log_{10}(x)$</p> <p><u>Output:</u> ค่า y ที่คำนวณได้</p>	

Problem	Code
<p>Input: รับจำนวนจริง 1 จำนวนจากแป้นพิมพ์ เก็บใน a</p> <p>Process: ให้ x มีค่าเป็น 1 จากนั้นทำคำสั่ง</p> $x = (x + a/x)/2$ <p>จำนวน 4 ครั้ง</p> <p>Output: ค่า x ที่ได้จากการทำงานข้างบนนี้</p>	
<p>Input: มี 2 บรรทัด แต่ละบรรทัดมีจำนวนจริง 3 จำนวน คั่นด้วยช่องว่าง อ่านบรรทัดแรกเก็บใน v1, v2, v3 แทนเวกเตอร์</p> <p>$v = (v1, v2, v3)$ อ่านบรรทัดที่สองเก็บใส่ u1, u2, u3 แทนเวกเตอร์ $u = (u1, u2, u3)$</p> <p>Process: คำนวณ dot product ของเวกเตอร์ v กับ u</p> <p>Output: ค่า dot product ที่คำนวณได้</p>	
<p>Input: อ่านจำนวนจริง 4 จำนวนคั่นด้วยช่องว่างจากแป้นพิมพ์ เก็บใน x1, y1, x2 และ y2 ค่าของ x1, y1 แทนพิกัดของจุดที่ 1 และ x2, y2 แทนพิกัดของจุดที่ 2 บนระนาบ x-y</p> <p>Process: คำนวณระยะทางสั้นสุดระหว่างจุดทั้งสอง</p> <p>Output: ระยะทางที่ทำได้</p>	
<p>Input: อ่านพิกัดเชิงขั้วของจุดบนระนาบ ซึ่งเป็นจำนวนจริง 2 จำนวนคั่นด้วยช่องว่าง เก็บในตัวแปร r และ theta (เป็นเรเดียน)</p> <p>Process: คำนวณค่า x และ y ซึ่งเป็นพิกัดคาร์ทีเซียนของจุด (r, theta) ที่อ่านเข้ามา</p> <p>Output: ค่า x และ y (คั่นด้วยช่องว่าง)</p>	
<p>Input: อ่านพิกัดคาร์ทีเซียนของจุดบนระนาบ ซึ่งเป็นจำนวนจริง 2 จำนวนคั่นด้วยช่องว่าง เก็บในตัวแปร x และ y</p> <p>Process: คำนวณค่า r และ theta (เป็นเรเดียน) ซึ่งเป็นพิกัดเชิงขั้วของจุด (x, y)</p> <p>Output: ค่า r และ theta (คั่นด้วยช่องว่าง)</p>	

Problem	Code
Input: อ่านจำนวนจริง 5 จำนวน คั่นด้วยช่องว่าง Process: คำนวณค่าเฉลี่ยของจำนวนทั้งห้า Output: ค่าเฉลี่ยที่ได้	
Input: รับข้อมูล 3 ตัว a, b กับ c คั่นด้วยช่องว่าง a และ b เป็นตัวอักษร ส่วน c เป็นจำนวนเต็ม Output: ตัวอักษรใน a ต่อกับตัวอักษรใน b ต่อกับ ค่าของจำนวนเต็มใน c ต่อกับ ชุดของตัวอักษรใน a ต่อกับตัวอักษร ใน b ที่ซ้ำ ๆ กันเป็นจำนวน c ชุด เช่นผู้ใช้ป้อน v o 5 จะแสดง vo5vovovovovo	



Triangle

จงเขียนโปรแกรมคำนวณพื้นที่สามเหลี่ยมที่ทราบความยาวด้านสองด้าน (a กับ b) และมุมระหว่างด้านสองด้านนั้น (C) จากสูตร

$$area = \frac{1}{2} ab \sin C$$

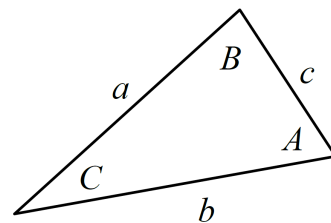
► ข้อมูลนำเข้า

บรรทัดแรกคือความยาวด้าน a (หน่วยเป็นเซนติเมตร)
บรรทัดที่สองคือความยาวด้าน b (หน่วยเป็นเซนติเมตร)
บรรทัดที่สามคือมุมระหว่างด้านทั้งสอง C (หน่วยเป็นองศา)

► ข้อมูลส่งออก

พื้นที่ของสามเหลี่ยมที่รับเป็นข้อมูลนำเข้า (หน่วยเป็นตารางเซนติเมตร) แสดงในรูปแบบที่แสดงตามตัวอย่างด้านล่าง

► ตัวอย่าง



Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
10.0 10 90.0	area = 50.0 (sq cm)
1e1 2e1 50.5	area = 77.162458338772 (sq cm)

ตัวอย่างการเขียนโปรแกรม

โปรแกรม	คำอธิบาย
<pre>a = input() b = input() C = input() area = 1/2 a*b*sin C print(area)</pre>	<p>Invalid Syntax ผิดที่บรรทัดที่ 4</p> <p>ลืมเขียน * --> เปลี่ยนเป็น <code>area = 1/2*a*b*sin C</code></p>
<pre>a = input() b = input() C = input() area = 1/2*a*b*sin C print(area)</pre>	<p>Invalid Syntax ผิดที่บรรทัดที่ 4</p> <p>ลืมใส่วงเล็บ --> เปลี่ยนเป็น <code>area = 1/2*a*b*sin(C)</code></p>
<pre>a = input() b = input() C = input() area = 1/2*a*b*sin(C) print(area)</pre>	<p>สั่ง run, ใส่ข้อมูล, ผิดที่บรรทัดที่ 4</p> <p><code>TypeError: can't multiply sequence by non-int of type 'float'</code></p> <p>แปลว่า ระบบไม่สามารถคูณได้ เพราะ a เป็นสตริง (b และ C ด้วย) จากบรรทัดที่ 1, 2 และ 3 จึงต้องแปลงให้เป็นจำนวนก่อน ในโจทย์ไม่ได้บอกว่าความยาวด้านและมุมเป็น int หรือ float แต่ถ้าดูตัวอย่างพบว่าใส่ได้ทั้ง int และ float จึงต้องแปลงสตริงจาก <code>input()</code> ให้เป็น float</p>
<pre>a = float(input()) b = float(input()) C = float(input()) area = 1/2*a*b*sin(C) print(area)</pre>	<p>Invalid Syntax บอกว่าผิดบรรทัดที่ 2</p> <p>ระบบบอกผิดบรรทัดใด ให้ดูบรรทัดก่อนหน้าด้วย เพราะผิดก่อนหน้า บางทีลามมาบรรทัดถัดมา ในที่นี้ เห็นได้ว่า ลืมใส่วงเล็บปิด</p>
<pre>a = float(input()) b = float(input()) C = float(input()) area = 1/2*a*b*sin(C) print(area)</pre>	<p>สั่ง run, ใส่ข้อมูล, ผิดที่บรรทัดที่ 4</p> <p><code>NameError: name 'sin' is not defined</code></p> <p>แปลว่า ระบบไม่รู้จักคำว่า sin ก็เพราะว่าต้องเขียน <code>math.sin</code></p>
<pre>a = float(input()) b = float(input()) C = float(input()) area = 1/2*a*b*math.sin(C) print(area)</pre>	<p>สั่ง run, ใส่ข้อมูล, ผิดที่บรรทัดที่ 4</p> <p><code>NameError: name 'math' is not defined</code></p> <p>แปลว่า ระบบไม่รู้จักคำว่า math ก็เพราะว่าต้อง <code>import math</code></p>

โปรแกรม	คำอธิบาย
<pre>import math a = float(input()) b = float(input()) C = float(input()) area = 1/2*a*b*math.sin(C) print(area)</pre> <div data-bbox="201 623 584 850" style="border: 1px solid black; padding: 10px; margin-top: 20px;"> <p>สามารถใช้ฟังก์ชัน <code>math.radians(d)</code> ซึ่งรับ <code>d</code> เป็นองศาได้ผลเป็นเรเดียน</p> </div>	<p>สั่ง run, ใส่ข้อมูลตามตัวอย่างแรก</p> <p>10 10 90 ได้ผล</p> <p>44.699833180027895</p> <p>ไม่ตรงกับที่แสดง ต้องได้พื้นที่ 50.0 ได้ผลผิด ก็น่าจะมีผิดที่การคำนวณ ลองคำนวณเองดู</p> $1/2 \times 10 \times 10 \times \sin(90) = 1/2 \times 10 \times 10 \times 1$ <p>ก็น่าจะได้ 50.0 แล้วทำไมไม่ใช่ ไม่ใกล้เคียงด้วย ตัวที่น่าสงสัยสุดก็น่าจะเป็น <code>math.sin(90)</code> เมื่อเรียกใช้ฟังก์ชัน เราต้องเข้าใจกฎเกณฑ์ของการเรียกใช้ด้วย ลองค้น python <code>math.sin</code> ในเน็ต จะพบข้อความว่า</p> <p><code>math.sin(x)</code></p> <p>Return the sine of x radians.</p> <p>แสดงว่า ต้องแปลงองศาเป็นเรเดียนก่อนส่งไปให้ <code>math.sin</code> ก็ต้องคิดวิธีแปลง: 180 องศา เท่ากับ π, C องศา ก็เท่ากับ $C \times \pi / 180$ แล้วจะใช้ค่า π เท่าไรดี จะใช้ $C \times (22/7) / 180$ หรือ $C \times 3.14159 / 180$ แต่น่าจะรู้ว่า ควรใช้ <code>math.pi</code> เพราะระบบเก็บค่า π ที่ละเอียดมากไว้ที่นี้ ดังนั้นใช้ <code>CR = C * math.pi / 180</code> เปลี่ยนเป็นเรเดียนก่อนแล้วค่อยไปใช้</p>
<pre>import math a = float(input()) b = float(input()) C = float(input()) CR = C*math.pi/180 area = 1/2*a*b*math.sin(CR) print(area)</pre>	<p>สั่ง run, ใส่ข้อมูลตามตัวอย่างแรก, ได้ 50.0 ถูกต้อง run อีกครั้ง, ใส่ข้อมูลของอีกตัวอย่าง</p> <p>1e1 2e1 50.5 ได้ 77.162458338772 ก็ถูกต้อง</p> <p>submit เข้า Grader ตรวจให้คะแนน --> ได้ 0, ทำไม ????</p> <p>คำนวณพื้นที่ได้ถูกต้อง แต่แสดงผลไม่เหมือนกับที่โจทย์บอก ดูที่ตัวอย่าง</p> <p><code>area = 50.0 (sq cm)</code></p> <p>แต่โปรแกรมแสดงแค่พื้นที่ แก่ไขบรรทัดสุดท้ายให้ตรงตามตัวอย่าง</p>
<pre>import math a = float(input()) b = float(input()) C = float(input()) CR = C*math.pi/180 area = 1/2*a*b*math.sin(CR) print("area =",area, "(sq cm)")</pre>	<p>สั่ง run, ใส่ข้อมูลตามตัวอย่างแรก, ได้</p> <p><code>area = 50.0 (sq cm.)</code></p> <p>มั่นใจว่าถูก, submit เข้า Grader, แต่ตรวจแล้วได้ 0, ทำไม ????</p> <p>ใจเย็น ๆ ดูให้มั่นใจว่าเหมือนกับที่โจทย์ต้องการไหม ?</p> <p><code>area = 50.0 (sq cm)</code></p> <p>พบว่า มีจุดเกินมาหนึ่งตัว ก็ลบจุดทิ้ง</p>
<pre>import math a = float(input()) b = float(input()) C = float(input()) CR = C*math.pi/180 area = 1/2*a*b*math.sin(CR) print("area =",area, "(sq cm)")</pre>	<p>สั่ง run, ใส่ข้อมูลตามตัวอย่างแรก, ได้</p> <p><code>area = 50.0 (sq cm)</code></p> <p>มั่นใจว่าถูก ชัวร์, submit เข้า Grader ตรวจ ได้ 100 เต็ม</p>

ตัวอย่างโจทย์ปัญหา

แปลงอุณหภูมิ

สูตรในการเปลี่ยนค่าจากองศาเซลเซียสไปเป็นองศาฟาเรนไฮต์และเคลวินมีดังนี้

$$F = \frac{9}{5}C + 32$$

$$K = C + 273.15$$

ให้อ่านข้อมูลอุณหภูมิ (หน่วยเป็นองศาเซลเซียส) จากนั้นคำนวณหาค่าองศาฟาเรนไฮต์และเคลวินด้วยสมการด้านบน เมื่อ C คือ องศาเซลเซียส F คือ องศาฟาเรนไฮต์ และ K คือ เคลวิน

► ข้อมูลนำเข้า

หนึ่งบรรทัดประกอบด้วยค่าองศาเซลเซียสเป็นจำนวนจริง

► ข้อมูลส่งออก

มีหนึ่งบรรทัดประกอบด้วยตัวเลขจำนวนจริงสองจำนวน ตัวแรกเป็นองศาฟาเรนไฮต์ และตัวที่สองเป็นเคลวิน

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
39.85	103.73 313.0

Triangle 2

จงเขียนโปรแกรมคำนวณหาความยาวของด้านที่สามของสามเหลี่ยม เมื่อเราทราบความยาวด้านสองด้าน (a กับ b) และมุมระหว่างด้านสองด้านนั้น (C) ซึ่งคำนวณได้จาก Law of Cosines

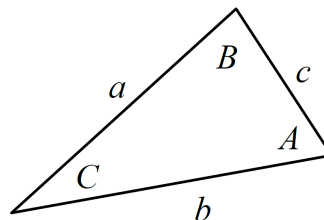
$$c^2 = a^2 + b^2 - 2ab \cos(C)$$

► ข้อมูลนำเข้า

บรรทัดแรกคือความยาวด้าน a (หน่วยเป็นเซนติเมตร)

บรรทัดที่สองคือความยาวด้าน b (หน่วยเป็นเซนติเมตร)

บรรทัดที่สามคือมุมระหว่างด้านทั้งสอง C (หน่วยเป็นองศา)



► ข้อมูลส่งออก

ความยาวด้านที่สามของสามเหลี่ยมที่รับเป็นข้อมูลนำเข้า (หน่วยเป็นเซนติเมตร) แสดงในรูปแบบที่แสดงตามตัวอย่างด้านล่าง

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
3 4 90	c = 5.0 cm.
7.0 24.0 90.0	c = 25.0 cm.
10 10 60	c = 9.999999999999998 cm.
3 3 60	c = 2.9999999999999996 cm.

ขั้นตอนการทำงานของโปรแกรม

1. รับข้อมูลจากแป้นพิมพ์ เปลี่ยนเป็นจำนวนจริง แล้วเก็บในตัวแปร a
2. รับข้อมูลจากแป้นพิมพ์ เปลี่ยนเป็นจำนวนจริง แล้วเก็บในตัวแปร b
3. รับข้อมูลจากแป้นพิมพ์ เปลี่ยนเป็นจำนวนจริง แล้วเก็บในตัวแปร D
4. นำ D ที่มีหน่วยเป็นองศา แปลงเป็น เรเดียน เก็บในตัวแปร C
5. คำนวณความยาวของด้านที่สาม ด้วยสูตร $c = \sqrt{a^2 + b^2 - 2ab \cos(C)}$
6. แสดงความยาวด้านที่คำนวณได้ทางจอภาพในรูปแบบที่แสดงตามตัวอย่าง

ISBN

ISBN (International Standard Book Number) เป็นตัวเลขจำนวน 10-13 หลักที่ใช้ระบุหนังสือแต่ละเล่ม โจทย์ข้อนี้สนใจเฉพาะ ISBN ที่มี 10 หลัก การตรวจสอบความถูกต้องของ ISBN จะใช้ตัวเลขหลักสุดท้ายเป็น check digit ในการตรวจสอบความถูกต้องของตัวเลขอื่น ๆ โดยวิธีที่ใช้ตรวจสอบคือ

$$10n_1 + 9n_2 + 8n_3 + 7n_4 + 6n_5 + 5n_6 + 4n_7 + 3n_8 + 2n_9 + n_{10} \text{ จะต้องหารด้วย 11 ลงตัว}$$

ตัวอย่างเช่น หากตัวเลข 9 หลักแรกคือ 020131452 จะได้ว่า

$$10*0 + 9*2 + 8*0 + 7*1 + 6*3 + 5*1 + 4*4 + 3*5 + 2*2 + n_{10} = 83 + n_{10} \text{ จะต้องหารด้วย 11 ลงตัว}$$

จะได้ว่า n_{10} ต้องมีค่าเท่ากับ 5 เพื่อให้ผลรวมเป็น 88 ซึ่งหารด้วย 11 ลงตัว และได้ ISBN คือ 0201314525

หากกำหนดตัวเลขหลักที่ 1-9 มาให้ จงคำนวณหา ISBN ทั้งสิบหลัก

► ข้อมูลนำเข้า

มีบรรทัดเดียว ระบุ ISBN หลักที่ 1-9

► ข้อมูลส่งออก

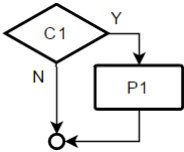
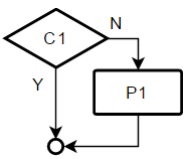
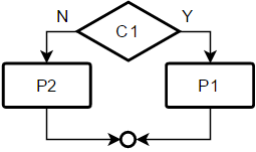
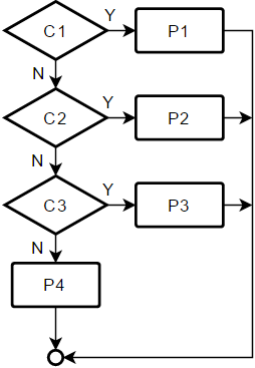
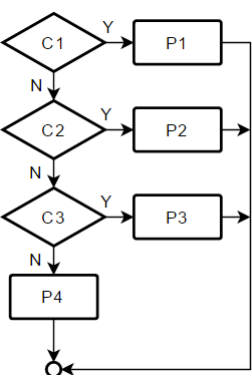
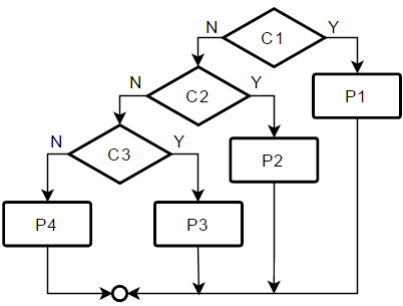
มีบรรทัดเดียว แสดง ISBN ทั้งสิบหลัก รับประกันว่ากรณีทดสอบจะไม่มีกรณีที่ n_{10} เท่ากับ 10

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
020131452	0201314525
100000000	1000000001

02 : Selection (if-elif-else)

สรุปเนื้อหา

Flowchart	Code
	<pre>if C1 : P1</pre>
	<pre>if C1 : pass else : P1</pre>
	<pre>if not C1 : P1</pre>
	<pre>if C1 : P1 else : P2</pre>
	<pre>if C1 : P1 elif C2 : P2 elif C3 : P3 else : P4</pre>
	

Flowchart	Code
<pre> graph TD Start(()) --> C1{C1} C1 -- Y --> P1[P1] C1 -- N --> C2{C2} C2 -- Y --> P2[P2] C2 -- N --> C3{C3} C3 -- Y --> P3[P3] C3 -- N --> P4[P4] P1 --> Join(()) P2 --> Join P3 --> Join P4 --> Join Join --> End(()) </pre>	<pre> if C1 : P1 if C2 : P2 if C3 : P3 else : P4 </pre>
<pre> graph TD Start(()) --> C1{C1} C1 -- Y --> C2{C2} C1 -- N --> P1[P1] C2 -- Y --> P2[P2] C2 -- N --> C3{C3} C3 -- Y --> P3[P3] C3 -- N --> P4[P4] P1 --> C4{C4} C4 -- Y --> P5[P5] C4 -- N --> C5{C5} C5 -- Y --> P6[P6] C5 -- N --> Join1(()) P5 --> Join1 P6 --> Join1 P4 --> Join2(()) Join1 --> Join2 Join2 --> End(()) </pre>	<pre> if C1 : if C2 : P2 if C3 : P3 else : P4 else : P1 if C4 : P5 elif C5 : P6 P7 </pre>

ตัวอย่างเงื่อนไขที่เขียนแทนกันได้

<code>not(x == 0)</code>	<code>x != 0</code>
<code>not(x==2 or x==4)</code>	<code>x!=2 and x!=4</code>
<code>not(x<2 and y>=4)</code>	<code>x>=2 or y<4</code>
<code>3<=x and x<9</code>	<code>3 <= x < 9</code>
<code>a < b and b < c and c < d and d <= e</code>	<code>a < b < c < d <= e</code>
<code>c=='a' or c=='e' or c=='i' or c=='o' or c=='u'</code>	<code>c in ('a', 'e', 'i', 'o', 'u')</code> หรือ <code>c in 'aeiou'</code>

<pre>if condition : t = value1 else : t = value2</pre>	<pre>t = value1 if condition else value2</pre> <p>ใช้เฉพาะกรณีการให้ค่ากับตัวแปร ถ้าเงื่อนไขเป็นจริงให้ค่าหนึ่ง เป็นเท็จให้อีกค่าหนึ่ง</p>
<pre>if s > a + b % 7 : t = True else : t = False</pre>	<pre>t = True if s > a+b%7 else False</pre> <p>หรือเขียนสั้น ๆ</p> <pre>t = (s > a + b % 7)</pre>

การเปรียบเทียบที่ใช้บ่อย

<code>if a%2 == 0 :</code>	<i>a เป็นเลขคู่หรือไม่</i>
<code>if a%100 == 0 :</code>	<i>aหารด้วย 100 ลงตัวหรือไม่</i>
<code>if (a//100)%10 == 9 :</code>	<i>เลขหลักร้อยของ a คือ 9 หรือไม่ หรือ</i>
<code>if (a%1000)//100 == 9:</code>	<i>ก็เหมือนกัน</i>
<code>if a <= x <= b :</code>	<i>x มีค่าในช่วงตั้งแต่ a ถึง b หรือไม่</i>
<code>if abs(a-b) <= max(abs(a),abs(b))*1e-10 :</code>	<i>ตรวจว่าจำนวนจริง a มีค่าใกล้กับ b หรือไม่ โดยตรวจว่า a กับ b ต่างกัน</i> <i>เชิงสัมพัทธ์ไม่เกิน 10⁻¹⁰ หรือไม่</i>
<code>mx = a</code>	
<code>if b > mx : mx = b</code>	
<code>if c > mx : mx = c</code>	
<code>if d > mx : mx = d</code>	<i>mx เก็บค่ามากที่สุดของ a,b,c และ d หรือเขียนโดยใช้ฟังก์ชัน max</i>
<code>mx = max(a,b,c,d)</code>	<i>max(a,b,c,d) หาค่ามากที่สุดของ a,b,c และ d</i>

เรื่องพิດบอย

ต้องการเปรียบเทียบความเท่ากัน แต่ใช้ =	<code>if x = 0 :</code>	ต้องเขียน <code>if x == 0 :</code>
ใช้ and กับ or ผิดความหมาย ทำให้ได้ค่าจริงหรือเท็จตลอด	<code>if x != 2 or x != 3 :</code> <code>if x <= 3 and x == 4 :</code>	แบบนี้ได้จริงตลอด แบบนี้ได้เท็จตลอด
เยื้องคำสั่งภายใน if หรือ else ไม่ตรงกัน	<code>if x > 0 :</code> <code> a = math.sqrt(x)</code> <code> print(a)</code>	ทุกบรรทัดใน if เยื้องบรรทัดไม่ตรงกัน ผิด
ใช้ tab ผสมกับ blank ในการเยื้องคำสั่ง	tab ผสมกับ blank ก็อาจดูว่าเยื้องตรงกัน แต่ผิด (IDLE จัดการเรื่อง tab กับ blank ให้ จึงไม่ผิด แต่ถ้าใช้ notepad จะผิด)	

<p>เข้าใจผิดเกี่ยวกับการเปรียบเทียบสตริง</p> <p>สตริงเปรียบเทียบกันตามลำดับแบบที่เขียนในพจนานุกรม โดย</p> <p>'0' < '9' < 'A' < 'Z' < 'a' < 'z'</p>	<pre>'abcdegh' < 'b' เป็นจริง '1234' < '9' เป็นจริง สมมติว่า print(x) ได้ 1234 print(y) ได้ 9 print(x < y) ได้ True ถ้า x = '1234', y = '9' print(x < y) ได้ False ถ้า x = 1234, y = 9</pre>
---	--

เรื่องที่ปรับปรุงได้

<p>คำสั่งที่เหมือนกันทั้งในกลุ่มหลัง if และกลุ่มหลัง else อาจแยกออกมาข้างนอกก็ได้</p>	<pre>if d > 0 : a = 9 c += d - 5 e = c else: a = 9 c -= d + 7 e = c</pre>	<pre>a = 9 if d > 0: c += d - 5 else: c -= d + 7 e = c</pre>
<p>การใช้ if-elif-else ที่ตรวจค่าว่าตกอยู่ในช่วงใด สามารถลดการเปรียบเทียบลงได้ ถ้าจัดลำดับการเปรียบเทียบให้เหมาะสม</p>	<pre>if s >= 80 : g = 'A' elif 70 <= s < 80 : g = 'B' elif 60 <= s < 70 : g = 'C' elif 50 <= s < 60 : g = 'D' else : g = 'F'</pre>	<pre>if s >= 80 : g = 'A' elif s >= 70 : g = 'B' elif s >= 60 : g = 'C' elif s >= 50 : g = 'D' else : g = 'F'</pre>

แบบฝึกหัด

Problem	Code
<p><u>Input</u>: รับจำนวนเต็ม 3 จำนวน คั่นด้วยช่องว่าง</p> <p><u>Process</u>: หามัธยฐานของจำนวนทั้ง 3</p> <p><u>Output</u>: มัธยฐานที่ได้</p>	
<p><u>Input</u>: รับข้อมูลของวงกลม 2 วง บรรทัดละหนึ่งวง ประกอบด้วยจำนวนจริง 3 จำนวน คั่นด้วยช่องว่าง แทน พิกัด x กับ y ของจุดศูนย์กลาง และรัศมีของวงกลม</p> <p><u>Process</u>: ตรวจสอบว่าวงกลมสองวงที่รับมาทับกันหรือแตะกันหรือไม่</p> <p><u>Output</u>: แสดงคำว่า touch เมื่อขอบของทั้งสองวงแตะกันพอดี แสดงคำว่า overlap เมื่อสองวงทับกัน ถ้าไม่แตะหรือทับ ให้แสดงคำว่า free</p>	
<p><u>Input</u>: รับจำนวนจริง 2 จำนวน คั่นด้วยช่องว่าง แทนพิกัด (x, y) บนระนาบสองมิติ</p> <p><u>Process</u>: ตรวจสอบว่าพิกัด (x, y) อยู่บริเวณใดในระนาบ</p> <p><u>Output</u>: ตำแหน่งของพิกัด (x, y) ว่า อยู่ในจุดภาคใด หรืออยู่บนแกน x หรือ y หรืออยู่ที่จุดกำเนิด</p>	
<p><u>Input</u>: รับจำนวนเต็ม 5 จำนวน คั่นด้วยช่องว่าง</p> <p><u>Process</u>: ตรวจสอบว่าลำดับจากซ้ายไปขวาของจำนวนที่รับมาเรียงจากน้อยไปมากหรือไม่</p> <p><u>Output</u>: ผลการตรวจสอบว่า True หรือ False</p>	

Problem	Code
<p><u>Input</u>: รับจำนวนเต็ม 4 จำนวน คั่นด้วยช่องว่าง</p> <p><u>Process</u>: หาผลรวมของจำนวนที่รับมา โดยไม่รวมจำนวนที่มากที่สุดหนึ่งจำนวน และจำนวนที่น้อยสุดหนึ่งจำนวน</p> <p><u>Output</u>: ผลรวมที่ได้</p>	
<p><u>Input</u>: รับจำนวนเต็มหนึ่งจำนวนเก็บในตัวแปร a</p> <p><u>Process</u>: ตรวจสอบว่ามีจำนวนเต็ม x ที่ค่า x^3 เท่ากับ a หรือไม่</p> <p><u>Output</u>: ถ้ามี แสดงค่าของ x ถ้าไม่มี แสดง Not Found</p>	
<p><u>Input</u>: รับจำนวนเต็มแทนรอบอก (หน่วยเป็นนิ้ว)</p> <p><u>Process</u>: หาขนาดของเสื้อยืดโปโลตามรอบอกดังนี้</p> <p>น้อยกว่า 37 นิ้ว ขนาด XS</p> <p>ตั้งแต่ 37 แต่ไม่ถึง 41 นิ้ว ขนาด S</p> <p>ตั้งแต่ 41 แต่ไม่ถึง 43 นิ้ว ขนาด M</p> <p>ตั้งแต่ 43 แต่ไม่ถึง 46 นิ้ว ขนาด L</p> <p>ตั้งแต่ 46 นิ้วเป็นต้นไป ขนาด XL</p> <p><u>Output</u>: ขนาดเสื้อโปโลตามรอบอกที่ได้รับ</p>	

ตัวอย่างการแก้โจทย์ปัญหา

สลากกินแบ่ง

หากเราซื้อสลากกินแบ่งเรียงหมายเลขตั้งแต่หมายเลข n1 ต่อเนื่องไปจนถึงหมายเลข n2 (เช่นหมายเลข 10300 ถึง 13999) และงวดนี้รางวัลที่ 1 คือหมายเลข p1 เลขท้ายสองตัวคือหมายเลข p2 และ เลขท้ายสามตัวคือหมายเลข p3 เราจะได้รางวัลรวมเป็นเงินเท่าไร

กำหนดให้สลากกินแบ่งที่ขายนี้เป็นรุ่นพิเศษ เป็นเลข 5 หลัก รางวัลที่หนึ่ง 10,000 บาท หนึ่งรางวัล รางวัลเลขท้ายสองตัวหนึ่งหมายเลข 25 บาท และรางวัลเลขท้ายสามตัวหนึ่งหมายเลข 100 บาท

► ข้อมูลนำเข้า

หนึ่งบรรทัดประกอบด้วยจำนวนเต็ม 5 จำนวน p1 p2 p3 n1 n2 คั่นด้วยช่องว่าง

► ข้อมูลส่งออก

เงินรางวัลรวมที่ได้รับ

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
01234 11 811 01000 01250	10075
99999 99 999 99950 99999	10125
19999 13 001 09015 13000	1275

ตัวอย่างการเขียนโปรแกรม

โปรแกรม	คำอธิบาย
<pre>p1,p2,p3,n1,n2 = \ [int(e) for e in input().split()] s = 0 if n1 <= p1 <= n2 : s += 10000 if n1 <= p2 <= n2 : s += 25 if n1 <= p3 <= n2 : s += 100 print(s)</pre>	<p>บรรทัดแรกรับข้อมูลใส่ตัวแปร p1, p2, p3, n1 และ n2 ให้ตัวแปร s เก็บเงินรางวัลรวม เริ่มด้วยการตรวจว่าหมายเลขของรางวัลที่ 1 อยู่ในช่วงหมายเลขที่ซื้อหรือไม่ (n1 <= p1 <= n2) ถ้าใช่ก็เพิ่มเงินรางวัล 10,000 บาท ตามด้วยการตรวจรางวัลเลขท้ายสองตัว แล้วก็สามตัว ในลักษณะเดียวกัน</p> <p>สั่ง run, ใส่ข้อมูลตามตัวอย่าง 01234 11 811 01000 01250, ได้ 10000 บาท ไม่ตรงตามตัวอย่าง ได้แค่รางวัลที่หนึ่ง เลขท้ายตรงไม่พบ</p>

โปรแกรม	คำอธิบาย
<pre> p1,p2,p3,n1,n2 = \ [int(e) for e in input().split()] s = 0 if n1 <= p1 <= n2 : s += 10000 print(s) </pre>	<p>การเขียนโปรแกรมที่มีหลาย ๆ กรณี ควรแยกทดสอบ หากเขียนรวบเดียว จะหาที่ผิดพลาดยาก ใจเย็น ๆ</p> <p>ขอเขียนและทดสอบกรณีรางวัลที่หนึ่งก่อน</p> <p>สั่ง run, ใส่ข้อมูล 12345 00 000 12000 13000 ได้ 10000 ถูกต้อง (ทดสอบกรณีอยู่ระหว่าง)</p> <p>สั่ง run, ใส่ข้อมูล 12345 00 000 12345 13000 ได้ 10000 ถูกต้อง (ทดสอบกรณีอยู่ที่ขอบล่าง)</p> <p>สั่ง run, ใส่ข้อมูล 12345 00 000 12000 12345 ได้ 10000 ถูกต้อง (ทดสอบกรณีอยู่ที่ขอบบน)</p> <p>สั่ง run, ใส่ข้อมูล 12345 00 000 12345 12345 ได้ 10000 ถูกต้อง (ทดสอบกรณีแค่ใบเดียวและถูกรางวัล)</p> <p>สั่ง run, ใส่ข้อมูล 12345 00 000 12346 14000 ได้ 0 ถูกต้อง (ทดสอบกรณีอยู่นอกช่วงทางซ้าย)</p> <p>สั่ง run, ใส่ข้อมูล 12345 00 000 12000 12344 ได้ 0 ถูกต้อง (ทดสอบกรณีอยู่นอกช่วงทางขวา)</p> <p>สรุปว่า กรณีรางวัลที่หนึ่ง ถูกต้อง</p>
<pre> p1,p2,p3,n1,n2 = \ [int(e) for e in input().split()] s = 0 # if n1 <= p1 <= n2 : # s += 10000 if n1%100 <= p2 <= n2%100 : s += 25 print(s) </pre>	<p>คราวนี้สนใจกรณีเลขท้ายสองตัว จะ comment คำสั่ง ตรวจสอบรางวัลที่หนึ่งออก ถ้ากลับไปดูโปรแกรมแรกที่เขียน คำสั่ง if n1 <= p2 <= n2 ตรวจสอบเลขท้ายไม่ครบทุกกรณี เช่นชื่อหมายเลข 10000 ถึง 10099 เลขท้าย p2 = 50 การทดสอบ n1 <= p2 <= n2 เป็นเท็จ แต่ความจริงแล้วถูกเลขท้ายสองตัว จึงต้องเปลี่ยนเป็นทดสอบเฉพาะสองหลักขวาเท่านั้น ด้วยคำสั่ง</p> <p>if n1%100 <= p2 <= n2%100</p> <p>สั่ง run, ใส่ 00000 50 000 10000 10099 ได้ 25 ถูกต้อง</p> <p>สั่ง run, ใส่ 00000 50 000 10000 10199 ได้ 25 ผิด น่าจะได้ 50</p> <p>สั่ง run, ใส่ 00000 50 000 10000 10299 ได้ 25 ผิด น่าจะได้ 75</p>

โปรแกรม	คำอธิบาย															
<pre>p1,p2,p3,n1,n2 = \ [int(e) for e in input().split()] s = 0 #if n1 <= p1 <= n2 : # s += 10000 if n1//100 == n2//100 : if n1%100 <= p2 <= n2%100 : s += 25 else: s += 25*(n2//100 - n1//100 + 1) print(s)</pre>	<p>คำสั่ง</p> <p>if n1%100 <= p2 <= n2%100 : s += 25</p> <p>ใช้ได้เฉพาะ กรณีที่ 3 หลักแรกของ n1 และ n2 เท่ากันเท่านั้น (เมื่อ n//100 มีค่าเท่ากับ n2//100)</p> <p>จึงขอจัดการเป็นสองกรณีคือ</p> <p>A. กรณี 3 หลักแรกเท่ากัน เช่น 10000 ถึง 10099 ทำเหมือนเดิม</p> <p>B. กรณี 3 หลักแรกไม่เท่ากัน เช่น 10000 ถึง 10299</p> <p>เมื่อนำ 102 - 100 = 2 แสดงว่า</p> <p>ถูกเลขท้ายสองตัว 2 หมายเลข ก็คูณด้วย 25</p> <p>สั่ง run, ใส่ 00000 50 000 10000 10199</p> <p>ได้ 50 ถูกต้อง</p> <p>สั่ง run, ใส่ 00000 50 000 10000 10299</p> <p>ได้ 75 ถูกต้อง</p> <p>สั่ง run, ใส่ 00000 50 000 10060 10680</p> <p>ได้ 175 ผิด น่าจะได้ 150</p>															
<pre>p1,p2,p3,n1,n2 = \ [int(e) for e in input().split()] s = 0 #if n1 <= p1 <= n2 : # s += 10000 if n1//100 == n2//100: if n1%100 <= p2 <= n2%100 : s += 25 else: if n1%100 <= p2 : # ช่วง 1 s += 25 if p2 <= n2%100 : # ช่วง 3 s += 25 s += 25*((n2//100-1) - \ # ช่วง 2 (n1//100+1) + 1) print(s)</pre>	<p>กรณีของ p2 = 50, n1 = 10060 และ n2 = 10680 การตรวจสอบน่าจะซับซ้อนเล็กน้อย แบ่งเป็นสามช่วง คือ</p> <table><tr><th>ช่วงที่ 1</th><th>ช่วงที่ 2</th><th>ช่วงที่ 3</th></tr><tr><td>10060 - 10099</td><td>10100 - 10599</td><td>10600 - 10680</td></tr><tr><td>3 หลักซ้ายเท่ากัน ดูเฉพาะ 2 หลักขวา</td><td>2 หลักขวา 00 ถึง 99 ถูกเลขท้ายแน่นอน ดูเฉพาะสามหลักซ้าย</td><td>3 หลักซ้ายเท่ากัน ดูเฉพาะ 2 หลักขวา</td></tr><tr><td>n1%100<=p2</td><td>ดูเฉพาะสามหลักซ้าย</td><td>p2<=n2%100</td></tr><tr><td>60<=50 เท็จ ไม่ถูกเลขท้าย</td><td>จาก 101 ถึง 105 ถูกรางวัล (105-101+1) ครั้ง</td><td>50<=80 จริง ถูก 1 ครั้ง</td></tr></table> <p>สั่ง run, ใส่ 00000 50 000 10000 10299</p> <p>ได้ 75 ถูกต้อง</p> <p>สั่ง run, ใส่ 00000 50 000 10060 10680</p> <p>ได้ 150 ถูกต้อง</p>	ช่วงที่ 1	ช่วงที่ 2	ช่วงที่ 3	10060 - 10099	10100 - 10599	10600 - 10680	3 หลักซ้ายเท่ากัน ดูเฉพาะ 2 หลักขวา	2 หลักขวา 00 ถึง 99 ถูกเลขท้ายแน่นอน ดูเฉพาะสามหลักซ้าย	3 หลักซ้ายเท่ากัน ดูเฉพาะ 2 หลักขวา	n1%100<=p2	ดูเฉพาะสามหลักซ้าย	p2<=n2%100	60<=50 เท็จ ไม่ถูกเลขท้าย	จาก 101 ถึง 105 ถูกรางวัล (105-101+1) ครั้ง	50<=80 จริง ถูก 1 ครั้ง
ช่วงที่ 1	ช่วงที่ 2	ช่วงที่ 3														
10060 - 10099	10100 - 10599	10600 - 10680														
3 หลักซ้ายเท่ากัน ดูเฉพาะ 2 หลักขวา	2 หลักขวา 00 ถึง 99 ถูกเลขท้ายแน่นอน ดูเฉพาะสามหลักซ้าย	3 หลักซ้ายเท่ากัน ดูเฉพาะ 2 หลักขวา														
n1%100<=p2	ดูเฉพาะสามหลักซ้าย	p2<=n2%100														
60<=50 เท็จ ไม่ถูกเลขท้าย	จาก 101 ถึง 105 ถูกรางวัล (105-101+1) ครั้ง	50<=80 จริง ถูก 1 ครั้ง														

โปรแกรม	คำอธิบาย
<pre> p1,p2,p3,n1,n2 = \ [int(e) for e in input().split()] s = 0 #if n1 <= p1 <= n2 : # s += 10000 if n1%100 <= p2 : #ช่วง 1 s += 25 if p2 <= n2%100 : #ช่วง 3 s += 25 s += 25*((n2//100-1) - \ #ช่วง 2 (n1//100+1) + 1) print(s) </pre>	<p>แต่ถ้าพิจารณาให้ละเอียด จะพบว่า คำสั่งที่พิจารณาทั้ง 3 ช่วงนั้น ครอบคลุมกรณี if $n//100 == n2//100$ ที่เราเขียนไว้ตอนต้น คือกรณีที่สามหลักซ้ายมีค่าเท่ากัน จึงลบทิ้งได้กลายเป็นโปรแกรมข้างซ้ายนี้</p> <p>สั่ง run, ใส่ 00000 50 000 10000 10299 ได้ 75 ถูกต้อง</p> <p>สั่ง run, ใส่ 00000 50 000 10060 10680 ได้ 150 ถูกต้อง</p>
<pre> p1,p2,p3,n1,n2 = \ [int(e) for e in input().split()] s = 0 if n1 <= p1 <= n2 : s += 10000 if n1%100 <= p2 : s += 25 if p2 <= n2%100 : s += 25 s += 25*(n2//100 - n1//100 - 1) if n1%1000 <= p3 : s += 100 if p3 <= n2%1000 : s += 100 s += 100*(n2//1000 - n1//1000 - 1) print(s) </pre>	<p>การตรวจเลขท้าย 3 ตัวก็คล้ายกับเลขท้าย 2 ตัว สามารถทำในลักษณะเดียวกัน โปรแกรมทางขวาเพิ่มการตรวจเงินรางวัล ทั้งรางวัลที่ 1 เลขท้าย 2 และ 3 ตัว คราวนี้ก็สั่งทดสอบการทำงานด้วยตัวอย่างที่โจทย์ให้มา</p> <p>สั่ง run, ใส่ 01234 11 811 01000 01250 ได้ 10075 ถูกต้อง</p> <p>สั่ง run, ใส่ 99999 99 999 99950 99999 ได้ 10125 ถูกต้อง</p> <p>สั่ง run, ใส่ 19999 13 001 09015 13000 ได้ 1275 ถูกต้อง</p>

ตัวอย่างโจทย์ปัญหา

Days in Month

ให้เขียนโปรแกรมเพื่อรับค่าเดือนและปีเป็นพุทธศักราช จากนั้นหาว่าในเดือนของปีนั้น จะมีจำนวนวันทั้งสิ้นกี่วัน

ตัวช่วย: เดือนกุมภาพันธ์มี 29 วัน ก็ต่อเมื่อ

(ปี ค.ศ.หารด้วย 4 ลงตัว แต่หารด้วย 100 ไม่ลงตัว) หรือ (ปี ค.ศ.หารด้วย 400 ลงตัว)

► ข้อมูลนำเข้า

มี 1 บรรทัด ประกอบด้วยจำนวนเต็ม 2 ตัว คือ เดือนและปีเป็นพุทธศักราช

► ข้อมูลส่งออก

จำนวนวันของเดือนและปีของข้อมูลนำเข้า

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
10 2557	31
2 2557	28
4 2556	30
2 2547	29

ตัดเกรด

การตัดเกรดของวิชานี้เป็นไปตามตารางด้านล่าง ให้เขียนโปรแกรมเพื่อตัดเกรดตามเกณฑ์ที่ระบุ โดยในกรณีที่คะแนนมีข้อผิดพลาด ให้แสดงผลว่า ERROR

คะแนนรวม (x)	เกรด
$80 \leq x \leq 100$	A
$75 \leq x < 80$	B+
$70 \leq x < 75$	B
$65 \leq x < 70$	C+
$60 \leq x < 65$	C
$55 \leq x < 60$	D+
$50 \leq x < 55$	D
$0 \leq x < 50$	F
กรณีอื่น ๆ	ERROR

► ข้อมูลนำเข้า

มีบรรทัดเดียว แทนคะแนนที่จะตัดเกรด เป็นจำนวนจริง

► ข้อมูลส่งออก

มีบรรทัดเดียว ระบุเกรดที่ได้รับ โดยในกรณีที่คะแนนมีข้อผิดพลาด ให้แสดงผลว่า ERROR

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
87.25	A
69.95	C+
120	ERROR

คิดค่าที่จอดรถ

ให้รับเวลาเข้าและออกของรถคันหนึ่ง (เปิดบริการตั้งแต่ 7:00 - 23:00) จากนั้นคำนวณค่าที่จอดรถที่ต้องจ่าย โดยหลักเกณฑ์การคำนวณมีดังนี้

1. จอดรถไม่เกิน 15 นาที ไม่คิดค่าบริการ
2. จอดรถเกิน 15 นาที แต่ไม่เกิน 3 ชั่วโมง คิดค่าบริการชั่วโมงละ 10 บาท เศษของชั่วโมงคิดเป็นหนึ่งชั่วโมง
3. จอดรถตั้งแต่ 4 ชั่วโมง ถึง 6 ชั่วโมง คิดค่าบริการชั่วโมงที่ 4-6 ชั่วโมงละ 20 บาท เศษของชั่วโมงคิดเป็นหนึ่งชั่วโมง
4. จอดรถเกิน 6 ชั่วโมงขึ้นไป เหม่าจ่ายวันละ 200 บาท

► ข้อมูลนำเข้า

มี 4 บรรทัด แต่ละบรรทัดมีจำนวนเต็มหนึ่งจำนวน

โดยบรรทัดที่ 1-2 เป็นชั่วโมงและนาทีของเวลาเข้า และบรรทัดที่ 3-4 เป็นชั่วโมงและนาทีของเวลาออก

► ข้อมูลส่งออก

มีบรรทัดเดียว เป็นค่าที่จอดรถที่ต้องจ่าย ให้แสดงผลลัพธ์เป็นจำนวนเต็ม

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
7 0 7 15	0
7 0 7 16	10
7 30 10 30	30
7 30 10 31	50
7 30 13 31	200

Issa โรจนะ
Intania 85

CEO & Co-Founder
Dek-D Interactive co., Ltd.

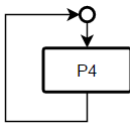
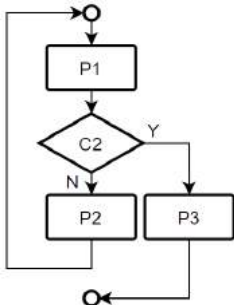
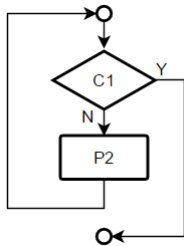
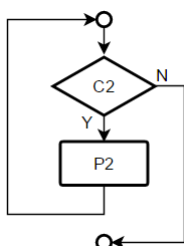


Programming จำเป็นมากในการเริ่มกิจการด้าน IT เพราะกิจการส่วนใหญ่ที่สามารถอยู่รอดได้ ผู้ก่อตั้งซึ่งเป็น
ผู้ที่เข้าใจผลิตภัณฑ์ที่จะสร้างมากที่สุด ต้องมีส่วนร่วมในการพัฒนาระบบเอง ถึงจะสามารถถ่ายทอดความคิด และ
ผลักดันผลิตภัณฑ์ที่ออกมาได้ เรียกว่ามีเงินมากแค่ไหน ถ้าต้องไปจ้างให้คนอื่นมาเขียนโปรแกรมแทนให้ยังงี้ก็สู้
คนที่มั่งมีทั้งใจเดียว และทักษะลงมือเขียนโปรแกรมเองได้ยาก

นอกจากนี้ **Programming** ยังเป็นการฝึกวิธีคิดแก้ปัญหาอย่างเป็นขั้นตอน ทำให้ผู้เรียนมีระบบการคิดซึ่งสามารถ
ประยุกต์ไปแก้ปัญหาได้ในหลายแขนงอย่างที่คุณเรียนคาดไม่ถึงอีกด้วย (เทพในหลาย ๆ วงการจบวิศวะฯ คอม)

03 : Repetition (while, for)

สรุปเนื้อหา

Flowchart	Code	
	<pre>while True : P4</pre>	
	<pre>while True : P1 if C2 : P3 break P2</pre>	
	<pre>while True : if C1 : break P2</pre>	
	<pre>while True : if not C2 : break P2</pre>	<pre>while C2 : P2</pre>

Flowchart	Code
<pre> graph TD Start(()) --> C1{C1} C1 -- N --> End1(()) C1 -- Y --> P1[P1] P1 --> C2{C2} C2 -- N --> P3[P3] P3 --> End2(()) C2 -- Y --> P2[P2] P2 --> C2 </pre>	<pre> while C1 : P1 while C2 : P2 P3 </pre>

ให้สังเกตว่า ภายในวงวน while ควรมีคำสั่งที่เปลี่ยนแปลงเงื่อนไขของ while ไม่งั้นมันจะวนทำงานไม่สิ้นสุด เช่น

```
while i < j :
```

```
...
```

```
i += 2
```

เงื่อนไข $i < j$ ข้างบนนี้จะเป็เท็จได้ (เพื่อให้ออกจากวงวน) ก็ด้วยการที่ค่าของ i เพิ่มขึ้น หรือค่าของ j ลดลง

คำสั่ง $i += 2$ ในตัวอย่างข้างบนสร้างความมั่นใจว่า วงวนนี้ทำงานแล้วจะมีจุดสิ้นสุดและออกจากวงวน

Flowchart	Code	Flowchart	Code
<pre>graph TD; A[k = 0] --> B(()); B --> C{k < n}; C -- N --> D(()); C -- Y --> E[P1]; E --> F[k += 1]; F --> B;</pre>	<pre>k = 0 while k < n : P1 k += 1</pre>	<pre>graph TD; A(()) --> B{k = 0 ... n-1}; B -- Done --> C(()); B -- Not Done --> D[P1]; D --> B;</pre>	<pre>for k in range(n) : P1</pre>
		<pre>graph TD; A(()) --> B{k = 0 ... m-1}; B -- Done --> C(()); B -- Not Done --> D{k = 0 ... n-1}; D -- Done --> B; D -- Not Done --> E[P1]; E --> D;</pre>	<pre>for p in range(m) : for k in range(n) : P1</pre>

Flowchart	Code
<pre> graph TD Start(()) --> Cond1{k = 0 ... m-1} Cond1 -- Done --> P4[P4] Cond1 -- Not Done --> P1[P1] P1 --> Cond2{C} Cond2 -- Y --> P3[P3] Cond2 -- N --> P2[P2] P3 --> Cond1 P2 --> Cond1 P4 --> End(()) </pre>	<pre> for k in range(m) : P1 if C : P3 break P2 else: P4 </pre> <p>จะมาทำหลัง else ของ for ก็เมื่อทำครบทุกรอบ หลังทำรอบที่ k = m-1 เสร็จ ก็มาทำที่ P4 ก่อนออกจากวงวน</p>
<pre> graph TD Start(()) --> Cond1{C1} Cond1 -- N --> P4[P4] Cond1 -- Y --> P1[P1] P1 --> Cond2{C2} Cond2 -- Y --> P3[P3] Cond2 -- N --> P2[P2] P3 --> Cond1 P2 --> Cond1 P4 --> End(()) </pre>	<pre> while C1 : P1 if C2 : P3 break P2 else: P4 </pre> <p>จะมาทำหลัง else ของ while ก็เมื่อทำงานเงื่อนไข C1 ของ while เป็นเท็จ ก็มาทำที่ P4 ก่อนออกจากวงวน</p>

range(start, stop, step)

- start, stop และ step ต้องเป็นจำนวนเต็ม
- for k in range(10) : k = 0, 1, 2, ..., 9
- for k in range(2,10) : k = 2, 3, 4, ..., 9
- for k in range(2,10,2) : k = 2, 4, 6, 8
- for k in range(10,1,-2) : k = 10, 8, 6, 4, 2
- for k in range(11,11) : ไม่ทำสักกรอบ เพราะ step เป็นบวก และ start >= stop
- for k in range(9,10,-1) : ไม่ทำสักกรอบ เพราะ step ติดลบ และ start <= stop

*** break จะย้ายการทำงานออกจากวงวนที่ break นั้นอยู่เท่านั้น

```

for i in range(5):
    for j in range(6):
        if condition1 :
            break          # break นี้ออกจาก for j
        if condition2 :
            break          # break นี้ออกจาก for i

```

วงวนที่พบบ่อย

<p>เมื่อต้องการทำอะไรบางอย่างซ้ำกัน n ครั้ง</p> <p>ใช้ <code>for k in range(n)</code></p>	<p>เช่น ต้องการอ่านข้อมูลจำนวน 10 ตัว เพื่อหาค่าเฉลี่ย</p> <pre>s = 0 for k in range(10) : # เป็นแค่คำสั่งควบคุมจำนวนรอบการทำซ้ำ a = float(input()) s += a print('average =', (s/10))</pre>
<p>เมื่อต้องการนำค่าที่สร้างจาก <code>range(start, stop, step)</code> มาใช้ในการประมวลผล</p> <p>ใช้ <code>for k in range(...)</code></p>	<p>เช่น ต้องการหาว่า q เป็นจำนวนเฉพาะหรือไม่ ใช้วงวน <code>for</code> หาว่าจำนวนเต็มมากกว่าหนึ่งขนาดเล็กสุดอะไร ที่หาร q ลงตัว</p> <pre>for k in range(2, q+1) : # k = 2,3,...,q if q % k == 0 : break # มีการนำ k มาใช้ if k == q : print(q,'is prime') else: print(q, '=', k, 'x', q//k)</pre>
<p>เมื่อต้องการประมวลผลชุดคำสั่งซ้ำ ๆ จนกว่าเงื่อนไขหนึ่งจะเป็นจริง</p> <p>ใช้ <code>while</code> หรือใช้ <code>if break</code> ในวงวน</p>	<p>เช่น ต้องการหาค่าเฉลี่ยจากชุดข้อมูลที่ผู้ใช้ป้อนเข้ามาเรื่อย ๆ จนกว่าจะรับจำนวนติดลบ</p> <div> <pre>s = 0 n = 0 while True : t = float(input()) if t < 0 : break s += t n += 1 if n == 0 : print('No Data') else : print('avg =',(s/n))</pre> <pre>s = 0 n = 0 t = float(input()) while t >= 0 : s += t n += 1 t = float(input()) if n == 0 : print('No Data') else : print('avg =',(s/n))</pre> </div>
<p>เมื่อต้องการแจกแจงวิธีการเลือกหมายเลขจากหมายเลข 0 ถึง n-1 จำนวน 2 หมายเลข แบบเลือกแล้วไม่เลือกอีก (คือแจกแจงการเลือกหมายเลขออกมาทีละคู่)</p> <pre>for i in range(n) : for j in range(i+1, n) : ได้ i<j ทุก ๆ กรณี</pre> <p>หรือถ้าต้องการให้ i = j ด้วย ก็เป็น</p> <pre>for i in range(n) : for j in range(i, n) : ได้ i≤j ทุก ๆ กรณี</pre>	<p>เช่น จงหาว่ามีจำนวนเต็ม x y และ z อะไรบ้างที่ทำให้สมการ $z^3 = x^2 + y^2$ เป็นจริง (x กับ y มีค่า 0 ถึง 19) เช่น $5^3 = 5^2 + 10^2$ แต่เราไม่ต้องการคำตอบ $5^3 = 10^2 + 5^2$ เพราะซ้ำ จึงต้องกำหนดว่า $x < y$ แต่ถ้าเราต้องการคำตอบ $8^3 = 16^2 + 16^2$ ด้วย ก็ต้องให้ $x \leq y$ โดยให้ y มีค่าเริ่มที่ x เป็นต้นไป ด้วย <code>for y in range(x,n)</code> ข้างล่างนี้</p> <pre>n = 20 for x in range(1,n) : for y in range(x,n): t = x**2 + y**2 z = int(round(t**(1/3),0)) # ทารากที่สามแล้ว # ปิดเคส if z**3 == t : print(z,x,y)</pre>

เรื่องพิศบอย

เข้าใจผิดเรื่องตัวสุดท้ายของ range	for k in range(1,5) k = 1,2,3,4 (ไม่รวม 5)
<p>ใช้วงวน ทำอะไรบางอย่าง เพื่อสรุปว่า เป็น A หรือ เป็น B โดยจะเป็น A เมื่อเงื่อนไข C เป็นจริง อย่างน้อยหนึ่งครั้ง แต่จะเป็น B เมื่อ C ต้องไม่เป็นจริงทุกครั้งทุกรอบ ถ้าเขียน</p> <pre>for k in range(...): if C: print('A') else: print('B')</pre> <p>แบบนี้ผิด เพราะสรุปว่าเป็น B เร็วไป ยังตรวจไม่ครบทุกรอบทุกกรณี แก้ไขด้วยการใช้ตัวแปรเก็บสถานะการตรวจ</p> <pre>found = False for k in range(...): if C: print('A') found = True break if not found: print('B')</pre> <p>หรือใช้ for-else ก็ง่ายกว่า</p> <pre>for k in range(...): if C: print('A') break else: print('B')</pre>	<p>เช่น ต้องการหาว่า q เป็นจำนวนเฉพาะหรือไม่</p> <pre>for k in range(2, q): if q % k == 0: print(q, 'is composite') # สรุปถูก เพราะ # หาตัวหารพบ else: print(q, 'is prime') # ผิด สรุปเร็วไป # ต้องวนทดสอบต่อ</pre> <p>แก้ไขโดยใช้ตัวแปรเก็บสถานะการตรวจ เป็นดังนี้</p> <pre>iscomposite = False for k in range(2,q): if q % k == 0: print(q, 'is composite') iscomposite = True break if not iscomposite: print(q, 'is prime')</pre> <p>หรือใช้ for-else ก็ง่ายกว่า</p> <pre>for k in range(2,q): if q % k == 0: print(q, 'is composite') break else: print(q, 'is prime')</pre>
วงวน ทำจำนวนรอบน้อยไปหรือมากไปกว่าที่ต้องการ (โดยทั่วไปมักขาดหรือเกินไปหนึ่งรอบ)	<p>เช่น จากชุดคำสั่งตรวจสอบจำนวนเฉพาะข้างบนนี้ ถ้าเขียน</p> <pre>for k in range(2,q+1): # เขียน q+1 แทนที่จะเป็น q if q % k == 0: print(q, 'is composite') break else: print(q, 'is prime')</pre> <p>แบบนี้เกินไปรอบ ทำให้ผลออกมาเป็น composite เสมอ เพราะอะไร ?</p> <p>หรือ อยากหาค่าของ $\sum_{k=1}^n k^3$ ถ้าเขียน</p> <pre>s = 0 for k in range(1,n): s += k**3 print(s)</pre> <p>ก็จะพบว่าขาดไปรอบ</p>

<p>ลืมนับค่าของตัวแปรที่ใช้ในเงื่อนไขของ while หรือไม่ก็ปรับค่าผิด ความผิดพลาดแบบนี้อาจทำให้วนทำงานไม่สิ้นสุด</p>	<p>เช่น ต้องการรับจำนวนจากผู้ใช้ มาหาค่าเฉลี่ยจนกว่าจะพบจำนวนลบ</p> <pre>s = n = 0 t = float(input()) while t >= 0 : s += t n += 1 print('avg =', (s/n))</pre> <p>ใช้ t ในการตรวจสอบเงื่อนไขของ while แต่ค่า t ไม่ได้เปลี่ยนแปลงเลยในวงวน ถ้าหลุดเข้ามาในวงวนได้จะเกิดอะไรขึ้น ?</p> <p>ควรแก้เป็น</p> <pre>s = n = 0 t = float(input()) while t >= 0 : s += t n += 1 t = float(input()) # เพิ่มบรรทัดนี้ print('avg =', (s/n))</pre>
<p>ตั้งค่าให้กับตัวแปรที่ควรจะให้ค่าก่อนเข้าวงวน แต่กลับไปเขียนในวงวน</p>	<p>เช่น ต้องการรับจำนวนจากผู้ใช้ มาหาค่าเฉลี่ยจนกว่าจะพบจำนวนลบ</p> <pre>while True : s = n = 0 # บรรทัดนี้ไม่น่ามาอยู่ในวงวน t = float(input()) if t < 0 : break s += t n += 1 print('avg =', (s/n))</pre> <p>จะเกิดอะไรขึ้น ?</p>

เรื่องแปลกของ for ใน Python

<p>หลังจากวนทำงานใน for k จนเรียบร้อยแล้ว ค่า k หลังออกจากวงวน จะมีค่าเท่ากับค่าสุดท้ายที่ทำงานในวงวน</p>	<pre>for k in range(1,5): ... print(k) # ได้ 4 เพราะเป็นค่าสุดท้ายที่ทำในวงวน for m in range(4,10): if m % 3 == 0 : break ... print(m) # ได้ 6 เพราะค่าสุดท้ายในวงวนคือ 6 ก่อนจะ break ออก for w in range(10,10): ... print(w) # ผิด เพราะไม่ได้เข้าไปทำในวงวน, w จึงไม่มีค่า</pre>
---	---

การปรับค่า k ภายในวงวน for k in range(...) จะไม่มีผลต่อการเปลี่ยนค่า k ในรอบถัดไป เพราะฉะนั้น ถ้าต้องการปรับค่าของ k ในวงวน ต้องใช้วงวน while แทน	for k in range(1, 4) : print(k) k += 2 print(k) # ได้ผลตามทางขวามือ ถ้าต้องการเปลี่ยน k ต้องใช้ while k = 1 while k < 4 : print(k) k += 2 print(k)	1 3 2 4 3 5 1 3 3 5
---	---	--



Problem	Code
Input: ไม่มี Process: หาจำนวนเต็มบวก k ที่มีค่าน้อยสุดที่ทำให้ มี $\left(\frac{1}{k}\right)k$ ค่าไม่เท่ากับ 1 (เนื่องจากความไม่แม่นยำ 100% ของจำนวนจริงในคอมพิวเตอร์) Output: จำนวนเต็ม k ที่ทำได้	
Input: ไม่มี Process: หาจำนวนเต็มบวก k ที่มีค่าน้อยสุดที่ทำให้ $1 - \left(\frac{365}{365}\right)\left(\frac{365-1}{365}\right)\left(\frac{365-2}{365}\right)\dots\left(\frac{365-k}{365}\right) \geq 0.5$ เป็นจริง Output: จำนวนเต็ม k ที่ทำได้	
Input: ไม่มี Process: คำนวณค่าประมาณของ π จากสูตร $4\left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \dots + \frac{1}{399997} - \frac{1}{399999}\right)$ Output: ค่าประมาณของ π ที่ทำได้	
Input: อ่านจำนวนเต็ม 2 จำนวน a กับ b Process: คำนวณค่าจากสูตร $\sum_{i=a}^{b-1} \left((-1)^i \sum_{j=i+1}^b (i+j) \right)$ Output: ค่าที่คำนวณได้	
Input: อ่านจำนวนเต็ม 2 จำนวน a กับ b Process: คำนวณค่าจากสูตร $\sum_{a \leq i < j \leq b} (-1)^i (i+j)$ Output: ค่าที่คำนวณได้	

Problem	Code
<p><u>Input</u>: บรรทัดแรกรับจำนวนเต็มเก็บในตัวแปร n (n จะมีค่ามากกว่า 0) และอีก n บรรทัด เป็นจำนวนเต็ม บรรทัดละจำนวน</p> <p><u>Process</u>: หาผลต่างของค่ามากที่สุดกับค่าน้อยสุด และหาว่ามีกี่จำนวนที่เป็นเลขลบ</p> <p><u>Output</u>: ผลต่าง และจำนวนเลขลบที่หาได้</p>	
<p><u>Input</u>: จำนวนเต็มเก็บในตัวแปร n</p> <p><u>Process</u>: หาจำนวนเต็มบวก w, x, y และ z ทั้งหมดที่ $w^3 = x^2 + y^2 + z^2$ โดยที่ $1 \leq x \leq y \leq z \leq n$</p> <p><u>Output</u>: ค่าของ w, x, y และ z ที่หาได้</p>	

ตัวอย่างการแก้โจทย์ปัญหา

Approximation of sine

ค่าของ $\sin(x)$ คำนวณได้ด้วยอนุกรมเทย์เลอร์ ดังแสดงข้างล่างนี้

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots = \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k+1}}{(2k+1)!}$$

จึงเขียนโปรแกรมรับค่าของ x เพื่อคำนวณค่า $\sin(x)$ ให้ได้ความแม่นยำมากที่สุดเท่าที่จะทำได้ ด้วยสูตรข้างบนนี้

► ข้อมูลนำเข้า

หนึ่งบรรทัดประกอบด้วยจำนวนจริง x (หน่วยเป็นองศา)

► ข้อมูลส่งออก

ค่าประมาณของ $\sin(x)$ จากสูตรข้างต้น

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
45	0.7071067811865475
36045	0.7071067811865475

ตัวอย่างการเขียนโปรแกรม

โปรแกรม	คำอธิบาย
<pre>import math x = math.radians(float(input())) n = 30 s = 0 for k in range(n+1): # k=0,1,...,n s+=(-1)**k*x**(2*k+1) \ /math.factorial(2*k+1) print(s)</pre>	<p>เริ่มด้วยการรับข้อมูล, แปลงเป็น float, แล้วเปลี่ยนเป็นเรเดียน สูตรการคำนวณมีลักษณะเป็นการบวกซ้ำ ๆ กันหลาย ๆ พจน์ที่ใช้ค่าของ k ที่เริ่มจาก 0 เพิ่มขึ้นทีละ 1 ไปเรื่อย ๆ ทำให้คิดถึงการใช้วงวน for ปัญหาคือจะต้องหมุนคำนวณกี่รอบ ก็พจน์ จึงจะละเอียดสุด ๆ ตามที่ โจทย์ต้องการ</p> $\sin(x) = \sum_{k=0}^n (-1)^k \frac{x^{2k+1}}{(2k+1)!}$ <p>ทดลองตั้งให้ n = 30 มีตัวแปร s เก็บผลลัพธ์ คำนวณค่าของพจน์ ตรงไปตรงมาตามสูตร ซึ่งมีการยกกำลังด้วย ** และการหาค่า แยกต่อเรียลด้วย math.factorial</p> <p>สั่ง run, ใส่ 45 เป็นข้อมูล, ได้ผลเป็น 0.7071067811865475 ถูกต้อง</p>
<pre>import math x = math.radians(float(input())) n = 100 s = 0 for k in range(n+1): # k=0,1,...,n s+=(-1)**k*x**(2*k+1) \ /math.factorial(2*k+1) print(s)</pre>	<p>อยากให้ละเอียดกว่านี้ ลองตั้ง n = 100 ดู</p> <p>สั่ง run, ใส่ 45 เป็นข้อมูล, ได้ผลผิดที่บรรทัดที่มีการคำนวณ</p> <p>OverflowError: int too large to convert to float</p> <p>แปลว่า มีการแปลงจำนวนเต็มเป็น float แต่จำนวนเต็มมีขนาดใหญ่ เกินที่ float จะรับได้ แล้วจะรู้ได้อย่างไรว่า ผิดตรงไหน</p> <p>หาก run ด้วย IDLE เมื่อเกิดข้อผิดพลาดแล้ว เราสามารถขอดูตัวแปร ต่าง ๆ ได้ เช่น อยากรู้ค่า k เมื่อเกิดความผิดพลาด ก็ใส่คำสั่ง print(k) ที่ Python shell จะแสดงว่า ผิดตอนที่ k มีค่าเป็น 85 ก็ลองคำนวณ ค่าต่าง ๆ ในสูตร ดูว่าผิดที่ใด</p> <pre>>>> print(k) 85 >>> print((-1)**k) -1 >>> print(x**(2*k+1)) 1.1491314574538792e-18 >>> print(math.factorial(2*k+1)) 12410180702176678234248405241031039926166055775 01693185388951803611996075221691752992751978120 48758557646495950167038705280988985869071076733 12420322184843643104735778899685482782907545415 61964852153468318044293239598173696899657235903 94761615227855818006117636510842880000000000000 000 >>></pre> <p>ก็ไม่มีอะไรผิด แต่จากที่บอกว่าผิดเพราะแปลงจำนวนเต็มที่ใหญ่เกินไป เป็น float ทำให้วิเคราะห์ได้ว่า ผลของการหาค่าแฟกตอเรียลต้องแปลง เป็น float เพื่อไปเป็นตัวหารในสูตร หากลองคำสั่งนี้ที่ Python shell</p> <pre>>>> print(1/math.factorial(2*k+1))</pre> <p>ก็พบว่าผิดด้วยเหตุผลเดียวกัน</p>

โปรแกรม	คำอธิบาย
<pre>import math x = math.radians(float(input())) n = 100 s = f = x for k in range(1,n+1): f *= (-1)*x**2/((2*k)*(2*k+1)) s += f print(s)</pre>	<p>แก้ปัญหาโดยไม่ใช้จำนวนเต็มขนาดใหญ่ แต่มองว่า การคำนวณหา พจน์ในแต่ละรอบนั้น สามารถคำนวณได้จากพจน์ในรอบก่อนได้ ไม่จำเป็นต้องคำนวณ $(-1)^k$, x^{2k+1} และ $(2k+1)!$ ทุกรอบ แต่สามารถ คำนวณจากความสัมพันธ์ดังนี้</p> $\begin{aligned} (-1)^k &= (-1)((-1)^{(k-1)}) \\ x^{2k+1} &= x^2(x^{2(k-1)+1}) \\ (2k+1)! &= (2k+1)(2k)((2(k-1)+1)!) \end{aligned}$ <p>ดังนั้น</p> $\underbrace{(-1)^k}_{\text{พจน์ในรอบที่ } k} \underbrace{\frac{x^{2k+1}}{(2k+1)(2k)}}_{\text{ตัวคูณ}} \underbrace{\left((-1)^{(k-1)} \frac{x^{2(k-1)+1}}{(2(k-1)+1)!} \right)}_{\text{พจน์ในรอบที่ } k-1}$ <p>เริ่มด้วย $f = x$ (คือพจน์ที่ $k = 0$)</p> $\begin{aligned} k=1 \quad f &= (-1)*x**2/((2*k+1)*(2*k)) * f \\ &= -(x**2)/(3*2) * x \\ &= -(x**3)/(3!) \\ k=2 \quad f &= -(x**2)/(5*4) * -(x**3)/(3!) \\ &= +(x**5)/(5!) \\ k=3 \quad f &= -(x**2)/(7*6) * (x**5)/(5!) \\ &= -(x**7)/(7!) \\ &\dots \end{aligned}$ <p>ด้วยแนวคิดนี้เขียนโปรแกรมได้ดัง code ทางซ้าย</p> <p>สั่ง run, ใส่ 45 เป็นข้อมูล, ได้ผลเป็น 0.7071067811865475 ถูกต้อง</p> <p>ถ้าลองเพิ่มคำสั่ง <code>print(k, s)</code> ในวงวน <code>for</code> จะพบว่า ค่า <code>s</code> มีค่าไม่เปลี่ยนแปลงตั้งแต่ $k = 8$ เป็นต้นไป แสดงว่าการคำนวณ ตั้งแต่รอบที่ 8 เป็นต้นไป ไม่มีประโยชน์เลย แต่ถ้าใส่ $x = 180$ เป็นข้อมูลนำเข้า พบว่า $k = 22$ เป็นต้นไปจะไม่เปลี่ยนแปลง</p>
<pre>import math x = math.radians(float(input())) s = f = x k = 1 while f != 0: f *= (-1)*x**2/((2*k)*(2*k+1)) s += f k += 1 print(s)</pre>	<p>จึงควรปรับการทำงานของโปรแกรมให้ตรวจสอบว่า หากค่าในตัวแปร <code>s</code> ไม่เปลี่ยนแปลง หรือใช้วิธีตรวจสอบว่า เมื่อ <code>f</code> เป็น 0 ก็สามารถหยุดการคำนวณได้ โดยเปลี่ยนจากการใช้</p> <p>วงวน <code>for</code> เป็นวงวน <code>while</code> จะเหมาะสมกว่า</p> <p>โดยมีเงื่อนไข <code>while f != 0</code> เป็นตัวกำหนดว่า ต้องทำต่อในวงวน เมื่อ <code>f</code> ยังไม่เท่ากับ 0 ตัวแปร <code>k</code> เริ่มต้นที่ 1 ในรอบแรก และเพิ่มทีละ 1 ในรอบถัด ๆ ไป</p> <p>สั่ง run, ใส่ 45 เป็นข้อมูล, ได้ผลเป็น 0.7071067811865475 ถูกต้อง</p> <p>และถ้าลอง <code>print(k)</code> ดูใน Python shell จะพบว่า $k = 8$ เมื่อออกจากวงวน ด้วยวิธีนี้เราไม่ต้องกำหนดจำนวนรอบให้กับวงวน มาลองตัวอย่างที่ 2 สั่ง run, ใส่ 36045 เป็นข้อมูล, ได้ผลเป็น 2.541635699930208e+252 ผิด เกิดอะไรขึ้น ?</p>

โปรแกรม	คำอธิบาย
<pre>import math x = float(input()) x = math.radians(x%360) s = f = x k = 1 while f != 0: f *= (-1)*x**2/((2*k)*(2*k+1)) s += f k += 1 print(s)</pre>	<p>ค่า 36045 เมื่อแปลงเป็นเรเดียนแล้ว x มีค่ามาก การคำนวณค่า f ในวงวนจะลดความแม่นยำลงด้วยเหตุที่จำนวนจุดหลังทศนิยมของ float มีจำนวนจำกัด ทำให้การคำนวณผิดพลาด ด้วยคุณสมบัติของฟังก์ชัน sin ที่เป็นเชิงคาบ เราควรลดขนาดของค่า x ลงด้วยการเปลี่ยนค่า 36045 องศาเป็น $36045 \% 360 = 45$ องศา ก่อนเปลี่ยนเป็นเรเดียน จะเพิ่มความแม่นยำและลดความผิดพลาดในการคำนวณ ได้ผลดัง code ทางซ้าย</p>



Multiples of 3 or 5

จงเขียนโปรแกรมที่คำนวณหาผลรวมของจำนวนเต็มบวกทุกจำนวนที่มีค่าต่ำกว่าจำนวนที่เป็นข้อมูลนำเข้าและมี 3 หรือ 5 เป็นตัวประกอบ เช่น หากข้อมูลนำเข้าคือ 20 คำตอบที่เราต้องการจะเท่ากับ $3 + 5 + 6 + 9 + 10 + 12 + 15 + 18 = 78$ (สังเกตว่าคำตอบของเราไม่รวมค่า 20 เนื่องจากเราสนใจเฉพาะจำนวนที่มีค่าต่ำกว่า 20)

► ข้อมูลนำเข้า

มีบรรทัดเดียว เป็นจำนวนเต็มบวก

► ข้อมูลส่งออก

มีบรรทัดเดียว แสดงผลรวมของจำนวนเต็มบวกทุกจำนวนที่มีค่าต่ำกว่าจำนวนที่เป็นข้อมูลนำเข้าและมี 3 หรือ 5 เป็นตัวประกอบ

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
20	78
25	143
3	0

Average until -1

ให้อ่านข้อมูลจากแป้นพิมพ์ที่เป็นจำนวนจริงจนกว่าจะพบค่า -1 เพื่อหาค่าเฉลี่ยของจำนวนเหล่านั้นทั้งหมด (ไม่รวม -1)

► ข้อมูลนำเข้า

จำนวนจริงบรรทัดละ 1 จำนวนหลายบรรทัด บรรทัดสุดท้ายคือ -1

► ข้อมูลส่งออก

มีบรรทัดเดียว แสดงค่าเฉลี่ยของจำนวนทั้งหมด (ไม่รวม -1) ออกทางหน้าจอ

ในกรณีที่ไม่มีจำนวนข้อมูลเป็น 0 ให้แสดง No Data

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
2295.498850599365 8502.139421733784 515.0100470901091 3705.6829835190097 8722.343211974356 4446.712951812571 6375.086965715 3801.511489785674 7638.911577747659 -1	5111.43305555306
-1	No Data

สามเหลี่ยมมุมฉาก

โจทย์นี้สนใจเฉพาะสามเหลี่ยมมุมฉากที่มีความยาวด้านทุกด้านเป็นจำนวนเต็ม

จงเขียนโปรแกรมอ่านค่าความยาวเส้นรอบรูปจากแป้นพิมพ์ เพื่อหาจำนวนเต็มมากที่สุดที่เป็นความยาวของด้านตรงข้ามมุมฉากของสามเหลี่ยมมุมฉากที่มีความยาวเส้นรอบรูปตามที่ได้รับ เช่น ให้เส้นรอบรูปของสามเหลี่ยมยาว 90 จะมีสามเหลี่ยมมุมฉากตามข้อกำหนดอยู่สองรูปคือ 15, 36, 39 และ 9, 40, 41 คำตอบที่ต้องการคือ 41 เพราะเป็นความยาวด้านตรงข้ามมุมฉากที่ยาวสุดของสามเหลี่ยมมุมฉากตามข้อกำหนด และมีเส้นรอบรูปยาว 90

► ข้อมูลนำเข้า

มีบรรทัดเดียวเป็นจำนวนเต็มบวก แทนความยาวเส้นรอบรูปของสามเหลี่ยมมุมฉากตามข้อกำหนด (รับประกันว่า มีสามเหลี่ยมมุมฉากที่มีความยาวเส้นรอบรูปเท่ากับจำนวนที่เป็นข้อมูลนำเข้า)

► ข้อมูลส่งออก

มีบรรทัดเดียวแสดงความยาวของด้านตรงข้ามมุมฉากที่ยาวที่สุดของสามเหลี่ยมมุมฉากตามข้อกำหนด

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
30	13
90	41

สนาวัฒน์ มาลาบุปผา

Intania 84

CEO & Co-Founder of Priceza



Programming เป็นหนึ่งในวิชาที่ผมชอบเรียนมากที่สุดวิชาหนึ่งเลยตอนปี 1 เหตุผลเพราะว่ามันช่วยให้ผมมีกระบวนการคิดและการแก้ปัญหาต่าง ๆ อย่างเป็นระบบ จนมาถึงตอนที่ยากทำ Startup สร้าง Priceza ขึ้นมาด้วยความเข้าใจใน **Programming** ทำให้ผมมีพื้นฐานวิธีคิดในการศึกษาพัฒนาระบบต่าง ๆ ที่เราไม่เคยพัฒนาขึ้นมาก่อน และทำให้เราศึกษาและวิจัยตั้งแต่ต้น จนพัฒนามันขึ้นมาได้ในที่สุด



ลิสา ตรงประกอบ

Intania 81

Google Inc.

Programming ทำให้เรารู้จักคิดและรู้จักสื่อสาร และไม่ใช่การคิดและการสื่อสารแบบทั่ว ๆ ไป แต่เป็นการคิดอย่างเป็นระบบ คิดอย่างรอบคอบ คิดในทุก ๆ ความเป็นไปได้ และเป็นการสื่อสารที่ต้องชัดเจนที่ทั้งผู้ส่งสารและผู้รับสารต้องเข้าใจไปในทางเดียวกัน ถ้ามีอะไรผิดพลาดแม้เพียงนิดเดียว ลืมคิดไปนิดนึง สื่อสารผิดไปเล็กน้อย ผลก็จะออกมาไม่ได้อย่างที่วางแผนไว้ การคิดและการสื่อสารแบบนี้สามารถนำไปใช้กับอะไรก็ได้ และสามารถสร้างผลลัพธ์เป็นอะไรก็ได้ตามแต่ที่ใจเราอยากให้เป็น

สรุปเนื้อหา

สตริง (string) เก็บอักขระ (character) ตั้งแต่ศูนย์ตัวขึ้นไป เรียงจากซ้ายไปขวาต่อกันไป แต่ละตัวมีเลข index ระบุตำแหน่ง โดย index ของตัวซ้ายสุดคือ 0 อักขระแต่ละตัวในสตริงเป็นได้ทั้งตัวอักษร ตัวเลข และสัญลักษณ์พิเศษต่าง ๆ การเขียนสตริงทำได้หลายแบบดังนี้

- เขียนครอบด้วย อัฒประภาคเดี่ยว `'I am "Python".'` `'I\'m Python.'`
- เขียนครอบด้วย อัฒประภาคคู่ `"I'm Python."` `"I am \"Python\"."`
- เขียนครอบด้วย อัฒประภาคเดี่ยวสามตัวติด `'''I'm "Python".'''`
- เขียนครอบด้วย อัฒประภาคคู่สามตัวติด `"""I'm "Python".""""`

หมายเหตุ : 12 เป็น int ไม่ใช่สตริง แต่ '12' คือสตริง ถ้าต้องการแปลงจำนวนในตัวแปร x ให้กลายเป็นสตริง ใช้ `str(x)`

ตัวอย่างการเข้าใช้อักขระและสตริงย่อยในสตริง (สมมติให้ `s = "ABCDEFGH"`)

- `len(t)` ได้จำนวนตัวอักขระใน t โดย `len('')` ได้ 0
- `s[0]` เหมือน `s[-len(s)]` ได้ "A" ส่วน `s[-1]` เหมือน `s[len(s)-1]` ได้ "H"
- อย่าลืมว่า index ของสตริง s ต้องอยู่ในช่วง 0 ถึง `len(s)-1` จากซ้ายไปขวา และ -1 ถึง `-len(s)` ถอยจากขวามาซ้าย ดังนั้นเราเขียน `s[k]` ได้ โดยที่ $-\text{len}(s) \leq k \leq (\text{len}(s)-1)$ เพราะฉะนั้น `"01234"[-6]` กับ `"01234"[5]` ผิด
- `s` เหมือน `s[:]` เหมือน `s[0:]` เหมือน `s[:len(s)]` เหมือน `s[::]` เหมือน `s[::1]`
- `s[::2]` หยิบตัวที่ index คู่ได้ "ACEG", `s[1::2]` หยิบตัวที่ index คี่ได้ "BDFH"
- `s[::-1]` เหมือน `s[-1::-1]` เหมือน `s[-1:-(-len(s)+1):-1]` ได้ "HGFEDCBA"
- ถ้าเขียน `s[a:b]` เพื่อเลือกสตริงย่อยออกมา ค่า a กับ b เป็นอะไรก็ได้ ไม่ผิด
 - `"01234"[2:50000]` ได้ "234", `"01234"[4999:50000]` ได้ ""
 - `"01234"[-500:-2]` ได้ "012", `"01234"[-3:-500:-1]` ได้ "210", `"01234"[-500:-300]` ได้ ""
- ใช้ `for c in s` : เพื่อแจกแจงอักขระทีละตัวใน s จากซ้ายไปขวาเก็บในตัวแปร c นำไปใช้ในวงวนได้

ตัวอย่างการจัดการสตริง (ให้ `s = " Python 3.6 "`)

- `t = s.upper()` ได้ t เก็บ " PYTHON 3.6 " s เหมือนเดิม
- `t = s.lower()` ได้ t เก็บ " python 3.6 " s เหมือนเดิม
- `t = s.strip()` ได้ t เก็บ "Python 3.6" s เหมือนเดิม
- `s = s.strip().upper()` ได้ s เก็บ "PYTHON 3.6"
- `k = s.find(c)` คืน index น้อยสุดที่พบ c ใน s เริ่มค้นตั้งแต่ index 0 ถ้าไม่พบ จะได้ผลเป็น -1
`k = "engineering".find("ng")` ได้ k เก็บ 1 เพราะ "ng" ปรากฏเริ่มที่ index 1 ใน "engineering"
คำสั่ง `if c in s` ก็เหมือนกับ `if s.find(c) >= 0`
- `k = s.find(c,j)` คืน index น้อยสุดที่พบ c ใน s เริ่มค้นตั้งแต่ index j เป็นต้นไป
- นำสตริงบวกกัน คือนำสตริงมาต่อกัน เช่น `'12'+ '23'` คือ '1223'
- สตริงคูณกับจำนวนเต็ม คือนำสตริงนั้นมาต่อกันเป็นจำนวนครั้งเท่ากับค่าของจำนวนเต็มนั้น เช่น `'12'*3` คือ '121212'

ตัวอย่างการจัดการสตริง

ทำ <code>s[2] = 'a'</code> ไม่ได้ แต่สร้างใหม่ได้	<code>s = s[:2] + 'a' + s[3:]</code>
ตรวจว่าตัวแปร <code>c</code> เก็บตัวอักษร ตัวเดียวและเป็นสระในภาษาอังกฤษ หรือไม่	<pre>if len(c) == 1 and c.lower() in 'aeiou' : # c contains a vowel ...</pre>
ตรวจว่าตัวแปร <code>c</code> เก็บตัวอักษร ตัวเดียวและเป็นตัวอักษรภาษาอังกฤษ หรือไม่	<pre>if len(c) == 1 and 'a' <= c.lower() <= 'z' : # c contains an English alphabet ...</pre>

รูปแบบการประมวลผลสตริงที่พบบ่อย

ต้องการหิบบัอักขระในสตริงจาก ซ้ายไปขวามาประมวลผลทีละตัว	ต้องการนับว่าสตริง <code>s</code> มีตัวเลขกี่ตัว <pre>c = 0 for e in s : if '0' <= e <= '9' : c += 1</pre>
ต้องการหิบบัอักขระที่ละตัว พร้อมกับ index ของตัวนั้น ๆ	<pre>for i in range(len(s)): c = s[i] print(i,c)</pre> <p>หรือ</p> <pre>for i,c in enumerate(s): print(i,c)</pre>
ต้องการหิบบัอักขระในสตริงจาก ขวามาซ้ายทีละตัว	<pre>for e in s[::-1] : ...</pre> <p>หรือ</p> <pre>for k in range(-1, -(len(s)+1), -1): e = s[k] ...</pre>
ต้องการหิบบัอักขระในสตริงจาก ซ้ายไปขวามาประมวลผลทีละคู่ที่ ติดกัน	ต้องการนับว่าสตริง <code>s</code> มีตัวที่ติดกันเป็นตัวเลขทั้งคู่อยู่กี่คู่ <pre>c = 0 for k in range(len(s)-1) : # ต้องการตัวติดกัน จึงวนถึงตัวรองสุดท้าย if '0' <= s[k] <= '9' and '0' <= s[k+1] <= '9' : c += 1</pre>

<p>ถ้าต้องการนับว่ามี t ปรากฏอยู่ใน s ก็คือ</p> <pre>c = 0 while t in s : c += 1</pre> <p>แบบนี้ผิด ถ้า มี t ใน s เงื่อนไข t in s จะเป็น True ตลอด ก็จะไม่รู้จบ</p>	<pre>c = 0 k = s.find(t) while k >= 0 : c += 1 k = s.find(t,k+1) # 'aaaaaa' มี 'aaa' 4 ครั้ง</pre> <p>หรือ</p> <pre>c = 0 for i in range(len(s)-len(t)+1) : if s[i:i+len(t)] == t : c += 1</pre>
<p>ใช้สตริงสะสมข้อมูลเพื่อนำมาแสดงที่หลัง</p>	<p>หาจำนวนเฉพาะที่มีค่าน้อยกว่า 30</p> <pre>result = "" for n in range(2,30): for k in range(2, n) : if n % k == 0 : break else: result += str(n) + ", "</pre> <p>print(result[:-2]) ได้ 2, 3, 5, 7, 11, 13, 17, 19, 23, 29</p>
<p>ต้องการหีบสตริงย่อยที่อยู่ระหว่างรูปแบบสตริงที่กำหนด</p>	<p>หาสตริงย่อยที่อยู่ระหว่าง "<title>" และ "</title>" (ให้ถือว่า ถ้าสตริงที่ได้รับมี <title> ก็ต้องมี </title> ด้วย)</p> <pre>t = input().strip() a = t.find("<title>") if a >= 0 : j = a+len("<title>") # a j b b = t.find("</title>",j) #..<title>.....</title> print(t[j:b]) else : print('No title')</pre>
<p>ใช้สตริงเป็นที่เก็บข้อมูลเพื่อแปลงจากจำนวนเต็มเป็นสตริง</p>	<p>แปลงเลขเดือนใน m เป็นชื่อย่อเดือน</p> <pre>months = "JANFEBMARAPR MAYJUNJUL AUGSEP OCTNOV DEC" m = int(input()) if 1 <= m <= 12 : print(months[(m-1)*3:m*3]) else : print('invalid month number')</pre>
<p>ใช้สตริงสองสายเป็นที่เก็บข้อมูลเพื่อการแปลงข้อมูลจากสตริงหนึ่งเป็นอีกสตริงหนึ่ง</p>	<p>หาชื่อย่อสกุลเงิน (3 ตัวอักษร) จากชื่อย่อประเทศ (2 ตัวอักษร)</p> <pre>c = input().strip().lower() # country code # USA, Thailand, Japan, China, Singapore countries = "usthjp cnsg" currency_codes = "USDTHBJPYCNYSGD" k = countries.find(c) if k >= 0: k //= 2 print(currency_codes[3*k:3*(k+1)]) else : print('Not found')</pre>

เรื่องพิศบอย

เครื่องหมายเปิดปิดสตริงไม่ครบ ไม่ตรงกัน หรือใส่ซ้อนภายในโดย ไม่นำหน้าด้วย \	<pre>s = "String s = String' s = "string" s = "You shouted "Hey!" very loudly yesterday"</pre>
ถ้าต้องการรับสตริงจาก แป้นพิมพ์ ควร strip ด้วย เพราะผู้ใช้อาจเผลอเพิ่มช่องว่าง ทางซ้ายหรือขวาเพิ่มเติม	<p>คำสั่งข้างล่างนี้ ถ้าผู้ใช้ป้อน ok ตามด้วยช่องว่างแล้วกด enter การเปรียบเทียบก็จะไม่เท่า 'ok'</p> <pre>s = input() if s == 'ok' : print('OK')</pre> <p>จึงควรเขียน s = input().strip() (ยกเว้นกรณีที่เราต้องการคงทุกสิ่งที่ผู้ใช้ป้อนทั้งหมด)</p>
ใช้ index ที่เกินช่วงที่ใช้ได้ ของสตริง อย่าลืมนะว่า index ของสตริง s อยู่ในช่วง 0 ถึง len(s)-1 จากซ้ายไปขวา และ -1 ถึง -len(s) ถอยจากขวามาซ้าย	<pre>s = input().strip() t = "" for i in range(10): t = s[i] + t</pre> <p># ถ้า len(s) < 10 ก็ผิดแน่</p>
เลข index ที่ใช้กับสตริงไม่ใช่ จำนวนเต็ม	<pre>print(s[n/2])</pre> <p># ผิดเพราะ n/2 ได้จำนวนจริง</p>
เปลี่ยนข้อมูลภายในสตริง	<pre>t = input().strip() t[0],t[-1] = t[-1],t[0]</pre> <p># สลับตัวแรกกับตัวท้ายแบบนี้ผิด</p> <p>ถ้าต้องการสลับตัวแรกกับตัวท้าย ต้องสร้างใหม่</p> <pre>t = t[-1] + t[1:-1] + t[0]</pre>
เขียนสตริงระบุตำแหน่งของ แฟ้มข้อมูลที่มีเครื่องหมาย \ แต่ไม่ได้เขียน \\	<pre>s = "c:\temp\data\input.txt" # ผิด ต้องเป็น s = "c:\\temp\\data\\input.txt"</pre>
นำสตริงไปบวกกับข้อมูล ประเภทอื่น	<pre>print("average = " + avg) # ผิด ต้องเป็น print("average = " + str(avg)) หรือ print("average =", avg)</pre>

แบบฝึกหัด

Problem	Code
<p><u>Input</u>: รับสตริงหนึ่งบรรทัด</p> <p><u>Process</u>: สร้างสตริงใหม่ที่ทุกอักขระในสตริงที่รับเข้ามาปรากฏซ้ำอีกตัว เช่น รับ 'pypy' จะได้สตริง 'ppypppyy'</p> <p><u>Output</u>: สตริงผลลัพธ์</p>	
<p><u>Input</u>: รับสตริงหนึ่งบรรทัด</p> <p><u>Process</u>: สร้างสตริงใหม่ที่ทุกอักขระในสตริงที่รับเข้ามาปรากฏซ้ำอีกตัว แต่ถ้ามีตัวซ้ำติดกันอยู่แล้ว ก็ไม่ต้องทำซ้ำ เช่น รับ 'pythonnaa' จะได้สตริง 'ppyytthhoonnaa'</p> <p><u>Output</u>: สตริงผลลัพธ์</p>	
<p><u>Input</u>: รับสตริงหนึ่งบรรทัด</p> <p><u>Process</u>: ตรวจสอบว่าสตริงนี้เป็น palindrome (ซึ่งคือสตริงที่กลับลำดับแล้วคือสตริงเดิม) หรือไม่</p> <p><u>Output</u>: ถ้าเป็น ก็แสดง Y ถ้าไม่เป็น ก็แสดง N</p>	
<p><u>Input</u>: รับจำนวนเต็มบวกสองจำนวนเก็บใส่ d กับ n</p> <p><u>Process</u>: สร้างสตริงใหม่จากจำนวน d ที่มี n หลัก โดยถ้า d มีไม่ครบ n หลัก ก็ต้องเติมเลข 0 ไว้ทางซ้ายให้ครบ n หลัก แต่ถ้า d มีจำนวนหลัก \geq n หลัก ก็ให้เป็นสตริงของ d เดิม</p> <p><u>Output</u>: สตริงใหม่ที่ต้องการ</p>	

Problem	Code
<p>Input: รับเลขฐานสิบหก 1 หลัก</p> <p>Process: แปลงเป็นจำนวนในฐานสิบ</p> <p>Output: จำนวนในฐานสิบที่แปลงได้</p> <p>Base 16 : 0, 1, ..., 9, A, B, C, D, E, F</p> <p>Base 10 : 0, 1, ..., 9, 10, 11, 12, 13, 14, 15</p>	
<p>Input: รับสตริงมาหนึ่งบรรทัด</p> <p>Process: นับจำนวนตัวเลขที่ปรากฏในสตริงที่รับเข้ามา</p> <p>Output: จำนวนที่นับได้</p>	
<p>Input: รับสตริงมาหนึ่งบรรทัด</p> <p>Process: นับว่ามีตัวอักษรติดกันกี่คู่ที่เป็นสระภาษาอังกฤษ</p> <p>Output: จำนวนที่นับได้</p>	
<p>Input: รับสตริงที่มีแต่เลข 0 กับ 1</p> <p>Process: แปลงสตริงเลขฐานสองที่ได้รับให้เป็นจำนวนเต็มในฐานสิบ (คือ int นั่นเอง) ตัวอย่างเช่น</p> $\text{"01101"} = 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$ $= 13$ <p>Output: จำนวนเต็มที่ได้</p>	

ตัวอย่างการแก้โจทย์ปัญหา

ตรวจสอบเลขซ้ำ

จงเขียนโปรแกรมที่รับสตริง เพื่อตรวจสอบว่าสตริงนี้มีเลขซ้ำกันหรือไม่ เช่น ถ้ารับ ...102...89...3... แบบนี้ไม่มีเลขซ้ำ แต่ถ้ารับ ...102...89...2... แบบนี้มีเลขซ้ำ (มีเลข 2 ซ้ำ)

▶ ข้อมูลนำเข้า

รับสตริงหนึ่งบรรทัด

▶ ข้อมูลส่งออก

ถ้าสตริงที่รับมามีเลขซ้ำ แสดง True แต่ถ้าไม่มีเลขซ้ำ แสดง False

▶ ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
..125.9.0.	False
..125.9.2.	True

ตัวอย่างการเขียนโปรแกรม

โปรแกรม	คำอธิบาย
<pre>t = '' for e in input().strip() : if '0' <= e <= '9' : t += e print(t)</pre>	อาจเริ่มด้วยการลบอักขระที่ไม่ใช่ตัวเลขออกจากสตริงที่รับมาก่อน หลังจากนั้นก็ค่อยหาวิธีตรวจสอบอักขระซ้ำ แต่เราเรียนมาแล้ว ไม่สามารถเปลี่ยนแปลงอักขระในสตริงได้ ดังนั้นต้องใช้วิธีการสร้างสตริงใหม่ เช่น ถ้าจะลบอักขระตัวที่ index 2 ออก ก็ใช้การสร้างใหม่แล้วทับตัวเดิมด้วยคำสั่ง <code>s = s[:2]+s[3:]</code> วิธีนี้ค่อนข้างยุ่ง จึงขอเปลี่ยนเป็นการเลือกเฉพาะตัวเลขในสตริงที่รับมา เพิ่มใส่สตริงใหม่อีกตัวหนึ่ง จะง่ายกว่า โปรแกรมทางซ้าย เริ่มด้วยสตริงว่าง <code>t</code> จากนั้นวนหยิบอักขระออกมาทีละตัวจากสตริงที่รับเข้ามา ถ้าอักขระนั้นเป็นตัวเลข ก็เพิ่มใส่ <code>t</code> สั่ง run, ใส่ 12a34bce5, ได้ผลเป็น 12345, ถูกต้อง
<pre>t = '' for e in input().strip() : if '0' <= e <= '9' : t += e b = '' for e in t : if e in b : break b += e</pre>	เราได้ <code>t</code> เก็บสตริงที่มีตัวเลขอย่างเดียว ก็มาถึงขั้นตอนตรวจว่า <code>t</code> ไม่มีเลขซ้ำหรือไม่ ? จะทำอย่างไร ? ถ้าเราค่อย ๆ ดูอักขระทีละตัวในสตริง ก็ต้องตรวจว่าอักขระตัวใหม่นี้ปรากฏในอักขระที่ดูผ่านมาหรือไม่ ถ้าเราใช้วงวน <code>for e in t</code> ก็จำตัวที่ดูผ่านมาด้วยการใช้ตัวแปรใหม่ <code>b</code> เริ่มด้วยสตริงว่าง ถ้าตัวใหม่ไม่มีใน <code>b</code> เพิ่มตัวใหม่ต่อท้ายเข้าไปใน <code>b</code> ดัง code ทางซ้ายนี้ เมื่อใดพบว่า <code>e in b</code> เป็นจริง คือตัวใหม่ซ้ำกับตัวที่ดูผ่านมา ก็ <code>break</code> ออกจากวงวนได้ (คือรู้ว่ามีเลขซ้ำแล้ว)

โปรแกรม	คำอธิบาย
<pre> t = '' for e in input().strip() : if '0' <= e <= '9' : t += e b = '' for e in t : if e in b : print(True) break b += e else: print(False) </pre>	<p>แต่การ break ออกจากวงวนทันทีที่พบตัวซ้ำนั้น พอออกจากวงวนแล้ว เราจะไม่ทราบว่า การหลุดออกจากวงวนมาจากกรณี break หรือมาจากกรณีที่วนครบทุกอักขระแล้ว ในโจทย์บอกว่า ถ้าพบซ้ำ ให้แสดง True ถ้าไม่ซ้ำเลย ให้แสดง False ดังนั้นควร print(True) เลยเมื่อพบซ้ำ แล้ว break ออกจากวงวน ส่วนการตรวจว่าได้วนครบทุกตัวโดยไม่ break ก็ทำได้โดยเพิ่มคำสั่ง print(False) หลัง else ของ for นั้นแสดงว่าวนครบทุกตัวแล้วไม่พบเลขซ้ำเลย ได้ code ทางซ้าย</p> <p>สั่ง run, ใส่ ..125.9.0., ได้ผลเป็น False, แสดงว่าไม่มีซ้ำ, ถูกต้อง สั่ง run, ใส่ ..125.9.2., ได้ผลเป็น True, แสดงว่ามีซ้ำ, ถูกต้อง</p>
<pre> t = '' for e in input().strip() : if '0' <= e <= '9' : t += e for k in range(len(t)) : if t[k] in t[:k] : print(True) break else: print(False) </pre>	<p>ถ้าคิดอีกนิต พบว่า หากเราวนถึงรอบที่ k การตรวจว่าตัวที่ k ซ้ำกับตัวที่ผ่านมาหรือไม่สามารถดูตัวที่ผ่านมาจากส่วนทางซ้ายของ t ไม่เห็นจำเป็นต้องสร้าง b ในแบบที่ทำมา ด้วยคำสั่ง if t[k] in t[:k] จึงเปลี่ยนจาก for e in t เป็น for k in range(len(t)) ได้ตั้ง code ทางซ้าย</p> <p>สั่ง run, ใส่ ..125.9.0., ได้ผลเป็น False, แสดงว่าไม่มีซ้ำ, ถูกต้อง สั่ง run, ใส่ ..125.9.2., ได้ผลเป็น True, แสดงว่ามีซ้ำ, ถูกต้อง</p>
<pre> t = input().strip() for k in range(len(t)) : if '0'<=t[k]<= '9' and \ t[k] in t[:k] : print(True) break else: print(False) </pre>	<p>เราสามารถปรับปรุงต่อได้อีก โดยยุบรวมวงวน for สองวงวนเข้าด้วยกัน for แรกจะจัดอักขระที่ไม่ใช่ตัวเลข for หลังตรวจเรื่องตัวซ้ำ เราก็ยุบรวมให้มี for เดียว ดูไล่ไปทีละตัว ถ้าไม่ใช่ตัวเลขก็ข้ามไป ถ้าใช่ก็ตรวจว่าซ้ำกับที่ผ่านมาหรือไม่ในทำนองเดียวกับที่ทำมา ได้ code ทางซ้าย</p> <p>สั่ง run, ใส่ ..125.9.0., ได้ผลเป็น False, แสดงว่าไม่มีซ้ำ, ถูกต้อง สั่ง run, ใส่ ..125.9.2., ได้ผลเป็น True, แสดงว่ามีซ้ำ, ถูกต้อง</p>
<pre> t = input().strip() for k,e in enumerate(t) : if '0' <= e <= '9' and \ e in t[:k] : print(True) break else: print(False) </pre>	<p>หรือจะเปลี่ยนมาใช้ for k,e in enumerate(t): ก็เป็นแบบที่นิยมกว่า แบบบน for แบบนี้จะได้ทั้ง index (เก็บใน k) และตัวข้อมูล (เก็บใน e) มาใช้งานในวงวน</p> <p>สั่ง run, ใส่ ..125.9.0., ได้ผลเป็น False, แสดงว่าไม่มีซ้ำ, ถูกต้อง สั่ง run, ใส่ ..125.9.2., ได้ผลเป็น True, แสดงว่ามีซ้ำ, ถูกต้อง</p>



ตัวอย่างโจทย์ปัญหา

แปลงวันที่

ให้อ่านวันเดือนปีในรูปแบบ เดือน/วันที่/ปี (mm/dd/yyyy) โดยรับค่าทางแป้นพิมพ์ และแปลงวันที่ที่ได้รับเป็นรูปแบบ วันที่ เดือน ปี (dd MMM yyyy) ค่าของเดือนที่รับมาเป็นตัวเลขจำนวนเต็ม ต้องแปลงให้อยู่ในรูปตัวย่อภาษาอังกฤษของเดือนที่รับเข้ามา

► ข้อมูลนำเข้า

มี 1 บรรทัด เป็นข้อความสตริง แทนวันที่ในรูปแบบ เดือน/วันที่/ปี (mm/dd/yyyy)

► ข้อมูลส่งออก

มี 1 บรรทัด เป็นวันที่ในรูปแบบ วันที่ เดือน ปี (dd MMM yyyy)

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
08/27/2014	27 AUG 2014
02/01/2018	01 FEB 2018

นับตัวอักษรพิมพ์ใหญ่

ให้เขียนโปรแกรมอ่านข้อมูลที่เป็นสตริงเข้ามาหนึ่งบรรทัด ประกอบด้วยอักขระอะไรก็ได้ เว้นวรรคก็ได้ จากนั้นให้นับเฉพาะตัวอักษรที่เป็นตัวใหญ่เท่านั้น และแสดงผลลัพธ์ออกมาทางหน้าจอ

► ข้อมูลนำเข้า

มี 1 บรรทัด ประกอบด้วยสตริงที่ประกอบไปด้วยตัวอักขระอะไรก็ได้ เว้นวรรคก็ได้

► ข้อมูลส่งออก

มี 1 บรรทัด แสดงจำนวนตัวอักษรที่เป็นตัวใหญ่ทั้งหมดที่นับได้จากข้อมูลนำเข้า

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
HeLLo WoRLd	5
PYTHON123	6
yes!	0

กลุ่มสระ

คำภาษาอังกฤษประกอบด้วยพยัญชนะและสระ (a, e, i, o, u) ขอเรียกสระที่อยู่ติดกันว่าเป็น กลุ่มสระ เช่น beautiful มีกลุ่มสระ eau, i, และ u จึงมีกลุ่มสระ 3 กลุ่ม, vowel มีกลุ่มสระ 2 กลุ่ม, group มีกลุ่มสระ 1 กลุ่ม และ rhythm มีกลุ่มสระ 0 กลุ่ม

ให้เขียนโปรแกรมเพื่อนับจำนวนกลุ่มสระในคำภาษาอังกฤษ

► ข้อมูลนำเข้า

มี 1 บรรทัด แทนคำภาษาอังกฤษ ประกอบด้วยอักขรตัวพิมพ์เล็กเท่านั้น

► ข้อมูลส่งออก

มี 1 บรรทัด แสดงจำนวนกลุ่มสระของคำที่กำหนด

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
beautiful	3
vowel	2
group	1
rhythm	0



วรวิธ สัตยวินิจ (Product Manager)
ภัทรารุณ ชื้อสัตยาศิลป์ (CTO)
ยอด ชินสุภักกุล (CEO)
ศุภฤกษ์ กฤตยาเกียรติ (Software Architect)
เอกลักษณ์ วิริยะโกวิทยา (COO)

(จากซ้ายไปขวา)

Intania 84

Wongnai

ไม่ว่าคุณจะเรียนคณะอะไรหรือภาควิชาอะไร **Computer Programming** ได้กลายมาเป็นความรู้ที่ทุกคนจำเป็นต้องมีไปแล้ว เพราะเทคโนโลยีเกิดใหม่ในธุรกิจและอุตสาหกรรมต่าง ๆ ไม่ว่าจะเป็นทางด้านการค้าขาย ธนาคาร การแพทย์ โฆษณา บันเทิง กีฬา เครื่องจักรและหุ่นยนต์ ฯลฯ ล้วนแล้วแต่ต้องใช้ความรู้ด้าน **Computer Programming** มาช่วยพัฒนาผลิตภัณฑ์หรือบริการให้ดีขึ้นกว่าเดิม โดยความต้องการนี้จะยิ่งมากขึ้นมากในอนาคต เมื่อผลิตภัณฑ์ บริการ หรือแม้กระทั่งเงินที่เราใช้ซื้อสินค้าต่าง ๆ ถูกแปลงไปอยู่ในรูปแบบดิจิทัล จึงเป็นเรื่องสำคัญมากที่เราจะใส่ใจกับความรู้ด้าน **Computer Programming** เพราะมันคือ “ทักษะที่จำเป็น” ของคนที่จะสามารถรับมือกับความเปลี่ยนแปลงของโลกดิจิทัล และพร้อมเก็บเกี่ยวโอกาสที่จะมาพร้อมกับคลื่นยักษ์ลูกใหม่ในอนาคตอันใกล้

สรุปเนื้อหา

การอ่านข้อมูลที่เก็บในแฟ้มข้อมูลเข้ามาประมวลผลทำได้หลายวิธีหลายคำสั่ง วิชานี้ใช้วิธีง่ายสุด ๆ คือ อ่านแฟ้มข้อความ (text file) ทีละบรรทัดมาเก็บในสตริง มีรูปแบบดังนี้

<p>อ่านแฟ้มข้อความทีละบรรทัด ใช้ <code>readline</code></p> <pre>infile = open(filename, 'r') first_line = infile.readline() second_line = infile.readline() for line in infile : # คำสั่ง for อ่านจาก infile รอบละหนึ่งบรรทัด # มาเก็บเป็นสตริงใน line จนหมดแฟ้ม ... infile.close() # ไม่อ่านแล้ว ก็ปิดแฟ้ม</pre> <p>ถ้าบรรทัดที่อ่านเข้ามาเป็นบรรทัดว่าง ๆ เก็บใน <code>t</code> จะได้ <code>t = '\n'</code> หรือ <code>len(t)</code> เป็น 1 แต่ถ้าอ่านตอนที่แฟ้มไม่มีข้อมูลให้อ่านแล้ว จะได้ <code>t = ''</code> หรือ <code>len(t)</code> เป็น 0</p> <pre>for line in infile : ... เหมือนกับ line = infile.readline() while len(line) > 0 : ... line = infile.readline()</pre>	<p>อ่านข้อมูลในแฟ้มมาหาค่าเฉลี่ย แต่ละบรรทัดเก็บรหัสกับคะแนน</p> <pre>infile = open('c:/temp/data.txt', 'r') n = 0 s = 0 for line in infile : sid, sc = line.strip().split() s += float(sc) n += 1 infile.close() print('average =', (s/n))</pre> <p>อีกแบบ อ่านข้อมูลในแฟ้มมาหาค่าเฉลี่ย แต่ละบรรทัดเก็บรหัสกับคะแนน บรรทัดแรกบอกจำนวนบรรทัดที่ต้องอ่าน</p> <pre>infile = open('c:/temp/data.txt', 'r') n = int(infile.readline()) # อ่านบรรทัดแรก s = 0 for k in range(n) : sid, sc = infile.readline().strip().split() s += float(sc) infile.close() print('average =', (s/n))</pre>
<p>การอ่านจากแฟ้มหนึ่งบรรทัด อาจมีรหัสขึ้นบรรทัดใหม่ <code>\n</code> อยู่ทีปลายบรรทัด (กรณีอ่านบรรทัดสุดท้ายของแฟ้มอาจไม่มีรหัสนี้)</p> <p>ถ้าไม่ต้องการรหัส <code>\n</code> ก็อย่าลืมลบทิ้ง แต่ถ้าอ่านมาแล้วมีการ <code>strip()</code> รหัส <code>\n</code> จะถูกขจัดไปด้วย ไม่ต้องทำอะไรเพิ่ม</p>	<pre>f = open('data.txt', 'r') t = f.readline() if len(t)>0 and t[-1]=='\n' : t = t[:-1] for line in f : if line[-1] == '\n' : line = line[:-1] ...</pre>

<p>บันทึกข้อมูลลงแฟ้มข้อความ</p> <pre>outfile = open('c:/temp/out.txt','w') outfile.write(any_string) outfile.write(any_string + '\n') ... outfile.close()</pre> <p>อย่าลืม : write ไม่ได้เพิ่มรหัส '\n' เพื่อขึ้นบรรทัดใหม่ให้ ต้องเพิ่มเองเมื่อต้องการ</p>	<p>หาจำนวนเฉพาะที่มีค่าน้อยกว่า m บันทึกลงแฟ้มบรรทัดละ 5 ตัว</p> <pre>m = int(input()) outfile = open('D:/primes.txt','w') c = 0 result = "" for n in range(2,m): for k in range(2,n): if n % k == 0: break else: result += str(n) + ", " c += 1 if c % 5 == 0: outfile.write(result[:-2] + '\n') result = "" if len(result) > 0: outfile.write(result[:-2] + '\n') outfile.close()</pre>
--	--

เรื่องพิศบอย

<p>เมื่ออ่านข้อมูลจากแฟ้มมาหนึ่งบรรทัด ลืมลบรหัส \n ทำให้การประมวลผลผิดพลาด</p>	<pre>t = input().strip() f = open('names.txt', 'r') for line in f : if line == t : # ผิด เพราะ line อาจมีรหัส \n print(t, ': found in names.txt') break else : print(t, ': not found')</pre>
<p>ชื่อแฟ้มมีเครื่องหมาย \ แต่ใส่แค่ตัวเดียว</p>	<pre>of = open('c:\temp\data.txt', 'w') ผิด เพราะ \t คือ tab ต้องเป็น of = open('c:\\temp\\data.txt', 'w') หรือใช้ / ก็ได้ เพราะระบบรับชื่อแฟ้มที่เขียนแบบ / ได้ (ไม่ได้หมายความว่า \ เหมือนกับ /) of = open('c:/temp/data.txt', 'w')</pre>
<p>เปิดแฟ้มที่ไม่มีอยู่ในเครื่องมาอ่าน</p>	<pre>infile = open('h:\\file.data') ถ้าไม่มีแฟ้ม h:/file.data ในเครื่อง ก็จะมี fn = input() infile = open(fn) อาจผิดได้ ถ้า fn มี blank หน้าหรือหลัง จึงควร strip ก่อน infile = open(input().strip())</pre>

พิมพ์ readline เป็น readlines
readlines อ่านทีเดียวหมดแฟ้ม (อ่าน
เสร็จแล้วได้เป็น list of strings ที่
ยังไม่ได้นำเสนอ จึงขอไม่ลงในรายละเอียด)

```
infile = open('c:/temp/data.txt')
first_line = infile.readlines() # ระวัง
second_line = infile.readline() # ควรใช้แบบบรรทัดนี้
```



Problem	Code								
<p>Input: หนึ่งบรรทัดเป็นชื่อแฟ้ม Process: อ่านข้อความในแฟ้มมากลบลำดับบรรทัด Output: แสดงข้อความในแฟ้มแบบกลับลำดับบรรทัด ออกทางจอภาพ เช่น</p> <table border="1"> <thead> <tr> <th>ข้อมูลในแฟ้ม</th><th>ผลลัพธ์ (ออกทางจอ)</th></tr> </thead> <tbody> <tr> <td>line1</td><td>line3</td></tr> <tr> <td>line2</td><td>line2</td></tr> <tr> <td>line3</td><td>line1</td></tr> </tbody> </table>	ข้อมูลในแฟ้ม	ผลลัพธ์ (ออกทางจอ)	line1	line3	line2	line2	line3	line1	
ข้อมูลในแฟ้ม	ผลลัพธ์ (ออกทางจอ)								
line1	line3								
line2	line2								
line3	line1								
<p>Input: หนึ่งบรรทัดเป็นชื่อแฟ้ม Process: อ่านข้อความในแฟ้มแล้วกลับลำดับบรรทัด แต่มีเงื่อนไขว่าจะไม่เอาบรรทัดที่ว่าง ๆ หรือมีแต่ blank Output: บันทึกข้อความแบบกลับลำดับบรรทัดลงแฟ้ม ชื่อ reverse.txt</p>									
<p>Input: หนึ่งบรรทัดเป็นชื่อแฟ้ม Process: แสดงหัวข้อข่าวทั้งหมดในแฟ้ม หัวข้อข่าวเป็น ข้อความที่อยู่ระหว่าง <headline> กับ </headline> ในแฟ้มนี้ (ทั้ง <headline> กับ </headline> อยู่ในบรรทัดเดียวกันแน่ ๆ และ แต่ละบรรทัดมีไม่เกิน 1 หัวข้อข่าว) Output: บรรทัดละหนึ่งหัวข้อออกทางจอภาพ ให้ครบทุกหัวข้อ</p>									

Problem	Code
<p>Input: สองบรรทัด แต่ละบรรทัดเป็นชื่อแฟ้ม</p> <p>Process: เปรียบเทียบว่าสองแฟ้มนี้มีค่าเหมือนกันหรือไม่</p> <p>Output: ถ้าแฟ้มทั้งสองมีข้อมูลเหมือนกัน แสดง True ต่างกันก็แสดง False</p>	



ตรวจสอบคำตอบ

จเขียนโปรแกรมอ่านแฟ้มเก็บคำตอบแบบปรนัยของนักเรียน มาตรวจให้คะแนน

► ข้อมูลนำเข้า

ข้อมูลจากแฟ้ม c:\t\answers.txt รูปแบบแฟ้มเป็นดังนี้

บรรทัดแรกเก็บเฉลย เป็นสตริงของตัวอักษร A, B, C หรือ D

บรรทัดต่อมาจนหมดแฟ้ม แต่ละบรรทัด เก็บเลขประจำตัวนักเรียน ตามด้วยช่องว่างหนึ่งช่อง ตามด้วยคำตอบแบบปรนัย ซึ่งเป็นสตริงของตัวอักษร A,B,C,D หรือเป็นช่องว่าง (กรณีไม่ตอบข้อนั้น) หรือเป็นตัวอื่น (กรณีกรอกมากกว่าหนึ่งคำตอบ)

► ข้อมูลส่งออก

แฟ้มใหม่ c:\t\results.txt เก็บผลการตรวจ แต่ละบรรทัดประกอบด้วยเลขประจำตัวนักเรียนตามด้วยคะแนนที่ได้ คั่นด้วยเครื่องหมายจุลภาค (comma)

► ตัวอย่าง

Input (อ่านจากแฟ้ม)	Output (บันทึกลงแฟ้ม)
AABBCCCBCCDDDABABDDCCDDCC	5630120421,25
5630120421 AABBCCCBCCDDDABABDDCCDDCA	5631010121,14
5631010121 A BB CD BDBAAABA ABDBCCCDCC	563102121,16
563102121 ABABCCNNAADDDABAB CCDDAC	5630121821,24
5630121821 AABBCCCBCCDDDABABDDCCDD	

ตัวอย่างการเขียนโปรแกรม

โจทย์กำหนดให้อ่านข้อมูลจากแฟ้มและบันทึกผลการทำงานลงแฟ้ม เพื่อให้การเขียนโปรแกรมและหาที่ผิดได้ง่ายขึ้น จะขอเขียนแบบแสดงผลออกหน้าจอก่อน เมื่อทุกอย่างถูกต้อง ค่อยเปลี่ยนให้บันทึกลงแฟ้ม

โปรแกรม	คำอธิบาย
<pre> fin = open('c:\t\answers.txt', 'r') soln = fin.readline().strip() for line in fin: sid,ans = line.strip().split() point = 0 for k in range(len(soln)): if ans[k] == soln[k]: point += 1 print(sid, point) fin.close() </pre>	<p>เริ่มด้วยการเปิดแฟ้ม, ใช้ <code>readline</code> หลังเปิดแฟ้มทันทีจะได้บรรทัดแรกของแฟ้มซึ่งคือเฉลย (<code>strip</code> เพื่อลบช่องว่างซ้ายขวาและรหัสขึ้นบรรทัดใหม่) จากนั้นใช้ <code>for</code> อ่านบรรทัดที่เหลือ ใช้ <code>split</code> แยกเลขประจำตัวกับคำตอบออกจากกัน แล้วใช้อีก <code>for</code> นำคำตอบทีละตัวเปรียบเทียบกับเฉลย (เราใช้ <code>for</code> แบบเปลี่ยนค่า <code>k</code> เป็น <code>index</code> ของทั้งคำตอบกับเฉลย) ถ้าตรงกันก็เพิ่มคะแนน วนตรวจครบทุกข้อก็แสดงผลทางจอภาพ วนครบทุกบรรทัดก็ปิดแฟ้ม ดัง code ทางซ้าย ก่อนสั่งทำงาน ก็ต้องสร้างแฟ้ม <code>answers.txt</code> จากนั้นสั่ง <code>run</code>, ได้</p> <p><code>OSError: [Errno 22] Invalid argument: 'c:\t\x07nswers.txt'</code> เกิดอะไรแปลก ๆ กับชื่อแฟ้ม</p>
<pre> fin = open('c:\\t\\answers.txt','r') soln = fin.readline().strip() for line in fin: sid,ans = line.strip().split() point = 0 for k in range(len(soln)): if ans[k] == soln[k]: point += 1 print(sid, point) fin.close() </pre>	<p>ถ้ายังจำได้ เครื่องหมาย <code>\</code> ที่ปรากฏในสตริงจะถูกตีความหลายแบบ ถ้าต้องการสัญลักษณ์ <code>\</code> ในสตริง ต้องเขียน <code>\\</code> แก้ให้ถูกต้อง</p> <p>สั่ง <code>run</code>, ได้ 5630120421 25, ผิดบรรทัดที่ 4 ของโปรแกรม</p> <p><code>ValueError: too many values to unpack (expected 2)</code> แปลว่าหลัง <code>split</code> แล้ว ได้สตริงมากกว่าตัวแปรที่จะมารับผล ซึ่งน่าจะเกิดหลังอ่านบรรทัดที่ 3 ของแฟ้ม</p> <p>5631010121 A BB CD BDBAAABA ABDBCCCDCC บรรทัดนี้มีช่องว่างในคำตอบ (ช่องว่างแปลว่าข้อนั้นไม่ตอบ) จึงทำให้ <code>split</code> แล้วได้สตริงมากกว่า 2 สตริง จึงต้องหาทางแยกบรรทัดให้เป็นสองสตริง เลขประจำตัว กับ คำตอบทั้งหลาย</p>

โปรแกรม	คำอธิบาย
<pre> fin = open('c:\\t\\answers.txt','r') soln = fin.readline().strip() for line in fin: line = line.strip() j = line.find(' ') if j > 0 : sid = line[:j] ans = line[j:] point = 0 for k in range(len(soln)): if ans[k] == soln[k] : point += 1 print(sid, point) fin.close() </pre>	<p>เราไม่ควรแก้ปัญหานี้โดยคิดว่าเลขประจำตัวมี 10 หลัก ก็แยกด้วย</p> <pre>sid = line[:10] ans = line[10:]</pre> <p>เพราะถ้าดูในแฟ้ม พบว่าเลขประจำตัวบางคนมีน้อยกว่า 10 หลัก จึงควรใช้วิธีหาช่องว่างแรกจากทางซ้ายด้วยบริการ <code>find</code> ของสตริง (แต่ต้องอย่าลืม <code>strip</code> ก่อน ไม่เช่นนั้น ถ้า <code>line</code> เริ่มต้นด้วยช่องว่าง ก็จะผิด) เมื่อหาช่องว่างพบที่ index <code>j</code> ก็แยกได้ด้วย</p> <pre>sid = line[:j] ans = line[j:]</pre> <p>แล้วก็ตรวจสอบคำตอบด้วยวิธีที่ทำมา สั่ง <code>run</code>, ได้</p> <pre>5630120421 12 5631010121 7 563102121 6</pre> <p>แล้วเกิดข้อผิดพลาดที่คำสั่ง <code>if ans[k] == soln[k]</code> IndexError: string index out of range ถ้าดูผลคะแนนที่ได้ก่อนเกิดข้อผิดพลาด จะพบว่าได้คะแนนรวมผิดด้วย</p>
<pre> fin = open('c:\\t\\answers.txt','r') soln = fin.readline().strip() for line in fin: line = line.strip() j = line.find(' ') if j > 0 : sid = line[:j] ans = line[j:].strip() point = 0 for k in range(len(soln)): if ans[k] == soln[k] : point += 1 print(sid, point) fin.close() </pre>	<p>ผลที่ผิด ได้คะแนนลดลงมาก เหมือนกับว่า คำตอบผิดมีมากผิดปกติ ถ้าแทรกคำสั่ง <code>print(ans)</code> กับ <code>print(soln)</code> ออกมาดูเทียบกัน จะได้ (ขอตัดมาให้ดูแค่ 2 บรรทัดแรก)</p> <pre>AABBCCCBCCDDDABABDDCCDDCA AABBCCCBCCDDDABABDDCCDDCC</pre> <p>เห็นได้ว่าบรรทัดบน <code>ans</code> มันเลื่อนไปทางขวา เพราะว่ามีช่องว่างทางซ้าย แก้ปัญหานี้ด้วย</p> <pre>ans = line[j:].strip()</pre> <p>เพื่อตัดช่องว่างออก</p> <p>สั่ง <code>run</code>, ได้</p> <pre>5630120421 25 5631010121 14</pre> <p>แล้วเกิดข้อผิดพลาดที่คำสั่ง</p> <pre>if ans[k] == soln[k]</pre> <p>IndexError: string index out of range ทำงานผิดเหมือนครั้งที่แล้ว แต่ได้คะแนนรวมถูกต้อง ส่วนที่ทำงานผิดพลาดกลายมาเกิดกับบรรทัดที่ 4 ในแฟ้ม ซึ่งมีข้อมูล</p> <pre>563102121 ABABCCNNAADDDABAB CCDDAC</pre>

โปรแกรม	คำอธิบาย
<pre> fin = open('c:\\t\\answers.txt','r') soln = fin.readline().strip() for line in fin: line = line.strip() j = line.find(' ') if j > 0 : sid = line[:j] ans = line[j+1:] point = 0 for k in range(len(soln)): if ans[k] == soln[k] : point += 1 print(sid, point) fin.close() </pre>	<p>เนื่องจากโจทย์กำหนดว่าเลขประจำตัวกับคำตอบขึ้นด้วยช่องว่าง 1 ช่อง แต่ในบรรทัดที่ 4 ห่างกัน 2 ช่อง แสดงว่าช่องว่างตัวที่ 2 นั้นแทนคำตอบ (ที่ไม่มี) ของข้อที่ 1 คำสั่ง <code>line[j:].strip()</code> จะลบช่องว่างออกหมด ทำให้ประมวลผลผิด ทำให้มีข้อมูลไม่ครบ และทำให้เมื่อนำ <code>ans[k]</code> มาเทียบคำตอบก็ผิด เพราะ <code>k</code> มีค่าเกินช่องในสตริง จึงแก้ไขคำสั่ง <code>line[j:].strip()</code> เป็น <code>line[j+1:]</code> สั่ง run, ได้</p> <pre> 5630120421 25 5631010121 14 563102121 16 </pre> <p>แล้วเกิดข้อผิดพลาดที่คำสั่ง</p> <pre> if ans[k] == soln[k] </pre> <p><code>IndexError: string index out of range</code> ไม่ผิดบรรทัดที่ 4 ของแฟ้มข้อมูลแล้ว แต่ผิดบรรทัดที่ 5 ของแฟ้ม</p>
<pre> fin = open('c:\\t\\answers.txt','r') soln = fin.readline().strip() for line in fin: line = line.strip() j = line.find(' ') if j > 0 : sid = line[:j] ans = line[j+1:] if len(ans) < len(soln) : ans += ' '*(len(soln)-len(ans)) point = 0 for k in range(len(soln)): if ans[k] == soln[k] : point += 1 print(sid, point) fin.close() </pre>	<p><code>index out of range</code> แปลว่าค่าของ index อยู่นอกช่วงที่ถูกต้อง คำสั่ง <code>ans[k] == soln[k]</code> ผิดได้ที่ <code>ans[k]</code> หรือไม่ก็ที่ <code>soln[k]</code> คำสั่ง <code>soln[k]</code> ไม่น่าผิด เพราะ <code>k</code> มีค่าใน <code>range(len(soln))</code> แต่ <code>ans[k]</code> อาจผิดได้ถ้า <code>ans</code> มีขนาดน้อยกว่า <code>soln</code> ถ้ากลับไปดูที่บรรทัดที่ 5 ในแฟ้มพบว่าคำตอบมีไม่ครบ นักเรียนไม่ตอบคำตอบท้าย ๆ จะเป็นช่องว่าง และถูก <code>strip</code> ทิ้ง</p> <p>วิธีแก้ไข ก็แค่ตรวจว่า ถ้า <code>len(ans) < len(soln)</code> จะเติมช่องว่างต่อทางขวาของ <code>ans</code> เป็นจำนวนเท่ากับผลต่างของความยาวทั้งสองสตริง</p> <p>สั่ง run, ได้</p> <pre> 5630120421 25 5631010121 14 563102121 16 5630121821 24 </pre> <p>ถูกต้อง ภาระที่เหลือก็แค่เปลี่ยนจากการแสดงผลออกหน้าจอเป็นการบันทึกลงแฟ้ม ซึ่งขอให้ผู้อ่านลองเขียนต่อเอง</p>

ตัวอย่างโจทย์ปัญหา

คะแนนเฉลี่ยของตอนเรียน

จงเขียนโปรแกรมเพื่ออ่านแฟ้ม data.txt แฟ้มนี้เก็บข้อมูล
คะแนนของนิสิตโดยมีรูปแบบ id:name:section:score

```
5913842721:Somsak Rakrian:1:56.6
5913845921:Somsri Deeying:2:78.0
5913856821:Rakchard Yingcheep:2:89.0
5913861321:Thumdee Tong Daidee:2:99
591387721:Somrak Rakrian:10:84.25
```

จากนั้นรับข้อมูลจากแป้นพิมพ์ เป็นตอนเรียนที่ต้องการหาค่าคะแนนเฉลี่ย
หากไม่พบนิสิตในตอนเรียนนั้น ให้พิมพ์ Not Found

► ข้อมูลนำเข้า

จำนวนเต็มหนึ่งจำนวน เป็นตอนเรียนที่ต้องการหาค่าคะแนนเฉลี่ย

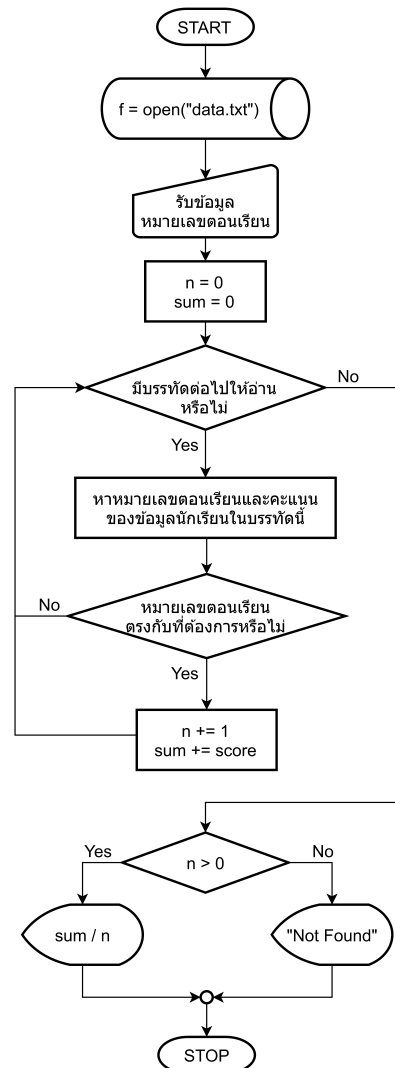
► ข้อมูลส่งออก

แสดงคะแนนเฉลี่ยของนิสิตในตอนเรียนที่ต้องการ

► ตัวอย่าง

สมมติให้แฟ้ม data.txt มีข้อความข้างต้น

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
1	56.6
2	88.66666666666667
3	Not Found



Find Student's Grade

ให้เขียนโปรแกรมเพื่ออ่านแฟ้ม `score.txt` ซึ่งมีรหัสนิสิตและเกรดของนิสิตแต่ละคน (0-4) คั่นด้วยช่องว่าง และรับค่ารหัสนิสิตจากแป้นพิมพ์ แล้วแสดงเกรดของนิสิตคนนั้น หากไม่พบรหัสนิสิตในแฟ้ม ให้แสดง `Not Found`

► ข้อมูลนำเข้า

มี 1 บรรทัด รับรหัสนิสิตเป็นจำนวนเต็ม

► ข้อมูลส่งออก

มี 1 บรรทัด แสดงเกรดของนิสิตเป็นจำนวนเต็ม หากไม่พบรหัสนิสิตในแฟ้ม ให้แสดงคำว่า `Not Found`

► ตัวอย่าง

ข้อมูลในแฟ้ม <code>score.txt</code>	Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
1111 3 12345 2	12345	2
9999 4 89431 0	9999	4
76238 1	1234	Not Found

รหัสลับ

จงเขียนโปรแกรมอ่านข้อมูลรหัสลับจากแฟ้มหนึ่ง ซึ่งระบุว่า จะต้องใช้รหัสตามลำดับอย่างไร จึงจะสามารถปล่อยอาวุธ
อานุภาพรุนแรงได้ วิธีการถอดรหัสจากแฟ้มนี้คือ ต้องรับอินพุตเป็นตัวอักษรสามตัว จากนั้นนับว่า ตัวอักษรทั้งสามตัวนั้น ปรากฏ
เป็นจำนวนเท่าใดในแฟ้ม แล้วจึงเรียงลำดับตัวอักษรทั้งสามตามจำนวนครั้งที่ปรากฏในแฟ้มจากมากที่น้อยที่สุด โดยถือว่า
ตัวอักษรตัวพิมพ์ใหญ่กับตัวพิมพ์เล็ก ไม่เหมือนกัน (case sensitive) และจำนวนตัวอักษรทั้งสามตัวนั้นจะไม่เท่ากัน ตัวอย่างเช่น
หากมีแฟ้ม data.txt เป็นอินพุตดังด้านล่าง

```
agAbggggDf
ffgFFFaaD
DaADDF
FFDFFF
```

จะเห็นว่า ตัวอักษร a มี 4 ตัว, A มี 2 ตัว, b มี 1 ตัว, D มี 6 ตัว, f มี 3 ตัว, F มี 10 ตัว และ g มี 6 ตัว

หากอินพุตเป็น a b f จะได้ผลลัพธ์เป็น afb

หากอินพุตเป็น F A f จะได้ผลลัพธ์เป็น FfA

► ข้อมูลนำเข้า

ข้อความสี่บรรทัด บรรทัดแรกแทนชื่อแฟ้ม อีกสามบรรทัดถัดมาแทนตัวอักษรสามตัว บรรทัดละหนึ่งตัว

► ข้อมูลส่งออก

แสดงข้อความผลลัพธ์ตามต้องการ

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
data.txt a b f	Afb
data.txt F A f	FfA

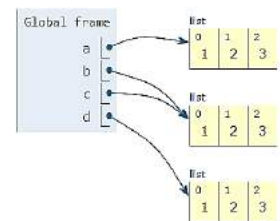
06.1 : List

สรุปเนื้อหา

รายการหรือลิสต์ (list) เป็นที่เก็บกลุ่มของข้อมูลที่มีลำดับ แต่ละตัวมีเลข index กำกับลำดับ ตัวซ้ายสุดมี index 0

- การสร้าง list

- o `x = []` หรือ `x = list()` ได้ลิสต์ว่าง ๆ มี `len(x)` เป็น 0
- o `x = [0]*10` ได้ `x = [0,0,0,0,0,0,0,0,0,0]`
- o `x = list(a)` ได้ลิสต์ที่มีข้อมูลตามที่หยิบออกมาจาก `a` (ด้วย `for e in a`)
`x = list('abcde')` ได้ `x = ['a', 'b', 'c', 'd', 'e']`
`x = list(range(1,10,2))` ได้ `x = [1,3,5,7,9]`
- o `a = [1,2,3]; b = [1,2,3]; c = b; d = list(b)`
ได้ผลดังรูปขวา ที่น่าสนใจคือ คำสั่ง `c = b` ทำให้ `c` กับ `b` เป็นลิสต์เดียวกัน



- ใช้ `+` เพื่อต่อ list และ `*` เพื่อ + หลาย ๆ ครั้ง

`x = 2*([1,2,3] + [3,4,5])` ได้ `x = [1,2,3,3,4,5,1,2,3,3,4,5]`

- `x.append(e)` เพิ่ม `e` ต่อท้าย (ทางขวา) ของลิสต์ `x`
- `x.insert(i, e)` แทรก `e` ไว้ที่ index `i` ของลิสต์ `x`
- `x.pop(i)` ลบข้อมูลตัวที่ index `i` ของลิสต์ `x` และคืนข้อมูลที่ถูกลบเป็นผลลัพธ์
- `x.sort()` ทำให้ข้อมูลในลิสต์ `x` เรียงจากน้อยไปมาก คำสั่งนี้ไม่มีผลคืนกลับมา
- `sorted(x)` คืนลิสต์ที่มีค่าเหมือนกับที่แฉงออกจาก `x` แต่เรียงลำดับข้อมูลจากน้อยไปมากให้เรียบร้อย (`x` ไม่เปลี่ยนแปลง)
- `sum(x)` คืนผลรวมของจำนวนในลิสต์ `x`
- `max(x)` คืนค่ามากสุดในลิสต์ `x`, `min(x)` คืนค่าน้อยสุดในลิสต์ `x`
- `x.count(e)` คืนจำนวนครั้งที่ `e` ปรากฏในลิสต์ `x`
- `if e in x` ใช้ตรวจสอบว่ามี `e` ในลิสต์ `x` หรือไม่
- `x.index(e)` คืน index น้อยสุดที่พบ `e` ในลิสต์ `x` ถ้าไม่พบจะทำงานผิดพลาด จึงต้องตรวจก่อน
`if e in x :`
 `k = x.index(e)`
 `...`
`else :`
 `...`

หมายเหตุ : ลิสต์ไม่มี `find` เหมือนกับของสตริง

- บริการ `split` กับ `join` ไม่ใช่บริการของลิสต์ แต่เป็นของสตริง ที่เกี่ยวข้องกับลิสต์
 - o `t.split()` คืนลิสต์ของสตริงย่อยที่แยกออกจากสตริง `t` โดยใช้ช่องว่างเป็นตัวคั่นสตริงย่อย
 - o `t.split(s)` คืนลิสต์ของสตริงย่อยที่แยกออกจากสตริง `t` โดยใช้สตริง `s` เป็นตัวคั่นสตริงย่อย
 - o `s.join(x)` คืนสตริงที่ได้จากการนำสตริงในลิสต์ `x` มาต่อกันคั่นด้วยสตริง `s`

ตัวอย่างการเข้าใช้ข้อมูลและลิสต์ย่อยในลิสต์ (สมมติให้ $x = [11, 12, 13, 14, 15]$)

- มีวิธีการเข้าใช้และการเลือกลิสต์ย่อยด้วย index เหมือนกับวิธีของสตริง
 - ใช้ `for e in x` : ในการแจกแจงข้อมูลในลิสต์ x จากซ้ายไปขวาออกมาเก็บใน e เพื่อนำไปใช้ในวงวน `for` ได้
คำเตือน : ไม่ควรเปลี่ยนแปลงลิสต์ ในวงวนที่กำลังแจกแจงข้อมูลในลิสต์ออกมาใช้งาน อาจมีพฤติกรรมที่ไม่ตรงกับที่คาดไว้
 - $x[0]$ เหมือน $x[-len(x)]$ ได้ 11
 - $x[-1]$ เหมือน $x[len(x)-1]$ ได้ 15
 - อย่าลืมว่า index ของลิสต์ x ต้องอยู่ในช่วง
 - 0 ถึง $len(x)-1$ จากซ้ายไปขวา
 - -1 ถึง $-len(x)$ ถอยจากขวามาซ้าย
- ดังนั้นเราเขียน $x[k]$ ได้ โดยที่ $-len(x) \leq k \leq (len(x)-1)$ เพราะฉะนั้น $x[-6]$ กับ $x[5]$ ผิด
- $x[a:b]$ (slice ของลิสต์) ได้ลิสต์เสมอ
 - เขียน $x[a:b]$ ค่า a กับ b เป็นอะไรก็ได้ ไม่ผิด
 - $x[2:50000]$ ได้ $[13, 14, 15]$, $x[4999:50000]$ ได้ $[]$
 - $x[-500:-2]$ ได้ $[11, 12, 13]$, $x[-3:-500:-1]$ ได้ $[13, 12, 11]$, $x[-500:-300]$ ได้ $[]$
 - x มีค่าเหมือน $x[:]$ เหมือน $x[0:]$ เหมือน $x[:len(s)]$ เหมือน $x[::]$ เหมือน $x[::1]$
 - $x[::2]$ ได้ลิสต์ย่อยเฉพาะ index คู่ $x[1::2]$ ได้ลิสต์ย่อยเฉพาะ index คี่
 - $x[::-1]$ เหมือน $x[-1::-1]$ เหมือน $x[-1:-len(x)+1:-1]$ ได้ $[15, 14, 13, 12, 11]$

ตัวอย่างการเปลี่ยนข้อมูลในลิสต์ (เปลี่ยนอักขระภายในสตริงไม่ได้ แต่เปลี่ยนข้อมูลภายในลิสต์ได้)

- $x[k] = e$ เหมือนกับ $x[k:k+1] = [e]$
- $x[a:b:c] = y$ นำข้อมูลในลิสต์ y ไปแทนข้อมูลใน $x[a:b:c]$
ถ้า $|c| > 1$ $len(y)$ ต้องเท่ากับ $len(x[a:b:c])$ เช่น
 $x = [1, 2, 3, 4, 5]$; $x[::2] = [0, 0, 0]$ ทำให้ x เปลี่ยนเป็น $[0, 2, 0, 4, 0]$
- $x[len(x):] = [e]$ เหมือนกับ $x.append(e)$
- $x[i:i] = [e]$ เหมือนกับ $x.insert(i, e)$
- $x[i:i+1] = []$ เหมือนกับ $x.pop(i)$
- $x += [1, 2]$ กับ $x = x + [1, 2]$ ทำให้ x มีสมาชิกเพิ่มอีก 2 ตัวคือ 1 กับ 2 เหมือนกัน แต่สองคำสั่งนี้มีการทำงานต่างกัน
 - $x += [1, 2]$ หมายความว่า ให้นำ 1 กับ 2 ต่อท้ายลิสต์ x
เหมือนกับทำ $x.append(1)$ ตามด้วย $x.append(2)$
 - $x = x + [1, 2]$ หมายความว่า ให้ x เก็บลิสต์ใหม่ที่สร้างจากการนำค่าในลิสต์ x เดิมมาต่อกับลิสต์ $[1, 2]$
 - ดังนั้น $x = [3]$; $y = x$ ทำให้ x กับ y เป็นลิสต์เดียวกัน
แต่ถ้าต่อด้วย $x = x + [9]$ จะทำให้ x กับ y เป็นลิสต์คนละตัว
 - ในขณะที่ $x = [3]$; $y = x$ เมื่อทำ $x.append(9)$ แล้ว x กับ y ก็ยังเป็นลิสต์เดียวกัน
- สรุปความแตกต่างของ $x = y$, $x = y[:]$, $x[:] = y$, และ $x[:] = y[:]$ เมื่อทั้ง x และ y เป็นลิสต์
 - $x = y$ x เปลี่ยนไปอ้างอิงลิสต์ตัวเดียวกับของ y (หมายความว่า x และ y อ้างอิงลิสต์เดียวกัน)
 - $x = y[:]$ x เปลี่ยนไปอ้างอิงลิสต์ใหม่ที่ถูกสร้างขึ้นเหมือนกับของ y เหมือนคำสั่ง $x = list(y)$
 - $x[:] = y$ x ยังอ้างอิงลิสต์ตัวเดิม แต่ข้อมูลในลิสต์ x เปลี่ยนไปเหมือนกับข้อมูลของ y (x กับ y เป็นคนละลิสต์)
 - $x[:] = y[:]$ ได้ผลเหมือน $x[:] = y$

รูปแบบการประมวลผลลิสต์ที่พบบ่อย

ใช้ลิสต์เก็บข้อมูลเพื่อนำมาใช้ภายหลัง	อ่านข้อมูลเข้ามา n ตัว <pre>n = int(input()) data = [] for k in range(n): data.append(float(input()))</pre>
ต้องการหุบข้อมูลในลิสต์จากซ้ายไปขวา มาประมวลผลทีละตัว ใช้ <code>for e in x</code>	ต้องการนับว่าลิสต์ x มีข้อมูลเท่ากับ e กี่ตัว <pre>c = 0 for d in x : if d == e : c += 1</pre> หรือแบบสั้น ๆ <pre>c = x.count(e)</pre>
ต้องการหุบข้อมูลของลิสต์ในช่วงที่สนใจมาประมวลผล ใช้ <code>for e in x[a:b:c]</code>	ต้องการหาผลรวมของคะแนนที่เก็บในลิสต์ x โดยขอไม่รวมคะแนนที่น้อยสุดและมากที่สุด (ตัดออกอย่างละหนึ่งตัว) <pre>s = 0 for d in sorted(x)[1:-1] : s += d</pre> หรือแบบสั้น ๆ <pre>s = sum(sorted(x)[1:-1])</pre>
ต้องการปรับเปลี่ยนค่าในลิสต์ ใช้ <pre>for i in range(len(x)) : x[i] = ...</pre>	x เป็นลิสต์เก็บคะแนน ต้องการปรับช่องที่มีค่าน้อยกว่า 30 ให้มีค่าเพิ่มอีก 10% <pre>for i in range(len(x)) : if x[i] < 30 : x[i] += 0.1*x[i]</pre> เขียนแบบข้างล่างนี้ไม่ได้ เพราะ e ที่แจงออกมาเป็นที่เก็บคนละที่กับที่อยู่ ในลิสต์ <pre>for e in x : if e < 30 : e += 0.1*e</pre>
ต้องการหุบข้อมูลแต่ละตัว พร้อมกับ index ของตัวนั้น ๆ	<pre>for i in range(len(t)) : c = t[i] print(i,c)</pre> หรือ <pre>for i,c in enumerate(t): print(i,c)</pre>
ต้องการหุบข้อมูลในลิสต์จากขวาไปซ้าย ทีละตัว	<pre>for e in x[::-1] : ...</pre> หรือ <pre>for k in range(-1,-(len(x)+1),-1) : # อ่านยาก โอกาสผิดสูง e = x[k] ...</pre>

ต้องการหุบข้อมูลในลิสต์มาประมวลผลจนกว่าเงื่อนไขหนึ่งจะเป็นจริง	<pre> for e in x : if เงื่อนไขที่ต้องการ : ... break ... else : ... # มาทำที่นี่ ถ้าไม่พบเงื่อนไขที่ต้องการเลย </pre>
ต้องการหุบข้อมูลในลิสต์ตามลำดับที่เรียงจากข้อมูลน้อยสุดไปข้อมูลมากสุดในลิสต์	<pre> for e in sorted(x) : ... </pre>
ต้องการหุบข้อมูลในลิสต์ตามลำดับที่เรียงจากข้อมูลมากสุดไปข้อมูลน้อยสุดในลิสต์	<pre> for e in sorted(x)[::-1] : ... </pre>
ต้องการหุบข้อมูลในลิสต์จากซ้ายไปขวา มาประมวลผลทีละคู่ข้อมูลที่ติดกัน	<p>ต้องการตรวจดูว่า ข้อมูลในลิสต์ s เรียงลำดับจากน้อยไปมากหรือไม่</p> <pre> for k in range(len(s)-1) : if s[k] > s[k+1] : print("False") break else: print("True") </pre>
ต้องการหุบข้อมูลทุก ๆ คู่ในลิสต์ (ไม่จำเป็นต้องติดกัน) มาประมวลผล	<p>ต้องการนับว่า มีข้อมูลกี่คู่ในลิสต์ที่ตัวทางซ้ายมีค่ามากกว่าตัวทางขวา (ไม่จำเป็นต้องอยู่ติดกัน)</p> <pre> c = 0 for i in range(len(x)) : for j in range(i+1,len(x)) : if x[i] > x[j] : c += 1 print(c) </pre>
เรียงลำดับข้อมูลในลิสต์ (ด้วยวิธี bubble sort)	<p># ต้องการเรียงลำดับข้อมูลในลิสต์ d จากน้อยไปมาก</p> <pre> for k in range(len(d)-1) : for i in range(len(d)-1) : if d[i] > d[i+1] : d[i],d[i+1] = d[i+1],d[i] </pre>

เรื่องพิศบอย

ใช้ index ที่เกินช่วงที่ใช้ได้ของลิสต์ อย่าลืมว่า index ของลิสต์ x อยู่ในช่วง 0 ถึง len(x)-1 จากซ้ายไปขวา และ -1 ถึง -len(x) ถอยจากขวามาซ้าย	<pre> x = [2,3,5,7,11,13,17,19,23] s = 0 for i in range(len(x),0,-1): s += i*x[i] # ผิด ค่าแรกของ i คือ len(x) อยู่นอกช่วง </pre>
เลข index ที่ใช้กับลิสต์ไม่ใช่จำนวนเต็ม	<pre> print(x[n/2]) # ผิด เพราะ n/2 ได้จำนวนจริง </pre>

เมื่อ x เป็นลิสต์ อย่าสับสนระหว่าง x.append(y) กับ x+=y ซึ่งได้ผล ไม่เหมือนกัน	<pre>x = [1,2,3,4]; x.append([5]) # ได้ [1,2,3,4,[5]] x = [1,2,3,4]; x += [5] # ได้ [1,2,3,4,5] x = [1,2,3,4]; x.append(5) # ได้ [1,2,3,4,5] x = [1,2,3,4]; x += 5 # ผิด</pre>
ต้องการลิสต์ y ที่มีค่าเหมือนกับลิสต์ x ต้อง เลือกจะใช้ y = x หรือ y = list(x) ปกติไม่ค่อยน่าจะใช้ y = x	<pre>x = [1,2,3] y = x # y เป็นลิสต์เดียวกับ x x[2] = 0 # y[2] ก็เปลี่ยนเป็น 0 ด้วย</pre>
เขียน x = list(str) เป็นการสร้างลิสต์ ที่ประกอบด้วยแต่ละอักขระใน str	<pre>x = list('abc') ได้ x = ['a','b','c'] ไม่ใช่ x = ['abc']</pre>
นำข้อมูลในลิสต์มา join กันให้เป็นสตริง แต่ลืมไปว่าข้อมูลในลิสต์นั้นต้องเป็นสตริง ถึงจะ join ได้	<pre>x = [1,2,3] s = ','.join(x) # ผิด เพราะ 1,2,3 ไม่ใช่สตริง t = [] for e in x : t.append(str(e)) s = ','.join(t) # ใช้ได้ เพราะ t เป็นลิสต์ของสตริง</pre>
สับสนคำสั่ง x.sort() กับ sorted(x) x.sort() เรียงลำดับข้อมูลในลิสต์ x คำสั่งนี้ไม่คืนผลใด ๆ ในขณะที่ sorted(x) นำข้อมูลที่ได้จาก x มาเรียง ลำดับแล้วคืนลิสต์ใหม่ที่มีข้อมูลเหมือนใน x แต่เรียงลำดับแล้ว โดยที่ x ไม่เปลี่ยนแปลง	<pre>x = [9,2,0,4] x = x.sort() # ผิด แบบนี้ทำให้ x เก็บค่า None # เพราะ x.sort() ไม่คืนผลใด ๆ for e in x.sort() : ... # ผิด x.sort() ไม่คืนผล ไม่มีอะไรให้ e ถ้าต้องการเรียงลำดับข้อมูลใน x ใช้ x.sort() หรือ x = sorted(x) for e in sorted(x) : ... # แบบนี้ได้ sorted(x) คืนลิสต์ที่เรียงแล้ว</pre>
มีการลบหรือเพิ่มข้อมูลในลิสต์ระหว่างที่ มีการแจกแจงข้อมูลในลิสต์	<pre>x = [1,2,3,3,2,1] for e in x : if e%2 == 1 : x.pop(x.index(e)) หรือ for i in range(len(x)) : if x[i]%2 == 1 : x.pop(i) จะไม่ได้ผลตามที่คาด และเกิดข้อผิดพลาดระหว่างการทำงานด้วย (ลอง run ดู) ควรใช้การสร้างลิสต์ชั่วคราวก่อน แล้วค่อยนำกลับไปใส่ในลิสต์เดิม t = [] for e in x : if e%2 != 1 : t.append(e) x[:] = t ให้สังเกตว่าคำสั่งสุดท้ายคือ x[:] = t แทนที่จะเป็น x = t เพราะต้องการ เปลี่ยนข้อมูลในลิสต์ x ให้เหมือน t ไม่ได้ต้องการให้ x ไปอ้างอิงลิสต์เดียวกับ t</pre>

แบบฝึกหัด

Problem	Code
<p><u>Input:</u> มี 2 บรรทัด แต่ละบรรทัดเก็บสมาชิกของเวกเตอร์ ซึ่งเป็นจำนวนจริงหลายจำนวนคั่นด้วยช่องว่าง อ่านทั้งสองบรรทัดเก็บในลิสต์ v1 และ v2</p> <p><u>Process:</u> คำนวณ dot product ของเวกเตอร์ v1 กับ v2</p> <p><u>Output:</u> แสดงคำว่า Error ถ้า v1 และ v2 มีขนาดไม่เท่ากัน แต่ถ้าเท่ากัน แสดงค่า dot product ที่คำนวณได้</p>	
<p><u>Input:</u> บรรทัดแรกเป็นจำนวนเต็ม n และมีอีก n บรรทัด แต่ละบรรทัดเป็นจำนวนเต็ม 1 จำนวน</p> <p><u>Process:</u> เรียงลำดับจำนวนเต็มทั้ง n ตัวจากน้อยไปมาก</p> <p><u>Output:</u> จำนวนเต็มทั้ง n ที่เรียงจากน้อยไปมากบนบรรทัดเดียวกันเรียงจากซ้ายขวา คั่นด้วยจุลภาค ,</p>	
<p><u>Input:</u> หนึ่งบรรทัดเป็นชื่อแฟ้ม แฟ้มนี้เก็บจำนวนเต็ม บรรทัดละจำนวน</p> <p><u>Process:</u> หาว่าจำนวนเต็มใดในแฟ้มปรากฏซ้ำกันมากที่สุด ถ้ามีซ้ำกันมากที่สุดหลายตัว ให้หาทุกตัว</p> <p><u>Output:</u> ข้อมูลทุกตัวที่ปรากฏซ้ำกันมากที่สุดในแฟ้ม เรียงตามลำดับที่ปรากฏในแฟ้ม</p>	

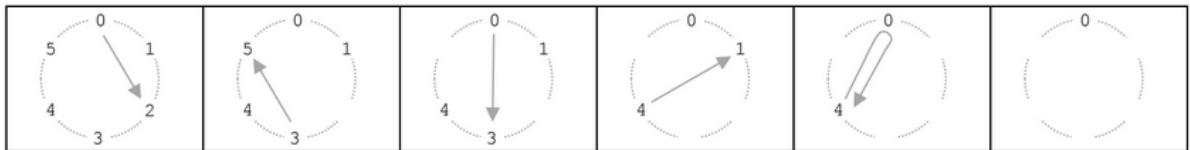
Problem	Code
<p><u>Input</u>: หนึ่งบรรทัดเป็นชื่อแฟ้ม</p> <p><u>Process</u>: แสดงหัวข้อข่าวทั้งหมดในแฟ้ม หัวข้อข่าวเป็นข้อความที่อยู่ระหว่าง <code><headline></code> กับ <code></headline></code> ในแฟ้มนี้ (ทั้ง <code><headline></code> กับ <code></headline></code> อยู่ในบรรทัดเดียวกันแน่ ๆ และแต่ละบรรทัดมีไม่เกิน 1 หัวข้อข่าว)</p> <p><u>Output</u>: หนึ่งบรรทัดหนึ่งหัวข้อ ให้ครบทุกหัวข้อ โดยแสดงเรียงหัวข้อตามตัวอักษรจากน้อยไปมาก</p>	
<p><u>Input</u>: หนึ่งบรรทัดเป็นชื่อแฟ้ม แฟ้มนี้เก็บชื่อบรรทัดละหนึ่งชื่อ</p> <p><u>Process</u>: เรียงลำดับชื่อที่อ่านจากแฟ้ม โดยเรียงลำดับตามความยาวของชื่อจากน้อยไปมาก ถ้ามีความยาวเท่ากัน ให้เรียงตามตัวอักษรแบบในพจนานุกรม</p> <p><u>Output</u>: ลำดับของชื่อตามที่เรียงได้</p>	

ตัวอย่างการแก้โจทย์ปัญหา

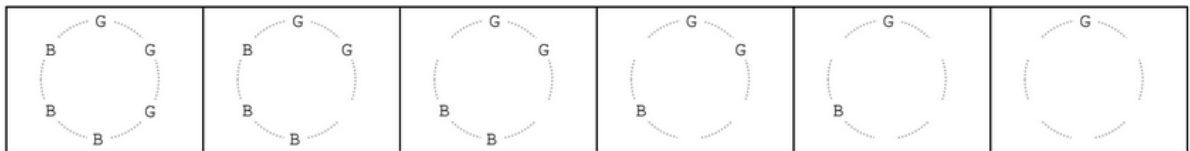
ขจัดคนเลว

กำหนดให้ n เป็นจำนวนเต็ม หากเรานำคนดี n คน กับคนเลว n คน มาเรียงเป็นวงกลม โดยให้คนดียืนเรียงติดกัน ตั้งแต่ตำแหน่งที่ 0 ถึง $n-1$ และคนเลวยืนเรียงต่อไปตั้งแต่ตำแหน่ง n ถึง $2n-1$ จงหาจำนวนเต็มบวก d ที่มีค่าน้อยสุด ที่เมื่อนำคนทั้งหมดมาใช้กับปัญหา Josephus แล้วจะเหลือคนดีเป็นคนสุดท้าย

ปัญหา Josephus เป็นดังนี้ : ข้อมูลนำเข้าคือจำนวนเต็ม m กับ d ให้มี m คนยืนเรียงเป็นวงกลม เริ่มคนที่ 0 นับไปอีก d คน ก็ให้คนนั้นออกจากวง แล้วก็เริ่มจากคนถัดไปนับไปอีก d คน ก็ให้คนนั้นออก ทำเช่นนี้ไปเรื่อย ๆ จนเหลือคนสุดท้าย คนนั้นเป็นผู้ชนะ เช่น ให้ $m = 6$ และ $d = 2$ การเปลี่ยนแปลงของคนที่ยืนในวงกลมแสดงได้ดังรูปข้างล่างนี้ หมายเลข 0 เป็นผู้ชนะ



ดังนั้นถ้าให้ $n = 3$ จะได้ค่า $d = 2$ เป็นค่าที่เมื่อขจัดคนออกแล้ว จะได้คนดีเป็นคนสุดท้าย (G แทนคนดี, B แทนคนเลว)



► ข้อมูลนำเข้า

จำนวนเต็มบวก 1 จำนวน แทนค่า n ข้างต้น

► ข้อมูลส่งออก

ค่า d ที่เป็นจำนวนบวกน้อยสุดที่ทำให้ขจัดคนออกแล้วเหลือคนสุดท้ายเป็นคนดี ดังที่อธิบายไว้ข้างต้น

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
1	1
3	2
6	3
13	4
14	6

ตัวอย่างการเขียนโปรแกรม

โปรแกรม	คำอธิบาย
<pre> n = int(input()) for d in range(1,2*n): q = ['G']*n + ['B']*n m = 2*n # solve Josephus problem if len(q)==1 and q[0]=='G': print(d) break else: print('Not found') </pre>	<p>เนื่องจากคอมพิวเตอร์ทำงานได้รวดเร็ว ขอแก้ปัญหาด้วยการจำลองการค่อย ๆ ขจัดคนในวงกลมออกตามกฎเกณฑ์ที่กำหนดไว้ โดยใช้ลิสต์เก็บคนที่ยืนในวงกลม แล้วค่อย ๆ ลบข้อมูลในลิสต์ออก</p> <p>โปรแกรมทางซ้ายนี้รับค่า n จากนั้นลู่ทดสอบการจัดคนด้วยค่า d ต่าง ๆ เริ่มที่ 1 ไปถึง $2n-1$ ภายในแต่ละรอบของ <code>for</code> จะสร้างลิสต์ของตัวอักษร G จำนวน n ตัวตามด้วยตัวอักษร B อีก n ตัว ('G' แทนคนดี, 'B' แทนคนเลว) แล้วก็เริ่มขั้นตอนการแก้ปัญหา Josephus ด้วย $m = 2*n$ และ d ตามค่าของ <code>for</code> หากผลการจัดคนในวงกลมของปัญหา Josephus เหลือคนสุดท้ายเป็นคนดี ก็แสดงค่า d และ <code>break</code> ออกจาก <code>for</code> ได้เลย ถ้าไม่ได้คนดีเป็นคนสุดท้าย ก็วนกลับไปเพิ่มค่า d เพื่อทำรอบต่อไปของ <code>for</code></p>
<pre> m = int(input()) d = int(input()) q = list(range(m)) k = 0 while len(q) > 1 : k += d if k >= len(q) : k -= len(q) q.pop(k) print(q[0]) </pre>	<p>ขอเก็บโปรแกรมข้างบนนี้ไว้ก่อน และมาเขียนโปรแกรมสำหรับปัญหา Josephus ทางซ้ายนี้ ที่รับค่าจำนวนเต็ม m กับ d แล้วสร้างลิสต์ที่เก็บหมายเลข 0 ถึง $m-1$ ด้วยคำสั่ง <code>list(range(m))</code> จากนั้นเข้าสู่ <code>while</code> ที่จะวนทำงานตราบเท่าที่ลิสต์ <code>q</code> ยังมีข้อมูลมากกว่าหนึ่งตัว มีตัวแปร k เก็บ <code>index</code> ของลิสต์ที่มีค่าเริ่มที่ 0 ในแต่ละรอบจะลบข้อมูลตัวที่ถัดจาก k ไปอีก d ตัว ซึ่งคือเพิ่มค่า k อีก d ถ้าค่า k เกินหรือเท่ากับขนาดของลิสต์ ก็ต้องวนกลับมาด้านซ้ายของลิสต์ ซึ่งก็คือการลดค่าของ k ด้วยขนาดของลิสต์ เช่น <code>q</code> มีข้อมูล 7 ตัว, $k = 5$, $d = 3$ คำสั่ง <code>k += d</code> ทำให้ $k = 8$ เกินขนาดของ <code>q</code> คำสั่ง <code>k -= len(q)</code> ทำให้ k กลับมามีค่าเป็น 1 สรุปคือ ถัดจากตำแหน่งที่ 5 ไปอีก 3 ตำแหน่งคือ 1 ($5 \rightarrow 6 \rightarrow 0 \rightarrow 1$)</p> <p>เมื่อได้ตำแหน่ง k ที่เราต้องลบข้อมูลออก ก็ทำคำสั่ง <code>q.pop(k)</code> วน <code>while</code> จะลบข้อมูลรอบละตัว เมื่อเหลือข้อมูลตัวเดียว ข้อมูลที่เหลือรอดนั้นก็คือ <code>q[0]</code></p> <p>สั่ง <code>run</code>, ใส่ m เป็น 6, d เป็น 2, ได้ผลเป็น 0 ถูกต้อง</p> <p>สั่ง <code>run</code>, ใส่ m เป็น 4, d เป็น 3, ทำงานผิด</p> <p><code>IndexError: pop index out of range</code></p> <p>ที่คำสั่ง <code>q.pop(k)</code> แปลว่า k เก็บ <code>index</code> ที่มีค่าเกินช่วงที่ลิสต์ <code>q</code> มีให้ลบ ถ้าลองแทรกคำสั่ง <code>print(k, len(q))</code> จะได้ 3 3 นั่นคือลิสต์ที่มี 3 ตัว ลบตัวที่ <code>index 3</code> ไม่ได้ เพราะมีให้ใช้แค่ <code>index 0</code> ถึง 2 เท่านั้น</p>
<pre> m = int(input()) d = int(input()) q = list(range(m)) k = 0 while len(q) > 1 : k = (k + d) % len(q) q.pop(k) print(q[0]) </pre>	<p>แก้ปัญหาข้างต้นได้ด้วยการใช้คำสั่ง $k = (k + d) \% \text{len}(q)$ เพื่อคำนวณ <code>index</code> ที่ถัดจาก k ไปอีก d ช่อง กรณีที่ <code>index</code> เลยไปทางขวาของลิสต์ การ <code>mod</code> ด้วยขนาดของลิสต์ จะได้ผลวนกลับมาทางซ้าย เช่น ถ้าลิสต์มีขนาด 4 ตัว ถัดจากตำแหน่ง 3 ไปอีก 5 ตำแหน่งก็คือ $(3+5) \% 4 = 0$</p> <p>สั่ง <code>run</code>, ใส่ m เป็น 6, d เป็น 2, ได้ผลเป็น 0 ถูกต้อง</p> <p>สั่ง <code>run</code>, ใส่ m เป็น 4, d เป็น 3, ได้ผลเป็น 1</p> <p>ลองจำลองการทำงานได้</p> <p>0,1,2,3 \rightarrow 0,1,2 \rightarrow 1,2 \rightarrow 1 ถูกต้อง</p>

โปรแกรม	คำอธิบาย
<pre> n = int(input()) for d in range(1,2*n) : q = ['G']*n + ['B']*n k = 0 while len(q) > 1 : k = (k + d) % len(q) q.pop(k) if q[0] == 'G' : print(d) break else: print('Not found') </pre>	<p>นำชุดคำสั่งในการหาคำตอบของปัญหา Josephus ข้างต้นไปแทรกในโปรแกรมตอนแรกที่เขียนได้โปรแกรมทางซ้ายนี้</p> <p>สั่ง run, ใส่ 1, ได้ 1, ถูกต้อง</p> <p>สั่ง run, ใส่ 3, ได้ 2, ถูกต้อง</p> <p>สั่ง run, ใส่ 6, ได้ 3, ถูกต้อง</p> <p>สั่ง run, ใส่ 13, ได้ 4, ถูกต้อง</p> <p>สั่ง run, ใส่ 14, ได้ 6, ถูกต้อง</p>

06.2 : Nested List

สรุปเนื้อหา

ลิสต์เก็บอะไรก็ได้ จำนวนเต็ม จำนวนจริง สตริง หรืออื่น ๆ หรือแม้กระทั่งลิสต์ ก็ได้ ให้ `x` เป็นลิสต์ และถ้า `x[k]` ก็เป็นลิสต์ จะได้ `x[k][j]` คือข้อมูลตัวที่ index `j` ของลิสต์ `x[k]`

```
x = []
x.append( 1 )           # [1]
x.append( [2,3] )       # [1,[2,3]]
x.append( [4,5,6] )     # [1,[2,3],[4,5,6]]
x.append( [[7,8]] )     # [1,[2,3],[4,5,6],[[7,8]]]
x.append( [] )          # [1,[2,3],[4,5,6],[[7,8]],[]]
len(x) คือ 5, x[1][1] คือ 3, x[2][1] คือ 5, x[3][0][1] คือ 8, len(x[3][0]) คือ 2, len(x[4]) คือ 0
```

รูปแบบการประมวลผลลิสต์ซ้อนลิสต์ที่พบบ่อย

ใช้ลิสต์ซ้อนลิสต์แทนการเก็บลิสต์ของข้อมูล โดยที่ข้อมูลแต่ละตัวเป็นลิสต์ที่เก็บข้อมูลย่อย ๆ จำนวนเท่า ๆ กัน

ใช้ลิสต์ซ้อนลิสต์เก็บข้อมูลของวงกลมหลาย ๆ วง ลิสต์ข้างในคือจำนวนสามจำนวนที่แทนพิกัด `x`, `y` และรัศมีของวงกลม เช่น

```
[[0.0, 0.0, 10.0], [1.0, 5.0, 3.0]]
```

โปรแกรมข้างล่างนี้อ่านข้อมูลวงกลมจากแฟ้มมาเก็บในลิสต์

```
circles = []
file1 = open('c:/temp/circles.txt')
for line in file1 :
    x,y,r = line.split()
    circles.append( [float(x),float(y),float(r)] )
```

ข้างล่างนี้หาวงกลมที่ไม่ทับหรือแตะวงกลมอื่นเลย

```
free = []
for i in range(len(circles)) :
    for j in range(i+1,len(circles)) :
        dx = circles[i][0] - circles[j][0]
        dy = circles[i][1] - circles[j][1]
        sumr = circles[i][2] + circles[j][2]
        if dx**2 + dy**2 <= sumr**2 : break
    else :
        free.append(circles[i])
```

```
for c in free :
    print(c)
```

<p>ใช้ลิสต์ซ้อนลิสต์ สร้างเมทริกซ์ ถ้าต้องการเมทริกซ์ขนาด $m \times n$ ก็ สร้างลิสต์ A ที่ <code>len(A)</code> มีค่าเท่ากับ m (จำนวนแถว) และ <code>len(A[0]) = len(A[1])</code> <code>= ... = len(A[m-1]) = n</code> (จำนวนคอลัมน์) เช่นสร้างเมทริกซ์ ที่มีค่า 0 ทั้งหมดขนาด 5×3 <code>A = []</code> <code>for k in range(5):</code> <code> A.append([0]*3)</code></p>	<p>ตัวอย่างการหาผลบวกของเมทริกซ์ A กับ B</p> $A = \begin{bmatrix} [1,2], & [1,1] \\ [0,1], & [4,3] \end{bmatrix} \quad A + B = \begin{bmatrix} 1 & 2 \\ 1 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 4 & 3 \end{bmatrix}$ <pre> C = [] for i in range(len(A)) : C.append([0]*len(A[i])) for j in range(len(A[i])) : C[i][j] = A[i][j] + B[i][j]</pre>
<p>ใช้ลิสต์ซ้อนลิสต์ โดยที่ลิสต์ข้างในเก็บ ข้อมูลที่ไม่จำเป็นต้องมีขนาดเท่ากัน</p>	<p>ใช้ลิสต์ซ้อนลิสต์ เก็บข้อมูลที่ประกอบด้วย username (ที่ช่อง 0), และลิสต์ของ usernames รายอื่นที่ขอติดตามข่าวสาร (ที่ช่อง 1 โดยลิสต์นี้ไม่จำเป็นต้องมี ขนาดคงตัว) (ตัวอย่างนี้ซ้อนตั้งสามชั้น) เช่น</p> <pre> f = [['noon',['pat','koi']], ['wii',['noon','koi']], ['pat',['koi','noon','wii']], ['koi',[]]]</pre> <p>ต้องการหาว่า username ใดมีคนติดตามเป็นจำนวนมากที่สุด</p> <pre> maxindex = 0 for k in range(1,len(f)) : if len(f[k][1]) > len(f[maxindex][1]) : maxindex = k print(f[maxindex][0], 'has max.# of followers :', \ ', '.join(f[maxindex][1]))</pre> <p>หรือเขียน</p> <pre> max_followers = [] max_username = '' for [username,followers] in f : if len(followers) > len(max_followers) : max_username = username max_followers = followers print(max_username, 'has max.# of followers :', \ ', '.join(max_followers))</pre>

ใช้ลิสต์ซ้อนลิสต์เป็นที่เก็บข้อมูลชั่วคราวเพื่อนำไป sort ตามข้อกำหนดที่ต้องการ โดยสร้างลิสต์ข้างในให้มีสมาชิกตัวแรกเป็นเกณฑ์ที่ใช้ในการ sort

หมายเหตุ : การ sort ลิสต์ซ้อนลิสต์จะเปรียบเทียบความน้อยกว่าของข้อมูลของลิสต์ข้างในทีละตัวจากซ้ายไปขวา เช่น

```
x=[[3,2],[3,1],[9],[2,5],[3]]
x.sort() จะได้ x เปลี่ยนเป็น
[[2,5],[3],[3,1],[3,2],[9]]
นั่นคือ
[2,5] < [3] < [3,1] < ...
```

จากตัวอย่างก่อนหน้านี้ เมื่อมี f แล้ว ถ้าต้องการเรียงลำดับ usernames ทั้งหลายตามจำนวนผู้ติดตาม ก็ใช้

```
c = []
for [username,followers] in f :
    c.append([len(followers),username])
c.sort()
for [x,username] in c :
    print(username)
```

จากตัวอย่างบน

```
f = [ ['noon','pat','koi']], ['wii',['noon','koi']],
      ['pat',['koi','noon'],'wii']], ['koi',[]] ]
```

จะได้ c = [[2,'noon'],[2,'wii'],[3,'pat'],[0,'koi']]
c.sort() ได้ [[0,'koi'],[2,'noon'],[2,'wii'],[3,'pat']]

หากต้องการให้ sort ด้วยเกณฑ์ที่ซับซ้อนขึ้น เช่น ให้เรียงตามจำนวนผู้ติดตามจากน้อยไปมาก และในกรณีที่จำนวนผู้ติดตามเท่ากัน ให้เรียงตาม username จากน้อยไปมากเช่นกัน แบบที่ต้องการนี้ทั้งสองข้อมูลย่อยในลิสต์เรียงแบบน้อยไปมาก สามารถใช้ sort() ได้เลย แต่ถ้าเปลี่ยนเป็น ให้เรียงตามจำนวนผู้ติดตามจากมากไปน้อย สำหรับกรณีที่จำนวนผู้ติดตามเท่ากัน ให้เรียงตาม username จากน้อยไปมาก ก็อาจใช้กลวิธีเล็กน้อย เช่น ดิดลบจำนวนที่ต้องการเรียงจากมากไปน้อย การเรียงเลขลบจากน้อยไปมาก ก็คือเรียงเลขบวกจากมากไปน้อย

```
c = []
for [username,followers] in f :
    c.append([-len(followers),username])
c.sort()
```

จากตัวอย่างบน

```
f = [ ['noon','pat','koi']], ['wii',['noon','koi']],
      ['pat',['koi','noon'],'wii']], ['koi',[]] ]
```

จะได้ c = [[-2,'noon'],[-2,'wii'],[-3,'pat'],[0,'koi']]
sort ได้ [[-3,'pat'],[-2,'noon'],[-2,'wii'],[0,'koi']]

เรื่องพิศบอย

การเพิ่มลิสต์ย่อย a เข้าไปในลิสต์ใหญ่ x ต้องใช้ x.append(a) ไม่ใช่
x = x + a หรือ x += a

```
x = [[2,3],[3,4]]
```

ได้มาจากการ append [2,3] กับ [3,4] เข้า x

```
x = []
```

```
x.append( [2,3] )
```

```
x.append( [3,4] )
```

แต่ไม่เหมือนคำสั่งข้างล่างนี้

```
x = []
```

```
x = x + [2,3] + [3,4] # ได้ [2,3,3,4]
```

อย่าสร้างลิสต์ซ้อนลิสต์ด้วย * อย่าเขียน [a]*n เมื่อ a เป็นลิสต์	<p>ต้องการสร้างเมทริกซ์ที่แทนด้วยลิสต์ซ้อนลิสต์ ที่มีขนาด 3x3 มีค่าเป็น 0 ทั้งหมด</p> <pre>x = [[0,0,0]] * 3</pre> <p>print(x) จะได้ [[0,0,0],[0,0,0],[0,0,0]]</p> <p>แต่ลิสต์ข้างในทั้งสามตัวนี้คือลิสต์ตัวเดียวกัน ถ้าสั่ง x[0][1] = 1 ทำงาน x จะเปลี่ยนเป็น [[0,1,0],[0,1,0],[0,1,0]]</p> <p>ใช้ x = [[0]*3]*3 ก็ได้ผลที่แปลกแบบข้างบนเหมือนกัน</p> <p>เราต้องสร้างลิสต์ข้างในให้เป็นคนละตัว โดยเขียน</p> <pre>x = [[0,0,0], [0,0,0], [0,0,0]]</pre> <p>หรือใช้วงวนสร้าง</p> <pre>n = 3 x = [] for i in range(n): x.append([0]*n)</pre> <p>สรุปคือ เมื่อใดเขียน [a]*n ต้องระวัง ถ้า a เป็นลิสต์ ให้ใช้วงวนสร้างแทน</p>
--	---



Problem	Code
<p>Input: บรรทัดแรกมีจำนวนเต็ม r กับ c ตามด้วยอีก r บรรทัด แต่ละบรรทัดมีจำนวนเต็ม c ตัว</p> <p>Process: สร้างเมทริกซ์ด้วยลิสต์ซ้อนลิสต์ ถ้ามีบรรทัดที่มีข้อมูลไม่ใช่ c ตัว ให้แสดงผลเป็นลิสต์ซ้อนลิสต์ว่าง [[]]</p> <p>Output: เมทริกซ์ลิสต์ซ้อนลิสต์ที่สร้างได้</p>	
<p>Input: หนึ่งบรรทัดเป็นชื่อแฟ้ม แฟ้มนี้มีหลายบรรทัดเท่ากับจำนวน username แต่ละบรรทัดประกอบด้วย username ตามด้วย usernames อื่น ๆ ที่ติดตามข่าวสารของ username แรกต้นบรรทัด เช่น</p> <pre>noon pat koi wii noon pat koi noon wii koi</pre> <p>Process: สร้างลิสต์ซ้อนลิสต์ที่มีรูปแบบตามตัวอย่างนี้</p> <pre>f = [['noon', ['pat', 'koi']], ['wii', ['noon']], ['pat', ['koi', 'noon', 'wii']], ['koi', []]]</pre> <p>Output: ลิสต์ที่สร้างได้</p>	

Problem	Code
<p>Input: จาก f ที่ได้ในข้อที่แล้ว</p> <p>Process: ต้องการรู้ว่า ใครบ้างที่ไม่มีใครติดตามเลย</p> <p>Output: รายชื่อของผู้ที่ไม่มีใครติดตามเลย</p>	
<p>Input: บรรทัดแรกมีจำนวนเต็ม n และอีก n บรรทัดที่ตามมาเป็นสตริง</p> <p>Process: เรียงลำดับสตริงที่อ่านเข้ามา จากนั้นน้อยไปมากตามความยาวสตริง ถ้าสตริงยาวเท่ากันให้เรียงตามตัวสตริงเอง เช่น 'xyz', 'xy', 'abc' เรียงแล้วได้ 'xy', 'abc', 'xyz'</p> <p>Output: สตริงที่เรียงแล้ว บรรทัดละสตริง</p>	



เรียงตามคะแนนรวม

จงเขียนโปรแกรมรับรหัสนักเรียนและรายการของคะแนนการสอบย่อยต่าง ๆ ของนักเรียนจำนวนหนึ่ง มาประมวลผลเพื่อแสดงรหัสนักเรียนและคะแนนรวมตามลำดับคะแนนรวมจากมากไปน้อย

► ข้อมูลนำเข้า

บรรทัดแรกมีจำนวนเต็มบวก n กับ m (คั่นด้วยช่องว่าง) n คือจำนวนนักเรียน และ m คือจำนวนการสอบย่อยของนักเรียนแต่ละคน n บรรทัดต่อมา แต่ละบรรทัดประกอบด้วย รหัสนักเรียนตามด้วยรายการของคะแนนย่อย คั่นด้วยช่องว่าง

► ข้อมูลส่งออก

ถ้ามีบรรทัดที่จำนวนคะแนนย่อยไม่ตรงกับ m ให้รวบรวมรหัสนักเรียนมาแสดงตามตัวอย่าง (เรียงตามที่ได้รับจากข้อมูลนำเข้า)

ถ้าทุกบรรทัดมีคะแนนครบจำนวนทุกคน ให้แสดงรหัสนักเรียนตามด้วยคะแนนรวมของนักเรียนบรรทัดละคน เรียงลำดับตามคะแนนรวมจากมากไปน้อย ในกรณีที่คะแนนรวมเท่ากัน ให้เรียงตามรหัสนักเรียนจากน้อยไปมาก

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
3 2 A 3.0 2.0 B 4.0 5.0 C 2.0 3.0	B 9.0 A 5.0 C 5.0
3 3 AA 4.0 B 9.0 8.0 7.0 AC 8.0 5.0	Invalid data: AA AC

ตัวอย่างการเขียนโปรแกรม

ขอเก็บข้อมูลในลิสต์ แบบลิสต์ซ้อนลิสต์ ลิสต์ข้างในแต่ละลิสต์เก็บรหัสนักเรียนที่ช่อง 0 ตามด้วยคะแนนตั้งแต่ช่องที่ 1 ถึง m
 [[รหัสนักเรียน, คะแนน, ..., คะแนน], [รหัสนักเรียน, คะแนน, ..., คะแนน], ...]
 ขอแบ่งการประมวลผลเป็นขั้นตอนที่ละเอียดดังนี้

1. อ่านข้อมูลนำเข้าเก็บเป็นลิสต์ซ้อนลิสต์ เนื่องจากข้อมูลที่เข้ามา รหัสเป็นสตริง ส่วนคะแนนต้องการเก็บเป็นจำนวนจริง แต่ขอเริ่มด้วยการอ่านเข้ามาเป็นสตริงให้หมดก่อน ถ้าใช้ข้อมูลนำเข้าของตัวอย่างที่สอง ได้
`d0 = [['AA', '4.0'], ['B', '9.0', '8.0', '7.0'], ['AC', '8.0', '5.0']]`
2. นำ d0 มาหาว่ารหัสนักเรียนใดที่มีจำนวนคะแนนไม่เท่ากับจำนวนที่กำหนดให้ (ตัวอย่างที่สองระบุว่าต้องมี 3 ข้อ) ได้ `err = ['AA', 'AC']`
3. ถ้า err มีขนาดเกิน 0 ก็แสดงว่ามีที่ผิด จึงแสดง Invalid data ตามด้วยรหัสนักเรียนที่เก็บใน err
4. ถ้า err มีขนาดเป็น 0 (คือไม่มีผิดเลย) ประมวลผลต่อ (คราวนี้ขอใช้ข้อมูลจากตัวอย่างแรก เพราะไม่มีที่ผิด) ได้ `d0 = [['A', '3.0', '2.0'], ['B', '4.0', '5.0'], ['C', '2.0', '3.0']]` และ `err = []`
 - 4.1. นำ d0 จากขั้นตอนที่แล้ว เปลี่ยนคะแนนให้เป็นจำนวนจริง แล้วหาผลรวม นำมาเก็บคู่กับรหัสนักเรียน ได้ `d1 = [[5.0, 'A'], [9.0, 'B'], [5.0, 'C']]`
 - 4.2. เรียงลำดับข้อมูลใน d1 ตามโจทย์ คือคะแนนรวมจากมากไปน้อย ถ้าคะแนนรวมเท่ากัน เรียงตามรหัส จากนั้นน้อยไปมาก ได้ `d1 = [[9.0, 'B'], [5.0, 'A'], [5.0, 'C']]`
 - 4.3. นำข้อมูลใน d1 มาแสดงเป็นผลลัพธ์

โปรแกรม	คำอธิบาย
<pre>n,m = [int(e) for e in input().split()] d0 = list() for k in range(n) : d0.append(input().split())</pre>	<p>ขั้นตอนที่ 1 : อ่านค่า n (จำนวนนักเรียน) กับ m (จำนวนคะแนนต่อคน) ใช้วงวน for วงจำนวน n รอบ อ่านจากแป้นพิมพ์แล้ว split ได้เป็นลิสต์ของสตริง เพิ่มแต่ละลิสต์ที่ได้เข้าในลิสต์ d0</p> <p>สั่ง run, ใส่ข้อมูลตามตัวอย่างที่สอง, แล้ว print(d0) ได้</p> <pre>['AA', '4.0'], ['B', '9.0', '8.0', '7.0'], ['AC', '8.0', '5.0']]</pre>
<pre>err = list() for x in d0 : if len(x[1:]) != m : err.append(x[0]) if len(err) > 0 : print('Invalid data: ') for sid in err : print(sid)</pre>	<p>ขั้นตอนที่ 2, 3 : นำแต่ละลิสต์ x ข้างในลิสต์ d0 จากขั้นตอนที่แล้ว มาตรวจ ถ้า x[1:] มีขนาดไม่เท่ากับ m คือมีจำนวนคะแนนไม่เท่ากับที่กำหนด จะนำ x[0] ใส่เพิ่มในลิสต์ err หลังจากสร้างเสร็จ ถ้า err มีขนาดเกิน 0 ก็แสดงรหัสนักเรียนที่มีความผิดพลาด</p> <p>สั่ง run, ใส่ข้อมูลตามตัวอย่างที่สอง, ได้</p> <p>Invalid data:</p> <pre>AA AC</pre>
<pre>else : d1 = list() for x in d0 : s = 0 for e in x[1:] : s += float(e) d1.append([s, x[0]])</pre>	<p>ขั้นตอนที่ 4.1 : เป็นกรณีที่มีคะแนนครบ แจกแจงลิสต์ x ข้างใน d0 ส่วนที่เก็บคะแนน ซึ่งคือ x[1:] มาหาผลรวม โดยต้องแปลงเป็น float ก่อน จากนั้นนำผลรวมมาต่อกับ x[0] ได้ลิสต์ย่อยเก็บใส่ลิสต์ใหม่ d1</p> <p>สั่ง run, ใส่ข้อมูลตามตัวอย่างแรก, แล้ว print(d1) ได้</p> <pre>[[5.0, 'A'], [9.0, 'B'], [5.0, 'C']]</pre>

โปรแกรม	คำอธิบาย
<pre> for k in range(len(d1)-1) : for i in range(len(d1)-1) : if d1[i][0] < d1[i+1][0] \ or (d1[i][0]==d1[i+1][0] \ and d1[i][1]>d1[i+1][1]) : d1[i],d1[i+1] = \ d1[i+1],d1[i] for [total,sid] in d1 : print(sid, total) </pre>	<p>ขั้นตอนที่ 4.2 และ 4.3 : ต้องการเรียงลำดับตามคะแนนรวม แต่ใช้คำสั่ง <code>d1.sort()</code> ไม่ได้ เพราะจะเรียงข้อมูลจากน้อยไปมาก สิ่งที่เราต้องการคือเรียงตามคะแนนจากมากมาน้อย แต่ใช้คำสั่ง <code>d1.sort(reverse=True)</code> ไม่ได้ เพราะถ้ากรณีที่คะแนนเท่ากัน จะเรียงรหัสนักเรียนจากมากมาน้อย ซึ่งไม่ตรงที่โจทย์ต้องการ</p> <p>จึงขอเขียนการเรียงลำดับเอง (ด้วยวิธี bubble sort ที่เรียนมา) โดยปรับคำสั่งการเปรียบเทียบ คือ จะสลับข้อมูลสองตัวที่ติดกันเมื่อ</p> <ul style="list-style-type: none"> • คะแนนรวมของตัวซ้ายน้อยกว่าคะแนนรวมของตัวขวา หรือ • ถ้าคะแนนรวมของทั้งสองตัวเท่ากัน รหัสนักเรียนของตัวซ้ายมากกว่ารหัสนักเรียนของตัวขวา <p>เมื่อเรียงลำดับเสร็จ ก็นำข้อมูลใน <code>d1</code> มาแสดงตามที่โจทย์กำหนด สั่ง <code>run</code>, ใส่ข้อมูลตามตัวอย่างแรก, ได้ผลถูกต้อง</p> <p>B 9.0 A 5.0 C 5.0</p>
<pre> else : d1 = list() for x in d0 : s = 0 for e in x[1:] : s += float(e) d1.append([-s, x[0]]) d1.sort() for sumsc,sid in d1 : print(sid, -sumsc) </pre>	<p>สำหรับขั้นตอนที่ 4.1 ถึง 4.3 เราอาจเปลี่ยนการจัดเก็บข้อมูลของ <code>d1</code> คือแทนที่ลิสต์ข้างในจะเป็น [คะแนนรวม, รหัสனிสิต] เปลี่ยนมาเก็บเป็น [-คะแนนรวม, รหัสனிสิต] แล้วใช้คำสั่ง <code>d1.sort()</code> เพื่อเรียงข้อมูลจากน้อยไปมาก เมื่อคะแนนรวมมีค่ามากติดลบคะแนนรวมจะมีค่าน้อย กลายเป็นต้องการเรียงคะแนนรวมจากมากไปน้อย และถ้าเท่ากัน ก็เรียงรหัสனிสิตจากน้อยไปมาก ตามต้องการ ตอนแสดงคะแนนรวม ก็ต้องแสดงค่าติดลบของค่าที่เก็บด้วย (เพราะที่เก็บเป็นค่าติดลบ)</p>

ลองเขียนโปรแกรมนี้นี้ใหม่ ถ้าเราเปลี่ยนการจัดเก็บข้อมูลจาก

[[รหัสนักเรียน, คะแนน, ..., คะแนน], [รหัสนักเรียน, คะแนน, ..., คะแนน], ...]

เป็น

[[รหัสนักเรียน, [คะแนน, ..., คะแนน]], [รหัสนักเรียน, [คะแนน, ..., คะแนน]], ...]

ดร. วิจารณ์ จิรพัฒน์กุล

Intania 87



กรรมการผู้จัดการ
บริษัท สคูติโอ จำกัด,
อดีตนักวิทยาศาสตร์ข้อมูล (Data Scientist) ที่ Facebook

ความสามารถในการเขียนโปรแกรมเป็นเหมือนพลังวิเศษที่ช่วยเพิ่มขีดความสามารถในการทำงานของเรา ไม่ว่าจะเป็นการแก้ปัญหาทางวิศวกรรมที่ซับซ้อน (simulation, optimization) การสั่งให้คอมพิวเตอร์ทำงานที่ซ้ำซากน่าเบื่อแทนเรา (automation) การสอนให้คอมพิวเตอร์มีความเฉลียวฉลาดและช่วยเราตัดสินใจได้ (artificial intelligence) รวมไปถึงการสร้างเว็บหรือแอปที่เป็นประโยชน์กับผู้คนในวงกว้าง เรียกได้ว่าเป็นทักษะที่ขาดไม่ได้สำหรับนวัตกรรมในยุคนี้

06.3 : List Comprehension

สรุปเนื้อหา

List comprehension เป็นวิธีการสร้างลิสต์ที่เขียนได้สั้นและทำงานได้รวดเร็ว

รูปแบบการสร้างลิสต์ด้วย list comprehension ที่พบบ่อย

สร้างลิสต์ด้วยวงวนเพิ่มข้อมูล	สร้างลิสต์ด้วย list comprehension
map: นำข้อมูลจากลิสต์หนึ่งมาประมวลผลเก็บใส่อีกลิสต์ เช่น สร้างลิสต์ b เก็บเฉพาะหลักหน่วยของจำนวนในลิสต์ a <pre>b = [] for e in a : b.append(e%10)</pre>	map: <pre>b = [e%10 for e in a]</pre>
filter: เลือกข้อมูลจากลิสต์หนึ่งมาใส่อีกลิสต์ เช่น สร้างลิสต์ b เก็บสตริงจากลิสต์ a เฉพาะตัวที่ยาวเกิน 5 <pre>b = [] for e in a : if len(e) > 5 : b.append(e)</pre>	filter: <pre>b = [e for e in a if len(e)>5]</pre>
map & filter: ผสมการประมวลผลสองแบบ เช่น นำข้อมูลความสูง (เป็นนิ้ว) ที่เกิน 10 ในลิสต์ a มาแปลงเป็นเซนติเมตรเก็บใส่ลิสต์ b <pre>b = [] for e in a : if e > 10 : b.append(2.54*e)</pre>	map & filter: <pre>b = [2.54*e for e in a if e>10]</pre>
สร้างลิสต์ที่มีการแจกแจงด้วยหลายวงวนซ้อนกันก็ได้ เช่น เลือกจำนวนเต็มที่มีค่าระหว่าง 0 ถึง 20 สองตัวที่มีผลรวมเท่ากับผลคูณ <pre>c = [] for a in range(0,21) : for b in range(a,21) : if a+b == a*b : c.append([a,b])</pre>	<pre>c = [[a,b] for a in range(0,21) \ for b in range(a,21) \ if a+b == a*b]</pre>

ตัวอย่าง list comprehension

```
x = [e for e in a] เหมือนกับ x = list(a)  
ได้ลิสต์ x เป็นลิสต์ใหม่มีค่าภายในเหมือนกับของ a x กับ a เป็นลิสต์คนละตัวกันแต่มีค่าเหมือนกัน  
แต่ถ้าเขียน x = a จะได้ x กับ a เป็นลิสต์เดียวกัน การเปลี่ยนค่าในลิสต์ x จะทำให้ a เปลี่ยนด้วย  
หรือการเปลี่ยนค่าในลิสต์ a ก็เปลี่ยน x เช่นกัน
```

นำแต่ละสตริงในลิสต์มาเปลี่ยนเป็นจำนวนเต็ม เก็บใส่ลิสต์ x

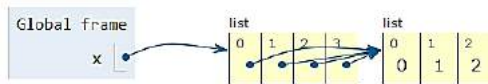
ใช้ list comprehension สร้างลิสต์ของสตริงที่นำข้อมูลจากลิสต์ x มาแปลง แล้วส่งผลลัพธ์ที่ได้ไป join กันอีกที

นับว่าลิสต์ x มีจำนวนคี่ตัว โดยสร้างลิสต์ที่เพิ่มเลข 1 ทุกครั้งที่พบจำนวนคี่ในลิสต์ x แล้วก็หาผลรวมด้วย sum

จึงเป็นการสร้างลิสต์ b โดยให้ $b[i] = 1$ ถ้า $x[i] \geq 0$ ไม่เช่นนั้นให้ $b[i] = -1$

$$x = [[0,1,2]] * 4$$

จะได้ x มีค่าเป็น $[9, 1, 2], [9, 1, 2], [9, 1, 2]$ ในขณะที่แบบอื่น ๆ ลิสต์ข้างในเป็นคนละตัวกันหมด


$$x = \begin{bmatrix} [0,1,2] \end{bmatrix} * 4$$

ปิดท้ายด้วยการใช้ list comprehension หยิบข้อมูลตามลำดับใน a มาสร้างลิสต์ใหม่ที่เพิ่มแค่พิกัด x, y เก็บใส่ d

เรื่องพิดบอย

<p>ไม่ควรใช้ list comprehension เพื่อให้ทำงานแบบวงวน แต่ไม่ได้นำผลจากการสร้างลิสต์ด้วย list comprehension ไปใช้งานเลย</p>	<p>ไม่ควรเขียน <code>[print(k) for k in range(5)]</code> ถึงแม้ว่าจะแสดงค่า 0 1 2 3 4 (บรรทัดละค่า) ตามต้องการ เนื่องจากมีการสร้างลิสต์ แล้วไม่ได้ใช้ เสียเวลาเปล่า ๆ หรือเขียน <code>[t.append(e) for e in x if e not in t]</code> เพื่อนำข้อมูลใน x ที่ไม่ปรากฏใน t ไปเพิ่มใน t จะเกิดการสร้างลิสต์ <code>[None, None, ...]</code> แล้วทิ้งไป ควรเขียนแบบวงวนปกติ</p> <pre>for e in x : if e not in t : t.append(e)</pre>
<p>อย่านำค่าของตัวแปรใน for ของ list comprehension มาใช้งานนอกคำสั่ง อาจทำให้สับสน</p>	<pre>x = [e for e in range(10)] print(e) # ผิด เพราะ e ไม่มีค่า</pre> <p>แต่ถ้าเขียน</p> <pre>x = [] for e in range(10) : x.append(e) print(e) # ได้ 9</pre> <p>แต่ถ้าแบบนี้</p> <pre>e = 99 x = [e for e in range(10)] print(e) # ได้ 99</pre>

แบบฝึกหัด

Problem	Code
<p><u>Input</u>: มีลิสต์ x เก็บสตริง และตัวแปร c เก็บตัวอักษร</p> <p><u>Process</u>: สร้างลิสต์ d เก็บจำนวนครั้งที่ตัวอักษรใน c ปรากฏในแต่ละสตริงของลิสต์ x</p> <p>เช่น <code>x = ['abba', 'babana', 'ann']; c = 'a'</code></p> <p>จะได้ <code>d = [2,3,1]</code></p>	
<p><u>Input</u>: ลิสต์ x เก็บจำนวนเต็ม</p> <p><u>Process</u>: ลบจำนวนเต็มใน x ทุกตัวที่ติดลบ</p>	

Problem	Code
<p>Input: x เป็น list of lists of integers</p> <p>Process: หาผลรวมของจำนวนเต็มใน x</p>	
<p>Input: รับหนึ่งบรรทัดที่มีจำนวนเต็มหลายจำนวน (คั่นด้วยช่องว่าง) จากแป้นพิมพ์</p> <p>Process: หาว่าที่รับมามีจำนวนติดต่อกี่จำนวน</p>	
<p>Input: รับข้อความหนึ่งบรรทัดจากแป้นพิมพ์</p> <p>Process: ตัดอักขระทุกตัวในข้อความที่รับมาที่ไม่ใช่พยัญชนะภาษาอังกฤษ</p>	
<p>Input: รับหนึ่งบรรทัดที่มีจำนวนเต็มหลายจำนวน (คั่นด้วยช่องว่าง) ชุดหนึ่ง เก็บใส่ลิสต์ x และรับอีกหนึ่งบรรทัดเก็บใส่ลิสต์ y ในทำนองเดียวกัน โดยลิสต์ทั้งสองมีจำนวนข้อมูลเท่ากัน</p> <p>Process: สร้างลิสต์ z โดยที่ $z[i]$ มีค่าเท่ากับ $x[i]+y[i]$</p>	
<p>Input: เมทริกซ์ m แทนด้วย list of lists of integers</p> <p>Process: แปลง m ให้กลายเป็น list of integers เช่น จาก $m = [[1,2,3],[4,5,6]]$ กลายเป็น $[1,2,3,4,5,6]$</p>	
<p>Input: รับหนึ่งบรรทัดที่มีจำนวนเต็มหลายจำนวน (คั่นด้วยช่องว่าง) เก็บใส่ลิสต์ x</p> <p>Process: สร้างลิสต์ใหม่จากข้อมูลใน x แต่ไม่มีตัวซ้ำ โดย</p> <ol style="list-style-type: none"> เรียงลำดับข้อมูลใน x จากน้อยไปมาก สร้างลิสต์ใหม่โดยนำข้อมูลในลิสต์ตัวที่ติดกันมาพิจารณาจากซ้ายไปขวา ถ้าค่าของตัวติดกันตัวซ้ายไม่เท่ากับตัวขวาให้นำตัวซ้ายเก็บใส่ลิสต์ใหม่ นำตัวขวาสุดใน x เพิ่มต่อท้ายลิสต์ใหม่นี้ <p>ลิสต์ใหม่นี้ก็จะเก็บข้อมูลใน x ที่ไม่มีตัวซ้ำ</p>	
<p>Input: รับจำนวนเต็ม n</p> <p>Process: สร้างลิสต์ c ที่เก็บจำนวนประกอบที่มีค่าน้อยกว่า n เริ่มจากสร้างลิสต์ใหม่ x โดย</p> <ul style="list-style-type: none"> เพิ่มค่า 4,6,8,10,... (ไม่เกิน n-1) ใน x เพิ่มค่า 6,9,12,15,... (ไม่เกิน n-1) ใน x เพิ่มค่า 8,12,16,20,... (ไม่เกิน n-1) ใน x ... <p>ไปเรื่อย ๆ トラバเท่าที่ยังไม่เกิน n-1 นั่นคือ เพิ่มค่า 2k,3k,4k,5k,... (ไม่เกิน n-1) ใน x โดยแปรค่า $k = 2,3,4,\dots,N//2-1$</p> <p>จากนั้นสร้างลิสต์ใหม่ c ที่ได้ข้อมูลจาก x แต่ไม่มีตัวซ้ำ (ใช้วิธีที่เขียนในข้อที่แล้ว)</p>	

Problem	Code
<p>Input: รับจำนวนเต็ม n</p> <p>Process: สร้างลิสต์ที่เก็บจำนวนเฉพาะที่มีค่าน้อยกว่า n โดย</p> <ol style="list-style-type: none"> 1. สร้างลิสต์ที่เก็บจำนวนประกอบที่มีค่าน้อยกว่า n 2. สร้างลิสต์ที่เก็บจำนวนเต็มจาก 2 ถึง n-1 แต่ไม่เอาจำนวนประกอบที่สร้างไว้ในขั้นตอนที่ 1 	



เรียงตามคะแนนรวม

ขอใช้ปัญหาเรียงคะแนนรวมในหัวข้อลิสต์ซ้อนลิสต์ มาเป็นตัวอย่าง โดยนำโปรแกรมที่ได้เขียนไปแล้ว มาปรับปรุงโดยใช้ list comprehension

ตัวอย่างการเขียนโปรแกรม

โปรแกรมเดิม	โปรแกรมปรับปรุงแบบใช้ list comprehension
<pre> n,m = [int(e) for e in \ input().split()] d0 = list() for k in range(n) : d0.append(input().split()) err = list() for x in d0 : if len(x[1:]) != m : err.append(x[0]) if len(err) > 0 : print('Invalid data: ') for sid in err : print(sid) else : d1 = list() for x in d0 : s = 0 for e in x[1:] : s += float(e) d1.append([-s, x[0]]) d1.sort() for ntotal,scode in d1 : print(scode, -ntotal) </pre>	<pre> n,m = [int(e) for e in \ input().split()] d0 = [input().split() for k in range(n)] err = [x[0] for x in d0 if len(x[1:]) != m] if len(err) > 0 : print('Invalid data: ') print('\n'.join(err)) else : # list comprehension ซ้อนใน # list comprehension อีกชั้น d1 = [[-sum([float(e) for e in x[1:])), \ x[0]] for x in d0] d1.sort() for ntotal,scode in d1 : print(scode, -ntotal) </pre>



Minus All

โจทย์ข้อนี้ให้หาค่าเขียนโปรแกรมรับชุดตัวเลขจำนวนเต็มชุดหนึ่ง ซึ่งจะจบด้วยเลขที่มีค่าติดลบ จากนั้นให้แสดงผลชุดตัวเลขใหม่ (ไม่รวมตัวเลขที่ติดลบ) ที่สมาชิกแต่ละตัวมีค่าเท่ากับค่าเดิมรวมกับค่าที่ติดลบนั่น

► ข้อมูลนำเข้า

เป็นชุดของตัวเลข แต่ละบรรทัดจะประกอบด้วยตัวเลขจำนวนเต็มที่ไม่ติดลบ 1 จำนวน และบรรทัดสุดท้ายจะจบด้วยตัวเลขจำนวนเต็มที่มีค่าติดลบ (รับประกันว่า จะมีตัวเลขจำนวนเต็มที่ไม่ติดลบอย่างน้อย 1 จำนวน)

► ข้อมูลส่งออก

ให้แสดงชุดตัวเลขใหม่ (ไม่รวมตัวเลขที่ติดลบ) ที่สมาชิกแต่ละตัวมีค่าเท่ากับค่าเดิมรวมกับค่าที่ติดลบ โดยแสดงผลบรรทัดละ 1 จำนวน

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
1 10 2 -5	-4 5 -3
100 0 -1	99 -1
3 -3	0

คำนวณหา rank

จงเขียนโปรแกรมเพื่อคำนวณหา rank ของรหัสனிสิตที่กำหนด เมื่อเรียงลำดับตามคะแนน (rank 1 คือมีคะแนนมากที่สุด)

► ข้อมูลนำเข้า

แต่ละบรรทัดจะระบุข้อมูลนิสิตแต่ละคน ประกอบด้วยรหัสนิสิต ตามด้วยคะแนนเป็นเลขทศนิยม

บรรทัดสุดท้าย ระบุรหัสนิสิตที่ต้องการคำนวณหา rank

ในการคำนวณ rank หากมีนิสิตที่ได้คะแนนเท่ากัน ให้เรียงลำดับตามรหัสนิสิต (เรียงแบบจำนวนเต็ม)

► ข้อมูลส่งออก

มีบรรทัดเดียว แสดง rank ของนิสิตที่ต้องการค้นหา หากไม่พบรหัสนิสิตดังกล่าว ให้แสดงว่า Not Found

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
5931111121 87.25 5932222221 77.00 5933333321 82.50 5934444421 69.75 5935555521 66.00 5934444421	4
111 100 13 96 1234 96 555 99 2121 96 99 99 1234	5
429801 78 359124 89 902316 91.25 773842 45.75 264336	Not Found

ความถี่เกินครึ่ง

ให้เขียนโปรแกรมเพื่อหาข้อมูลที่มีความถี่มากกว่าครึ่งหนึ่งของจำนวนข้อมูลทั้งหมดที่รับเข้ามา โดยอ่านค่าจำนวนเต็มจนกระทั่งพบ -1

► ข้อมูลนำเข้า

ให้อ่านข้อมูลจำนวนเต็มบรรทัดละ 1 จำนวน จนกระทั่งพบค่า -1

► ข้อมูลส่งออก

มีบรรทัดเดียว แสดงข้อมูลที่มีความถี่มากกว่าครึ่งหนึ่งของข้อมูลทั้งหมด (ไม่รวม -1) ถ้าไม่มีข้อมูลที่มีความถี่มากกว่าครึ่งหนึ่งเลย ให้แสดง Not found

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
4 5 5 10 -1	Not found
4 5 5 -1	5
10 5 5 3 5 -1	5
2 4 2 4 2 4 -1	Not found

07 : Tuple, Dictionary and Set

สรุปเนื้อหา

Tuple

สร้าง tuple เหมือนสร้าง list แต่ใช้วงเล็บโค้ง	<pre>my_list = [1, 2.5, 3, 'A'] ได้ list my_tuple = (1, 2.5, 3, 'A') ได้ tuple my_tuple_2 = (1,) ได้ tuple ที่มีตัวเดียว (สังเกตที่ comma) not_a_tuple = (1) ได้จำนวนเต็ม เหมือน not_a_tuple = 1 t1 = tuple('abc') สร้างจากสตริง ได้ ('a','b','c') t2 = tuple([1, 2, 3]) สร้างจากลิสต์ ได้ (1,2,3)</pre>
การเข้าถึงข้อมูลทำเหมือน list	<pre>my_tuple[0] ได้ 1 my_tuple[1:3] ได้ (2.5, 3) my_tuple[-2:-1] ได้ (3,)</pre>
เครื่องหมาย + ใช้ต่อ tuple เครื่องหมาย * ใช้ต่อ tuple หลายครั้ง	<pre>a = (2, 3, 5) b = (7,) p = a + b ได้ (2, 3, 5, 7) q = p*2 ได้ (2, 3, 5, 7, 2, 3, 5, 7) a += (2,) เหมือนกับเขียน a = a + (2,) คือนำ a เดิมมาต่อกับ (2,) สร้าง tuple ใหม่ ได้ (2, 3, 5, 2) แล้วให้ค่า tuple นี้กับตัวแปร a</pre>
แก้ไขข้อมูลใน tuple ไม่ได้ (เหมือนสตริง)	<pre>my_tuple[3] = 'B' ผิด ถ้าต้องการเปลี่ยน ต้องตัดต่อสร้างใหม่ my_tuple = my_tuple[:3] + ('B',) + my_tuple[4:]</pre>
ระบบจะแปลงสิ่งที่มี comma คั่น ให้กลายเป็น tuple เช่น t = 1,2,3 เมื่อ print(t) ได้ (1, 2, 3)	<pre>a,b,c = (1,2,3) ได้ a = 1, b = 2, c = 3 x,y = 'A','Z' ได้ x = 'A', y = 'Z' a,x = x,a เป็นการสลับค่าในตัวแปร a กับ x</pre>
สามารถสร้าง list ของ tuple ได้	<pre>L = [('A','B'), (1,20,300), (9.9,)] เป็น list ของ tuple L[1] ได้ (1,20,300) L[1][2] ได้ 300</pre>

สามารถสร้าง tuple ของ list ได้	<pre>T = (['A', 'B'], [1,20,300], [9.9]) เป็น tuple ของ list T[1] ได้ [1,20,300] T[1][2] ได้ 300 T เป็น tuple จึงเปลี่ยน T[1] = 9 ไม่ได้ แต่เนื่องจาก T[1] เป็นลิสต์ จึงเปลี่ยน T[1][2] = 9 ได้</pre>
สามารถสร้าง tuple ของ tuple ก็ได้	<pre>T = ((1,2), (3,4)) เป็น tuple ของ tuple T[1][0] ได้ 3</pre>
<p>ข้อแตกต่างระหว่าง</p> <p><code>tuple1 += (0,)</code> กับ <code>list1 += [0]</code></p> <p><code>tuple1 += (0,)</code> เหมือน</p> <p><code>tuple1 = tuple1 + (0,)</code></p> <p>จะสร้าง tuple ใหม่ (ทำงานช้า)</p> <p>ในขณะที่ <code>list1 += [0]</code></p> <p>ไม่เหมือนกับ <code>list1 = list1 + [0]</code></p> <p>แต่จะเหมือนกับ <code>list1.append(0)</code></p> <p>เป็นการเพิ่ม 0 ต่อท้ายลิสต์ ไม่สร้างลิสต์ใหม่</p> <p>จะทำงานได้เร็วกว่า</p>	<p>ลองสั่งทำงานโปรแกรมทางขวานี้ แล้วสังเกตเวลาการทำงาน</p> <pre>import time n = 100000 t0 = time.time() list1 = [] for k in range(n): list1 += [0] print(time.time()-t0) t0 = time.time() tuple1 = tuple() for k in range(n): tuple1 += (0,) print(time.time()-t0)</pre>

Dictionary

- ใช้เก็บข้อมูลแบบคู่อันดับ (key, value)
- ไม่มี key ที่ซ้ำกัน หนึ่ง key มี value ที่คู่กันเพียงค่าเดียว (แต่ value อาจเป็น list, tuple, ... ที่เก็บข้อมูลย่อย ๆ ได้)
- ถ้าเรามี key จะสามารถหา value ที่คู่กับ key ได้เร็วมาก ๆ

การสร้าง dict ทำได้หลายแบบ อาจจะระบุข้อมูลที่ต้องการเก็บด้วยก็ได้	<pre>dict1 = {} หรือ dict1 = dict() เป็นการสร้าง dict ว่าง ๆ dict2 = {'Name': 'Tom', 'Age': 39} เป็นการสร้าง dict ที่เก็บคู่อันดับ ('Name', 'Tom') และ ('Age', 39)</pre>
สามารถเรียกใช้ แก้ไข และเพิ่ม value โดยการอ้างถึงด้วย key	<pre>D = {'Name': 'Sarah'} print(D['Name']) ได้ Sarah D['Age'] = 20 เป็นการเพิ่มข้อมูล ('Age', 20) D['Name'] = 'Somsri' เป็นการแก้ไขข้อมูลให้เป็น ('Name', 'Somsri')</pre>

key ของ dict อาจเป็นจำนวนเต็ม จำนวนจริง สตริง หรือ tuple ก็ได้ แต่ห้ามเป็น list, dict หรือ set ส่วน value ที่คู่กับ key จะเป็นประเภทใดก็ได้	<pre>my_dict = {} my_dict[1] = 8.00 my_dict[3.75] = [3, 7, 5] my_dict['Hello'] = 'World' my_dict[(2,99)] = 999 my_dict[[1,2,3]] = '123' ผิด ห้ามใช้ list เป็น key</pre>
มีบริการ keys(), values(), items() ซึ่งใช้กับ for...in ได้	<pre>d = {1:7, 2:8, 3:9} for e in d : แจกแจกแต่ละ key ของ d ให้กับ e for e in d.keys() แจกแจกแต่ละ key ของ d ให้กับ e for e in d.values() แจกแจกแต่ละ value ของ d ให้กับ e for e in d.items() แจกแจกแต่ละคู่ข้อมูลของ d เป็น tuple ให้กับ e e[0] คือ key, e[1] คือ value for k,v in d.items() แจกแจก key, value ของ d ให้กับ k, v</pre> <p>ลำดับของข้อมูลที่แจกแจกออกมาไม่จำเป็นต้องเหมือนกับที่เขียนหรือที่เพิ่ม</p>
บริการ keys(), values() และ items() ไม่ได้ผลเป็น list แต่เป็น อะไรบางอย่างที่คล้าย list นำไปใช้กับ for...in ได้ แต่ใช้ index เพื่อหยิบ ข้อมูลไม่ได้	<pre>d = {'aka':'as known as', 'so':'significant other'} x = d.keys() print(x[1]) ผิด ทำไม่ได้ เพราะ x ไม่ใช่ลิสต์ ถ้าต้องการขอ key ทั้งหมดให้เป็นลิสต์จริง ๆ ต้องเขียน x = list(d.keys()) ได้ x เป็นลิสต์ ใช้ x[1] ได้ print(x[1]) ทำได้ (แต่ไม่รู้ว่าจะได้ 'aka' หรือ 'so')</pre>
บริการ d.update(d1) เป็นการเพิ่ม ข้อมูลใหม่จาก dict d1 เข้าใน d ถ้ามี key ใน d1 ซ้ำกับใน d จะเป็นการแก้ไข value เดิม	<pre>A = {1:'one', 2:'two'} B = {1:'nueng', 3:'sarm'} A.update(B) ได้ A เป็น {1:'nueng', 2:'two', 3:'sarm'} ส่วน B ไม่เปลี่ยนแปลง</pre>
บริการ pop(key) เป็นการลบคู่ข้อมูล ที่มี key นั้นออก ถ้าไม่มี key อยู่ จะเกิด error	<pre>C = {1: 'one', 2: 'two', 3: 'three'} x = C.pop(1) ได้ x = 'one', C = {2:'two', 3:'three'} y = C.pop(4) เกิด error</pre>
การใช้ in เพื่อตรวจสอบว่ามี key หรือ value หรือ คู่ (key,value) ใน dict อยู่หรือไม่ โดยการตรวจ value จะไม่เร็วเท่ากับอีกสองกรณี	<pre>D = {1: 'one', 2: 'two', 3: 'three'} x = 1 in D ได้ x = True y = 4 in D.keys() ได้ y = False z = 'two' in D.values() ได้ z = True w = (2,'two') in D.items() ได้ w = True</pre> <p>การค้นใน keys และ items เร็วมาก แต่การค้นใน values ไม่เร็วเท่า</p>

Set

- เหมือนกับเซตทางคณิตศาสตร์ เซตไม่เก็บข้อมูลซ้ำ สามารถ union, intersect ได้
- เหมือนกับ dict แบบที่เก็บแค่ key ไม่มี value (เพราะ key ไม่ซ้ำกัน)
- การค้นด้วย in สามารถทำได้เร็วมาก ๆ

การสร้างเซต สามารถสร้างเซตจากข้อมูลใน string, tuple, list, set หรือ dict ได้	<pre>set_1 = set() ได้เซตว่าง set_2 = {1, 2, 3} ได้เซตที่มีสมาชิกเป็น 1, 2, 3 set_3 = set('Hello') ได้เซต {'H','e','l','o'} set_4 = set(['oh', 'no']) ได้เซต {'oh', 'no'} set_5 = set(set_2) ได้เซตใหม่มีสมาชิกเหมือนของ set_2 set_6 = set({1:2, 3:4}) ได้เซตของ key ของ dict คือ {1,3}</pre>
บริการ add(e) ใช้เพิ่มข้อมูล 1 ตัว ถ้าเพิ่มตัวที่ซ้ำกับที่มีอยู่ใน set ก็ไม่เพิ่มให้	<pre>S = {1} S.add((2, 3)) ได้ {1, (2, 3)} S.add('Hello') ได้ {1, (2, 3), 'Hello'}</pre>
บริการ update(t) ใช้เพิ่มข้อมูลที่แจกแจงได้จาก t ที่เป็นกลุ่มข้อมูล ซึ่งเป็นได้ทั้ง string, tuple, list, set และ dict (ในกรณีของ dict จะเพิ่ม key) คำสั่ง s.update(t) ทำงานเหมือนคำสั่ง for e in t : s.add(e)	<pre>S = {1} S.update([2, 3]) ได้ {1,2,3} S.update('Hello') ได้ {1,2,3,'H','e','l','o'} S.update(6) เกิด error</pre>
การใช้ e in S เพื่อตรวจสอบความเป็นสมาชิกของ e ในเซต S	<pre>S = {1, 2, 3, 'H', 'e', 'l', 'o'} x = 1 in S ได้ x = True y = 'h' in S ได้ y = False</pre>
บริการ remove(e) และ discard(e) เพื่อลบข้อมูล ถ้าไม่มีข้อมูลที่ต้องการจะลบ remove() จะเกิด error ดังนั้นควรใช้ discard()	<pre>S = {1, 2, 3, 'H', 'e', 'l', 'o'} S.discard(1) เหลือ {2, 3, 'H', 'e', 'l', 'o'} S.discard(4) เหลือเท่าเดิม ไม่เกิด error S.remove('e') เหลือ {2, 3, 'H', 'l', 'o'} S.remove('h') เกิด error เพราะไม่เจอ 'h'</pre>

<p>การดำเนินการของ set</p> <p>union() หรือใช้สัญลักษณ์ </p> <p>intersection() หรือใช้สัญลักษณ์ &</p> <p>difference() หรือใช้สัญลักษณ์ -</p> <p>$(A \cup B) - (A \cap B)$ หรือใช้สัญลักษณ์ ^</p> <p>โดยเซตที่มากระทำกัน จะไม่เปลี่ยนแปลง</p>	<pre> a = {1, 2, 3} b = {2, 3, 4} c = a.union(b) หรือ c = a b ได้ c = {1, 2, 3, 4} c = a.intersection(b) หรือ c = a & b ได้ c = {2, 3} c = a.difference(b) หรือ c = a - b ได้ c = {1} c = b.difference(a) หรือ c = b - a ได้ c = {4} c = a ^ b ได้ c = {1, 4} </pre>
<p>บริการ issubset() และ</p> <p>issuperset() ใช้ตรวจสอบความเป็น</p> <p>subset และ superset</p>	<pre> a = {1, 2, 3} b = {1, 2} a.issubset(b) ได้ False b.issubset(a) ได้ True a.issuperset(b) ได้ True b.issuperset(a) ได้ False </pre>

สรุปการใช้งาน list, tuple, dict, set

- ใช้คำสั่ง len, sum, max, min, in ได้ทั้งหมด
- ใช้ `x = sorted(q)` ได้ โดยที่ q เป็นได้ทั้ง list, tuple, dict, set
ผลที่ได้เป็น list ที่นำข้อมูลที่แจกแจงได้จาก q ไปเรียงลำดับ

	list	tuple	dict	set
การใช้งาน	ลำดับมีความหมาย สร้างแล้วแก้ไขได้	ลำดับมีความหมาย สร้างแล้วแก้ไขไม่ได้	เก็บคู่ลำดับ (key, value) key ไม่ซ้ำ, ไม่สนลำดับ	เก็บข้อมูลไม่ซ้ำ ไม่สนลำดับ สามารถใช้ set operation ได้
ประเภท ข้อมูลที่ เก็บ	อะไรก็ได้	อะไรก็ได้	key เป็น int, float, str, tuple, bool ส่วน value เป็นอะไรก็ได้	int, float, str, tuple, bool
การ เข้าถึง ข้อมูล	ใช้จำนวนเต็มระบุ ตำแหน่งหรือช่วง <code>x[i]</code> <code>x[a:b:c]</code>	ใช้จำนวนเต็มระบุ ตำแหน่งหรือช่วง <code>t[i]</code> <code>t[a:b:c]</code>	ใช้ key ระบุตำแหน่ง <code>d[key]</code> ได้ value ที่คู่กัน ไม่มีแบบรับ value แล้วได้ key	ไม่มี
การ แจกแจง ข้อมูล	<code>for e in x</code> ได้ลำดับจากซ้ายไปขวา	<code>for e in t</code> ได้ลำดับจากซ้ายไปขวา	<code>for k in d</code> <code>for k in d.keys()</code> <code>for v in d.values()</code> <code>for k,v in d.items()</code> ได้ลำดับไม่แน่นอน	<code>for e in s</code> ได้ลำดับไม่แน่นอน
การค้น ด้วย in, not in	ค้นจากซ้ายไปขวา (ช้า) ใช้ <code>x.index(e)</code> หา index ของ e ใน x	ค้นจากซ้ายไปขวา (ช้า) ใช้ <code>x.index(e)</code> หา index ของ e ใน x	ค้น key เร็วมาก	ค้นข้อมูลเร็วมาก

	list	tuple	dict	set
การสร้าง	<pre>x = [1,2,3,4] x = list() x = [] x = list(q) เมื่อ q เป็นสิ่งที่ใช้กับ for in ได้</pre>	<pre>t = (1, 2, 3, 4) t = (5,) t = tuple() t = () t = tuple(q) เมื่อ q เป็นสิ่งที่ใช้กับ for in ได้</pre>	<pre>d = {'k1':1,'k2':2} d = dict() d = {}</pre>	<pre>s = {1, 2, 3, 4} s = set() s = set(q) เมื่อ q เป็นสิ่งที่ใช้กับ for in ได้</pre>
	<pre>ใช้ x = list(x1) เมื่อ x1 เป็น list ไม่ควรเขียน x = x1 จะเป็น list ตัวเดียวกัน</pre>	<pre>ใช้ t = t1 ได้ เมื่อ t1 เป็น tuple เพราะ tuple ไม่เปลี่ยนค่า</pre>	<pre>ใช้ d = dict(d1) เมื่อ d1 เป็น dict ไม่ควรเขียน d = d1 จะเป็น dict ตัวเดียวกัน</pre>	<pre>ใช้ s = set(s1) เมื่อ s1 เป็น set ไม่ควรเขียน s = s1 จะเป็น set ตัวเดียวกัน</pre>
การเพิ่มข้อมูล	<pre>x.append(9) x.insert(1,9)</pre>	<pre>ต้องสร้างตัวใหม่ t=t + (9,) t=t[:1]+(9,)+t[1:]</pre>	<pre>d['k3'] = 9 d.update({'k3':9})</pre>	<pre>s.add(9) s.update({9})</pre>
การลบข้อมูล	<pre>x.pop(2)</pre>	<pre>ต้องสร้างตัวใหม่ t = t[:2] + t[3:]</pre>	<pre>d.pop('k3')</pre>	<pre>s.discard(99)</pre>
การแก้ไขข้อมูล	<pre>x[2] = 7</pre>	<pre>ต้องสร้างตัวใหม่ t=t[:2]+(7,)+t[3:]</pre>	<pre>d['k3'] = 7</pre>	<pre>ต้องลบแล้วเพิ่ม</pre>

การใช้ tuple, dict, set ที่พบบ่อย

ใช้ dict เพื่อจับคู่ข้อมูล หรือสร้างความสัมพันธ์ระหว่างคู่ข้อมูล key กับ value โดยหวังจะขอ value จากค่า key	<pre>month = {'JAN':1, 'FEB':2, 'MAR':3} ใช้ month[k] เพื่อขอเลขเดือนจากชื่อย่อเดือนที่เก็บในตัวแปร k num2en = {11:'eleven', 2:'two', 3:'three'} ใช้ num2en[a] เพื่อขอคำภาษาอังกฤษจากจำนวนเต็มในตัวแปร a</pre>
ใช้ set เพื่อเก็บกลุ่มข้อมูลที่ไม่ซ้ำ ต้องการค้นข้อมูลว่ามีอยู่หรือไม่ในเซตอย่างรวดเร็ว หรือต้องการบริการ intersection, union, issubset และอื่น ๆ ของ set	<pre>ต้องการหาจำนวนประกอบที่มีค่า ระหว่าง 2 ถึง 12 n = 13; c = set() for i in range(2, n//2) : for j in range(2*i, n, i) : c.add(j) # มีการเพิ่ม j ที่ซ้ำกัน แต่ add ไม่เพิ่มข้อมูลซ้ำ # c = {4, 6, 8, 9, 10, 12}</pre>

<p>ใช้ tuple เมื่อต้องการเก็บข้อมูลมีลำดับ คล้าย list แต่มันเ็นค่าหลังสร้างแล้วจะไม่เปลี่ยนค่าภายใน tuple ประหยัดหน่วยความจำ</p> <p>ใช้คำสั่ง <code>t1 = t2</code> ได้โดยไม่ต้องห่วงเรื่องการ ใช้ tuple ร่วมกัน (เพราะค่าภายในเปลี่ยนไม่ได้) และสามารถใช้เป็น key ของ dict และเป็นข้อมูลทีเก็บใน set ได้</p>	<p>มีพิกัด x, y, z จำนวนหนึ่งที่ต้องเก็บ ถ้าต้องการหา z จาก x,y ที่กำหนดให้บ่อย ๆ ก็ไม่ควรเก็บเป็น list of tuples (x,y,z) เช่น</p> <pre>d = [(1,1,3), (2,1,8), (3,1,2)]</pre> <p>แบบนี้การหา z จาก x,y ก็ต้องเป็น</p> <pre>for a,b,z in d : if x == a and y == b : print('z =', z) break else: print('Not Found')</pre> <p>ควรใช้ dict {(x,y):z} เช่น <code>d={(1,1):3,(2,1):8,(3,1):2}</code></p> <p>แบบนี้ การหา z จาก x,y ก็ง่ายและที่สำคัญคือเร็ว</p> <pre>if (x,y) in d : print('z =', d[(x,y)]) else : print('Not Found')</pre>
<p>การแจกแจงข้อมูลต่าง ๆ ใน dict</p>	<pre>d = {1:7, 2:8, 3:9} for k in d: print(k) for k in d.keys(): print(k) for v in d.values(): print(v) for k,v in d.items(): print(k,v)</pre>
<p>การแจกแจงข้อมูลใน dict เรียงตาม key หรือเรียงตาม value จากน้อยไปมาก</p>	<pre>d = {2:8, 1:8, 3:9} for k in sorted(d.keys()): print(k,d[k]) for v in sorted(d.values()): print(v)</pre>
<p>การรับคู่อันดับข้อมูล แล้วเพิ่มลงใน dict</p>	<pre>d = {} n = int(input()) for i in range(n): k,v = input().split() d[k] = v</pre>
<p>การรับคู่อันดับข้อมูล แล้วเพิ่มลงใน dict ที่มี value เป็น list หรือ set</p>	<pre>d = {} n = int(input()) for i in range(n): k,v = input().split() if k not in d: d[k] = [v] # ถ้า d[k] เป็นเซต, ใช้ d[k] = {v} else : d[k].append(v) # ถ้า d[k] เป็นเซต, ใช้ d[k].add(v)</pre> <p>ข้อควรระวัง : คำสั่ง <code>d[k] = [v]</code> ถ้าเขียนเป็น <code>d[k] = list(v)</code> จะมีปัญหา ถ้า v เป็นสตริง (เพราะอะไร ?)</p>

สร้างลิสต์ของ tuple เก็บข้อมูลชั่วคราว เพื่อการประมวลผล (เช่น เรียงลำดับข้อมูล)	d เป็น dict ที่ key เป็นรหัสนิสิต ส่วน value เป็นลิสต์ของคะแนน ต้องการแสดงชื่อเรียงลำดับตามคะแนนรวมจากน้อยไปมาก x = [(sum(scores),sid) for sid,scores in d.items()] t = [sid for sum_scores,sid in sorted(x)] print('\n'.join(t))
dict มี key เป็น a, value เป็นเซตของ b ต้องการสร้างอีก dict ที่กลับลักษณะ การจัดเก็บคือมี key เป็น b, value เป็นเซต ของ a	c เป็น dict ที่ key เป็นรหัสนิสิต ส่วน value เป็นเซตของรหัสวิชา จึงใช้ c ตอบคำถามว่ารหัสนิสิต sid เรียนวิชาอะไร ได้อย่างรวดเร็ว ถ้า อยากรู้ด้วยว่า รหัสวิชา cid มีรหัสนิสิตใดเรียนบ้าง ก็ต้องสร้างอีก dict stu เป็น dict ที่ key เป็นรหัสวิชา ส่วน value เป็นเซตของรหัสนิสิต stu = dict() for (sid,cids) in c.items() : for cid in cids : if cid not in stu : stu[cid] = {sid} else : stu[cid].add(sid)
นำชุดข้อมูลที่อาจมีค่าซ้ำกันมาเพิ่มใส่ set แล้วได้ชุดข้อมูลที่ไม่มีค่าซ้ำ	จาก dict c ในข้อที่แล้ว อยากทราบว่า มีรหัสวิชาอะไรบ้างที่มีนิสิตเรียน d = set() for cids in c.values() : d.update(cids)

เรื่องพิศบอย

คำสั่ง a = b โดยที่ b คือ set, dict หรือ list จะทำให้ a กับ b เป็นตัวแปร ของที่เก็บข้อมูลเดียวกัน ถ้าต้องการเป็น คนละตัว แต่เก็บข้อมูลเหมือนกัน ต้องใช้ a = set(b) หรือ a = dict(b) หรือ a = list(b)	A = {1:'one', 2:'two', 3:'three', 10:'ten'} B = A B[4] = 'four' จะเป็นการแก้ทั้ง A และ B เพราะเป็น dict เดียวกัน ถ้าใช้คำสั่ง C = dict(A) จะได้ A และ C เป็นคนละ dict กัน แต่มีข้อมูลเหมือนกัน
ไม่สามารถใช้ d.sort() เมื่อ d เป็น tuple, dict หรือ set ได้ แต่สามารถ ใช้ sorted(d) ได้ โดย d คือ tuple, dict, set หรือ list	S = {3, 1, 2}; D = {'A':5, 'C':2, 'B':7} S.sort() <i>ผิด เพราะ set ไม่มีบริการ sort</i> L = sorted(S) <i>ได้ L = [1, 2, 3]</i> L = sorted(D) <i>ได้ L = ['A', 'B', 'C']</i> L = sorted(D.values()) <i>ได้ L = [2, 5, 7]</i>

tuple ที่มีตัวเดียวต้องมี comma ต่อท้าย	<pre>my_tuple = (1,) ได้ tuple ที่มีตัวเดียว (สังเกตที่ comma) my_tuple += (4,) ได้ (1, 4) my_tuple += 5 ผิด นำ 5 ไปรวมกับ tuple ไม่ได้ my_tuple += (5) ผิด เขียน (5) ก็เหมือน 5 not_a_tuple = (1) ได้จำนวนเต็มธรรมดา</pre>	
tuple แก่ข้อมูลไม่ได้ tuple ไม่มีบริการ append, insert, add, pop, remove, discard	<pre>my_tuple[3] = 'B' ผิด ถ้าจะแก้ไขข้อมูล ต้องสร้างใหม่ my_tuple = my_tuple[:3] + ('B',) + my_tuple[4:]</pre>	
การอ้างถึงข้อมูลใน dict ที่ไม่มีมาก่อนจะผิด	<pre>D = {'Name':'Tom', 'Age':39} print(D['Gender']) ผิดเพราะอ้างได้แค่ 'Name' และ 'Age' หรือ ต้องการนับจำนวนตัวอักษรภาษาอังกฤษแต่ละตัวว่าปรากฏกี่ครั้งในสตริง t c = dict() for e in t : c[e] += 1 # ผิด เพราะอาจไม่มี key e ใน t ต้องเปลี่ยนเป็น for e in t : if e not in c : c[e] = 1 else : c[e] += 1</pre>	
key ของ dict ห้ามเป็น list, dict หรือ set	<pre>my_dict = {} my_dict[[1,2,3]] = 'list' ผิด เพราะ [1,2,3] ใช้เป็น key ไม่ได้</pre>	
การเก็บคู่อันดับใน dict อาจจะไม่เรียงลำดับตามลำดับการใส่ข้อมูล	<pre>D = {} D[1] = 1.00; D[2] = 2.00 for k in D.keys(): # อาจได้ 1 แล้ว 2 หรือ 2 แล้ว 1 ก็ได้ ไม่แน่นอน</pre>	
ใช้วงวนหา key เพื่อให้ได้ value ที่คู่กัน ทำแบบนี้ไม่ได้ใช้ความสามารถของ dict เลย	ต้องการหา value ของ key ที่มีค่าเท่ากับ key1 ใน d ไม่ควรเขียนแบบข้างล่างนี้	
	<pre>for k,v in d.items() : if k == key1 : print('value =', v) break else : print('Not found')</pre>	<pre>for k in d.keys() : if k == key1 : print('value =', d[k]) break else : print('Not found')</pre>
	เขียนแบบข้างล่างนี้ง่ายกว่า และเร็วกว่ามาก <pre>if key1 in d : print('value =', d[key1]) else : print('Not found')</pre>	

Problem	Code
<p><u>Input</u>: รับจำนวนเต็มบวก 1 จำนวนจากแป้นพิมพ์เก็บใน x</p> <p><u>Process</u>: สร้าง tuple ของจำนวนเต็มคู่ตั้งแต่ 0 และน้อยกว่า x</p> <p><u>Output</u>: tuple ที่สร้าง</p> <p>เช่น รับค่า 10 ให้แสดงผล (0, 2, 4, 6, 8)</p>	
<p><u>Input</u>: รับจำนวนเต็มบวก 1 จำนวนจากแป้นพิมพ์ เก็บใน x</p> <p><u>Process</u>: สร้าง tuple ของการแยกหลักของ x</p> <p><u>Output</u>: tuple ที่สร้าง</p> <p>เช่น รับค่า 12803 ให้แสดงผล (1, 2, 8, 0, 3)</p>	
<p><u>Input</u>: รับสตริงจากแป้นพิมพ์ เก็บใน x</p> <p><u>Process</u>: สร้าง dict แสดงการนับตัวอักษรของ x</p> <p><u>Output</u>: dict ที่สร้าง (ไม่สนใจลำดับที่แสดงผล)</p> <p>เช่น รับค่า book ให้แสดงผล {'b':1, 'k':1, 'o':2}</p>	
<p><u>Input</u>: รับสตริง 2 บรรทัดจากแป้นพิมพ์ เก็บใน x และ y</p> <p><u>Process</u>: สร้าง set ของตัวอักษรใน x และ set ของตัวอักษรใน y จากนั้นนำมาหาตัวอักษรที่ปรากฏในทั้งสองสตริง</p> <p><u>Output</u>: set ของ ตัวอักษรที่ปรากฏในทั้งสองสตริง (ไม่สนใจลำดับที่แสดงผล) เช่น รับค่า book และ bank ให้แสดงผล {'b', 'k'}</p>	

ตัวอย่างการแก้โจทย์ปัญหา

ทะเบียนนิสิต

จงเขียนโปรแกรมรับรายการของข้อมูล ซึ่งประกอบด้วยชื่อของนิสิตและคณะที่นิสิตคนนั้นอยู่ จากนั้นจะกำหนดชื่อคณะมาให้จำนวนหนึ่ง ให้ตอบว่านิสิตในคณะเหล่านั้นมีชื่ออะไรบ้าง ถ้ามีชื่อซ้ำกัน ให้ตอบเพียงครั้งเดียว

► ข้อมูลนำเข้า

บรรทัดแรกมีจำนวนเต็มบวก n คือจำนวนรายการข้อมูลทั้งหมด

n บรรทัดต่อมา แต่ละบรรทัดประกอบด้วย ชื่อของนิสิตและคณะที่นิสิตอยู่ โดยคั่นด้วยช่องว่าง

บรรทัดสุดท้ายจะเป็นรายชื่อคณะที่ต้องการถาม ถ้ามีหลายชื่อจะคั่นด้วยช่องว่าง

► ข้อมูลส่งออก

แสดงชื่อนิสิตในคณะเหล่านั้น โดยเรียงตามตัวอักษร ให้คั่นแต่ละชื่อด้วยช่องว่าง ถ้ามีชื่อซ้ำกัน ให้ตอบเพียงครั้งเดียว

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
3 Tom Engineering Pam Arts Jim Engineering Engineering	Jim Tom
5 Tom Engineering Pam Arts Jim Engineering Tom Arts Jenny Science Engineering Arts	Jim Pam Tom

ตัวอย่างการเขียนโปรแกรม

เนื่องจากโจทย์จะกำหนดชื่อคณะมาให้ และให้เราหาชื่อนิสิตที่อยู่ในคณะเหล่านั้น ดังนั้นเราควรเก็บข้อมูลโดยใช้ dict ซึ่งมี key คือชื่อคณะ และ value เป็น set ของชื่อนิสิต (เพราะในแต่ละคณะ มีนิสิตได้หลายคน) ตัวอย่างการเก็บข้อมูล เช่น

```
{
    'Engineering' : {'Tom', 'Jim'},
    'Arts'         : {'Pam', 'Tom'},
    'Science'      : {'Jenny'}
}
```

ขอแบ่งการประมวลผลเป็นขั้นตอนที่ละขั้นดังนี้

1. อ่านข้อมูลนำเข้าเก็บเป็น dict
2. รับรายการของชื่อคนที่ต้องการถาม มาเก็บเป็น list
3. เนื่องจากเราต้องการพิมพ์ชื่อที่ไม่ซ้ำกัน ดังนั้นเราจะใช้ set มาช่วย ให้สร้าง set คำตอบเริ่มต้นเป็นเซตว่าง
4. วาดรูปที่ละคนที่ต้องการถาม ให้นำชื่อนิสิตของคณะนั้นไปรวมกับเซตคำตอบที่มีอยู่เดิม
5. พิมพ์คำตอบ โดยเรียงลำดับชื่อตามตัวอักษร

โปรแกรม	คำอธิบาย
ขั้นตอนที่ 1	
<pre>n = int(input()) fac2name = {} for i in range(n): name,fac = input().split() fac2name[fac] = {name}</pre>	อ่านค่า n ซึ่งเป็นจำนวนข้อมูลเข้ามา สร้าง dict ว่าง ๆ ชื่อ fac2name ซึ่งจะใช้เก็บว่า คนนี้มี นิสิตชื่ออะไรบ้าง จากนั้นวนลูป n รอบ เพื่อเก็บข้อมูลคู่ (key, value) คือ ชื่อคณะและ set ของชื่อนิสิต แต่ตัวอย่างนี้ผิด เพราะแต่ละ key ของ dict จะเก็บ value ได้แค่ค่าเดียว ถ้าใส่ค่าแบบนี้จะทำให้ค่าที่เก็บใหม่ไปทับค่าเดิม
<pre>n = int(input()) fac2name = {} for i in range(n): name,fac = input().split() fac2name[fac].add(name)</pre>	เปลี่ยนมาใช้ add เพิ่มใน value ของ dict แต่การเก็บค่า แบบนี้ยังผิดอยู่ เพราะไม่ได้ตั้งค่าเริ่มต้นของ dict ไว้ก่อน จึงทำให้ add ไม่ได้
<pre>n = int(input()) fac2name = {} for i in range(n): name,fac = input().split() if fac in fac2name: fac2name[fac].add(name) else: fac2name[fac] = {name}</pre>	แบบที่ถูกต้อง ต้องมีการตรวจสอบก่อนว่า dict ของเรามี key นี้เก็บไว้หรือยัง ถ้ายังไม่มี ให้สร้าง set ขึ้นมาก่อน แต่ ถ้ามีแล้ว จะสามารถให้ค่าสั่ง add เพิ่มได้
ขั้นตอนที่ 2	
<pre>ask_fac = input().split()</pre>	รับข้อมูลรายชื่อคนที่ต้องการถาม มาเก็บไว้ใน list ชื่อ ว่า ask_fac
ขั้นตอนที่ 3	
<pre>ans_set = {}</pre>	สร้างเซตคำตอบเริ่มต้นเป็นเซตว่าง แต่แบบนี้ผิด เพราะจะได้เป็น dict ว่างแทน
<pre>ans_set = set()</pre>	สร้างแบบนี้ถึงจะได้เซตว่างที่ถูกต้อง

โปรแกรม	คำอธิบาย
ขั้นตอนที่ 4	
<pre>for f in ask_fac: ans_set.union(fac2name[f])</pre>	นำชื่อสินค้าในคณะมาเพิ่มในเซต <code>ans_set</code> แบบ union แต่แบบนี้ผิด เพราะการใช้คำสั่ง <code>union</code> แบบนี้ไม่ได้ทำให้ค่าของ <code>ans_set</code> เปลี่ยนไปด้วย
<pre>for f in ask_fac: ans_set = ans_set.union(fac2name[f])</pre>	การใช้คำสั่ง <code>union</code> แบบนี้จะทำให้ค่าใน <code>ans_set</code> เปลี่ยน แต่แบบนี้ยังผิดอยู่ เพราะคณะที่ถามมา อาจจะไม่ได้อยู่ใน <code>dict</code> ของเราก็ได้ ต้องตรวจสอบก่อน
<pre>for f in ask_fac: if f in fac2name: ans_set = ans_set.union(fac2name[f])</pre>	แบบนี้ถูกต้องแล้ว
ขั้นตอนที่ 5	
<pre>print(' '.join(ans_set.sort()))</pre>	พิมพ์ค่าใน <code>ans_set</code> โดยเรียงลำดับตามตัวอักษร แต่แบบนี้ผิด เพราะ <code>sort()</code> ใช้กับ <code>set</code> ไม่ได้
<pre>print(' '.join(list(ans_set).sort()))</pre>	ใช้วิธีแปลงเป็น <code>list</code> แล้วค่อยใช้ <code>sort()</code> แบบที่แสดงทางซ้ายนี้ก็ได้ เพราะ <code>list(ans_set).sort()</code> เรียงลำดับได้ แต่ไม่ได้คืนค่าอะไรกลับไปให้ <code>join</code>
<pre>ans_list = list(ans_set) ans_list.sort() print(' '.join(ans_list))</pre>	ต้องแปลงเป็น <code>list</code> เก็บใส่ในตัวแปร แล้วค่อยเรียก <code>sort</code> กับตัวแปร แล้วส่งตัวแปรนั้นให้ <code>join</code> ไปใช้
<pre>print(' '.join(sorted(ans_set)))</pre>	หรือใช้ <code>sorted</code> แทน เพราะ <code>sorted</code> รับ <code>set</code> ได้ และให้ผลเป็นลิสต์ที่เรียงลำดับแล้วกลับคืนมาส่งให้ <code>join</code>



Union & Intersection

เขียนโปรแกรมเพื่อหา union และ intersection ของเซตที่กำหนด

► ข้อมูลนำเข้า

บรรทัดแรก ระบุจำนวนเต็ม n แทนจำนวนเซต
n บรรทัดถัดมา ระบุเซตของจำนวนเต็มบรรทัดละ 1 เซต โดยระบุจำนวนเต็มที่อยู่ในเซต คั่นด้วยช่องว่าง

► ข้อมูลส่งออก

บรรทัดแรก แสดงขนาดของเซตที่เกิดจากการ union ทุกเซต
บรรทัดที่ 2 แสดงขนาดของเซตที่เกิดจากการ intersect ทุกเซต

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
3 1 2 1 2 3 1 2 1 2 1 2 3 2 5 4 3	5 2
6 100 1000 101 123 200 201 -1 -2 -3	9 0
6 -1 0 1 -1 1 0 0 -1 1 0 1 -1 1 -1 0 1 0 -1	3 3

อักษรสองตัวหน้าที่พบมากในประเภทของคำภาษาอังกฤษ

จากข้อมูลคำศัพท์ภาษาอังกฤษแยกตามประเภทของคำ จงเขียนโปรแกรมเพื่อหาตัวอักษรสองตัวแรกยอดฮิตของคำศัพท์ที่เป็นข้อมูลนำเข้า จำนวนคำศัพท์ และรายการคำศัพท์ที่ขึ้นต้นด้วยอักษรสองตัวแรกยอดฮิตนี้

► ข้อมูลนำเข้า

บรรทัดแรก บอกจำนวนรายการข้อมูลที่ต้องอ่านเข้ามา

บรรทัดที่เหลือ เป็นรายการข้อมูล โดยข้อมูลแรกเป็นประเภทของคำศัพท์ ข้อมูลที่สองเป็นคำศัพท์ ค้นด้วยเครื่องหมายแท็บ

► ข้อมูลส่งออก

ให้พิมพ์อักษรสองตัวหน้ายอดฮิต จำนวนคำศัพท์ และรายการคำศัพท์ที่ขึ้นต้นด้วยตัวอักษรสองตัวหน้ายอดฮิตนี้ พร้อมประเภทของคำศัพท์ ค้นด้วยเว้นวรรค โดยเรียงลำดับคำตามลำดับเดียวกันกับข้อมูลนำเข้า ถ้ามีอักษรสองตัวที่ปรากฏบ่อยมากที่สุดเท่ากัน ให้เลือกอักษรสองตัวแรกที่มาก่อนเรียงตามพจนานุกรม

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
10 Adjective <u>g</u> ood Noun <u>g</u> oose Verb <u>w</u> rap Verb <u>w</u> rite Noun <u>w</u> rinkle Noun <u>w</u> reck Noun <u>w</u> rangler Noun <u>h</u> all Adjective <u>h</u> appy Noun <u>h</u> obby	wr 5 wrap Verb write Verb wrinkle Noun wreck Noun wrangler Noun
6 Adjective <u>g</u> ood Verb <u>w</u> rap Verb <u>w</u> rite Noun <u>h</u> all Noun <u>h</u> obby Noun <u>g</u> oose	go 2 good Adjective goose Noun

ใครเรียนอะไร

ให้อ่านข้อมูลจากแป้นพิมพ์ โดยอ่านข้อมูลเป็นบรรทัด แต่ละบรรทัดมีรหัสนักเรียนคั่นด้วยเว้นวรรค ตามด้วยรหัสวิชา (อาจมีมากกว่า 1 วิชา) เมื่อพบว่า รหัสนักเรียนเป็น -1 ให้หยุดอ่าน จากนั้นให้อ่านรหัสวิชาสองรหัสวิชา แล้วให้แสดงจำนวนนักเรียนที่เรียนทั้งสองวิชานั้น จำนวนนักเรียนที่เรียนวิชาเดียวเท่านั้น และจำนวนนักเรียนทั้งหมดซึ่งเรียนวิชาใดวิชาหนึ่งหรือทั้งสองวิชา

► ข้อมูลนำเข้า

รหัสนักเรียนคั่นด้วยเว้นวรรค ตามด้วยรหัสวิชา (อาจมีมากกว่า 1 วิชา คั่นด้วยเว้นวรรค)

รับประกันว่าจะไม่มีนักเรียนที่รหัสซ้ำกัน และนักเรียน 1 คนจะไม่มีรหัสวิชาซ้ำกัน

เมื่อใส่ข้อมูลนักเรียนครบทุกคนแล้ว บรรทัดต่อไปจะเป็นข้อมูล -1

บรรทัดสุดท้ายเป็นรหัสวิชาสองรหัสวิชา คั่นด้วยเว้นวรรค

► ข้อมูลส่งออก

มี 3 จำนวนคั่นด้วยเว้นวรรค เรียงลำดับดังนี้

จำนวนนักเรียนที่เรียนทั้งสองวิชานั้น

จำนวนนักเรียนที่เรียนวิชาใดวิชาหนึ่งเท่านั้น

จำนวนนักเรียนซึ่งเรียนวิชาใดวิชาหนึ่งหรือทั้งสองวิชา

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
001 c001 c002 c003 002 c002 c003 c004 003 c003 c005 -1 c002 c003	2 1 3
5931111121 2110101 2109101 593222221 2109101 -1 2110101 5500101	0 1 1

► คำแนะนำ (ถ้ารู้วิธีทำแล้ว ไม่ต้องอ่านก็ได้)

ควรเก็บข้อมูลด้วย dict ที่มี key คือรหัสวิชา และ value เป็นเซตของนักเรียน (ลองคิดว่าทำไม) เช่น ตัวอย่างที่ 1 จะได้ dict ดังนี้

```
{  
    'c001':{'001'},  
    'c002':{'001', '002'},  
    'c003':{'001', '002', '003'},  
    'c004':{'002'},  
    'c005':{'003'}  
}
```

08 : Function and Recursion

สรุปเนื้อหา

การใช้งานฟังก์ชัน

- อาจเรียกว่า subprogram หรือ subroutine
- เป็นการแยกส่วนคำสั่งที่ซ้ำ ๆ กัน หรือเข้าใจยาก ออกมาจากโปรแกรมหลัก
- ทำให้โปรแกรมอ่านง่าย เข้าใจง่าย
- ทำให้โปรแกรมหลักเรียกใช้ฟังก์ชันได้ โดยไม่ต้องเขียนคำสั่งหลายรอบ
- ฟังก์ชันควรมีหน้าที่การทำงานชัดเจน เช่น ฟังก์ชันหาค่าเฉลี่ยของจำนวนในลิสต์ ฟังก์ชันกลับสตริง เป็นต้น
- ต้องเขียนฟังก์ชันไว้ก่อนส่วนที่จะเรียกใช้

องค์ประกอบของฟังก์ชัน

- ชื่อฟังก์ชัน มีข้อกำหนดเหมือนการตั้งชื่อตัวแปร
- ค่าที่รับเข้ามา หรือ "พารามิเตอร์" (ไม่จำเป็นต้องมีก็ได้)
- การคืนค่าจากฟังก์ชันด้วยคำสั่ง `return` (ไม่จำเป็นต้องมีก็ได้)

```
def average3(x,y,z):  
    s = x+y+z  
    return s/3
```

การคืนการทำงานจากฟังก์ชันและการคืนค่าจากฟังก์ชัน

- สามารถใช้คำสั่ง `return` ได้หลายทีในฟังก์ชัน
- เมื่อทำคำสั่ง `return` แล้ว จะหยุดการทำงานของฟังก์ชันทันที และกลับไปทำงานต่อหลังจากจุดที่เรียกใช้ฟังก์ชัน
- ฟังก์ชันคืนค่าได้ 1 ค่าเท่านั้น ถ้าต้องการคืนหลายค่า ให้ใช้ tuple เช่น `return (answer1, answer2)`
- หากคำสั่งสุดท้ายของฟังก์ชันไม่ใช่ `return` ระบบจะเพิ่มคำสั่ง `return` (ไม่คืนค่าใด ๆ) ที่ท้ายฟังก์ชัน
- คำสั่ง `return` ที่ไม่ได้กำหนดให้คืนค่าใด ๆ ระบบจะคืนค่า `None` (`None` เป็นค่าพิเศษในระบบ ไม่ใช่สตริง `'None'`)

ตัวแปรในฟังก์ชัน (local variables)

- ตั้งชื่อซ้ำกับตัวแปรในฟังก์ชันอื่นได้ ถือว่าเป็นคนละตัวแปรกัน
- เรียกใช้ตัวแปรในฟังก์ชันอื่นไม่ได้
- หากส่งตัวแปรประเภท `int`, `float`, `string`, `boolean` เข้ามาในฟังก์ชัน จะถือว่าเป็นคนละตัวกัน
- หากส่งตัวแปรประเภท `list`, `dict`, `set` เข้ามาในฟังก์ชัน
ถ้ามีการแก้ค่าในฟังก์ชัน ตัวแปรเดิมของผู้เรียกฟังก์ชันจะเปลี่ยนค่าด้วย

ฟังก์ชันเวียนเกิด

- ฟังก์ชันเวียนเกิดมี 2 ส่วนคือ ส่วนการคำนวณแบบพื้นฐาน และส่วนที่มีการเรียกซ้ำ
- บางครั้งเขียนง่ายกว่า loop เหมาะกับการทำซ้ำที่ไม่รู้จำนวนรอบ
- ทำงานช้ากว่า loop และใช้หน่วยความจำมากกว่า

```
def factorial(n):
    if n < 2: return 1
    return n * factorial(n-1)
```

การแปลงความสัมพันธ์ทางคณิตศาสตร์เป็นฟังก์ชันเวียนเกิด

- เขียนกรณีพื้นฐานก่อน คือกรณีที่เราคำนวณที่ ไม่มีการเรียกซ้ำ
- แล้วค่อยเขียนกรณีที่ต้องเรียกซ้ำ

$$f_n = \begin{cases} 0, & n = 0 \\ 1, & 1 \leq n \leq 2 \\ f_{n-1} + f_{n-2}, & \text{otherwise} \end{cases}$$

```
def f(n):
    if n == 0: return 0
    if 1 <= n <= 2: return 1
    return f(n-1) + f(n-2)
```

เรื่องพิศบอย

เขียนฟังก์ชันไว้หลังส่วนที่เรียกใช้	<pre>print(median(3,1,2))</pre> <pre>def median(x,y,z): return (x+y+z)-min(x,y,z)-max(x,y,z)</pre>	ผิด เพราะหาฟังก์ชัน median ไม่เจอ ต้องย้ายฟังก์ชันขึ้นไปไว้ก่อนบรรทัดนี้
ตั้งชื่อฟังก์ชันซ้ำกันเอง หรือซ้ำกับชื่อตัวแปร	<pre>def average(x,y): return (x+y)/2</pre> <pre>def average(x,y,z): return (x+y+z)/3</pre> <pre>print(average(1,2,3))</pre> <pre>print(average(4,5))</pre> <pre>average = 0</pre> <pre>print(average(1,2,3))</pre>	<p>average เป็นฟังก์ชันที่รับ 2 ตัวแปร</p> <p>average กลายเป็นฟังก์ชันที่รับ 3 ตัวแปร</p> <p>ยังเรียกได้อยู่</p> <p>ผิด เพราะต้องใช้แบบ 3 พารามิเตอร์</p> <p>average กลายเป็นตัวแปรแล้ว</p> <p>ผิด เพราะ average เป็นตัวแปรแล้ว</p>

<p>ถ้าพารามิเตอร์ของฟังก์ชันเป็น list, set หรือ dict การเปลี่ยนแปลงข้อมูลที่เกิดขึ้นในพารามิเตอร์นี้ จะส่งผลให้ข้อมูลที่เก็บในตัวแปรของผู้เรียกที่ส่งมาให้พารามิเตอร์เปลี่ยนแปลงด้วย (เพราะเป็นที่เก็บตัวเดียวกัน)</p> <pre>def f(x) : ... # ลิสต์ x เปลี่ยนแปลง ลิสต์ y จะเปลี่ยน x.append(e) x[i] = e x[:] = [0,0,0] x.pop(0) ...</pre> <p>f(y) ←</p>	<pre>def sum_double(x): for i in range(len(x)): x[i] *= 2 # x เป็นที่เก็บเดียวกับตัวแปรที่ส่งให้ x return sum(x)</pre> <p>y = [1, 2, 3] print(sum_double(y)) ได้ 12 print(y) ได้ [2, 4, 6]</p>
<p>ถ้าตั้งค่าใหม่ให้กับพารามิเตอร์ของฟังก์ชัน จะไม่ส่งผลถึงตัวแปรของผู้เรียกที่ส่งมาให้พารามิเตอร์</p> <pre>def f(x) : ... # ตัวแปร x เปลี่ยนค่า ลิสต์ y ไม่เปลี่ยน x = [1,2,3] ...</pre> <p>f(y) ←</p>	<pre>def double(x) : x *= 2</pre> <pre>def sum_double(x): x = [2*e for e in x] # x ถูกเปลี่ยนเป็นลิสต์ตัวใหม่แล้ว return sum(x)</pre> <p>a = 8 double(a) print(a) ได้ 8 เหมือนเดิม ไม่ใช่ 16</p> <p>y = [1, 2, 3] print(sum_double(y)) ได้ 12 print(y) จะได้ [1, 2, 3]</p>
<p>ลืม return ผลลัพธ์</p>	<pre>def square(x): x = x**2</pre> <p>ลืมคืนค่า จะทำให้ได้ None</p> <pre>def clip(x): if x > 255 : return 255</pre> <p>กรณี x ไม่เกิน 255 จะได้ None</p>
<p>ลืม () เมื่อเรียกใช้ฟังก์ชัน</p>	<pre>def read_next_positive_int(): n = int(input()) while n <= 0 : n = int(input()) return n</pre> <p>a = read_next_positive_int ผิด ต้องมี () หลังชื่อฟังก์ชัน a = read_next_positive_int() ถูกต้อง</p>

ส่งค่าหรือตัวแปรที่เก็บค่าผิประเภทให้กับพารามิเตอร์ของฟังก์ชัน	<pre>def f(n): n ต้องเป็นจำนวนเต็ม เพราะใช้ใน range g = 1 for k in range(n) : g = 1 + 1/(1+g) return g a = int(input()) b = f(a/2) ส่ง float ไป จะทำงานผิดในฟังก์ชัน f c = f(10*a) ส่งจำนวนเต็ม ทำงานได้ถูกต้อง</pre>
ลึ้มกรณีพื้นฐานของ recursive	<pre>def factorial(n): return n * factorial(n-1) จะเรียกฟังก์ชันวนไปเรื่อย ๆ</pre>
เขียนตรวจสอบกรณีพื้นฐานไว้ทีหลัง	<pre>def factorial(n): return n * factorial(n-1) if n == 0: return 1 บรรทัดนี้ไม่เคยทำงานเลย</pre>
ถ้ามีการคำนวณค่า recursive ที่ซ้ำกัน ควรคำนวณทีเดียว แล้วเก็บไว้ในตัวแปร	<pre>def f(n): if n == 0: return 1 return f(n-1) + f(n-1)**2 มีการคำนวณซ้ำ โปรแกรมจะช้า def f(n): if n == 0: return 1 x = f(n-1) return x + x**2 คำนวณครั้งเดียว จะทำงานเร็วกว่า</pre>



Problem	Code
<p>ชื่อฟังก์ชัน: f1</p> <p>Parameter: รับข้อมูล a เป็นสตริง และ รับ b เป็นจำนวนเต็ม</p> <p>Process: พิมพ์ a ออกทางหน้าจอจำนวน b บรรทัด</p> <p>Return: ไม่ต้องคืนค่า</p>	
<p>ชื่อฟังก์ชัน: f2</p> <p>Parameter: รับข้อมูล a เป็นสตริง และ รับ b เป็นจำนวนเต็ม</p> <p>Process: สร้างลิสต์ที่เก็บสตริง a จำนวน b ตัว</p> <p>Return: ลิสต์ที่สร้าง</p>	

Problem	Code
<p>ชื่อฟังก์ชัน: g</p> <p>Parameter: รับจำนวนจริงจำนวน m b n และ c</p> <p>Return: คืนค่าจุดตัดของเส้นตรง $y = mx+b$ และ $y = nx+c$ เป็น tuple ของจุดตัด (x,y)</p> <p>หากเป็นเส้นตรงที่ขนานกัน ให้คืนค่าจำนวนเต็ม 1</p> <p>หากเป็นเส้นตรงเดียวกัน ให้คืนค่าจำนวนเต็ม 2</p>	
<p>ชื่อฟังก์ชัน: h</p> <p>Parameter: รับลิสต์ของจำนวนเต็ม x</p> <p>Return: คืนลิสต์ใหม่ ที่มีสมาชิกจาก x เฉพาะที่เป็นเลขคู่เท่านั้น โดยห้ามแก้ไขค่าในลิสต์ x</p>	
<p>เขียนฟังก์ชันเวียนเกิดเพื่อคำนวณค่าดังนี้</p> $a_n = \begin{cases} 1, & n = 0 \\ -2, & n = 1 \\ a_{n-2} \times n, & otherwise \end{cases}$	
<p>เขียนฟังก์ชันเวียนเกิดเพื่อคำนวณค่าดังนี้</p> $\begin{aligned} k(2n) &= k(n) + (k(n) \% 10) & \text{if } n > 0 \\ k(2n+1) &= k(n-1) * n & \text{if } n > 0 \\ k(0) &= 1 \\ k(1) &= 2 \end{aligned}$	
<p>เขียนฟังก์ชันเวียนเกิดเพื่อคำนวณค่าดังนี้</p> $s_{i,k} = \begin{cases} 0, & i \geq k \\ k + t_{i+1,k}, & i < k \end{cases}$ $t_{j,k} = \begin{cases} 0, & j \geq k \\ j + s_{j,k-1}, & j < k \end{cases}$	
<p>ชื่อฟังก์ชัน: is_palindrome</p> <p>Parameter: รับข้อมูล s เป็นสตริง</p> <p>Process: เขียนฟังก์ชันเวียนเกิดเพื่อตรวจสอบว่า s เป็น palindrome (อ่านจากหน้าไปหลังเหมือนหลังไปหน้า) หรือไม่</p> <p>Return: ถ้า s เป็น palindrome ให้คืนค่า True ถ้าไม่เป็น ให้คืนค่า False</p>	

ตัวอย่างการแก้โจทย์ปัญหา

Function Refactoring

กำหนดโปรแกรมคำนวณจำนวนวันตั้งแต่วันที่ A จนถึงวันที่ B มาให้ จงแยกเป็นฟังก์ชันต่อไปนี้ และเติมคำสั่งให้ครบสมบูรณ์

- เส้นประ ฟังก์ชัน `is_leap_year(y)` คืน True เมื่อปีคริสต์ศักราช `y` เป็นปีอธิกสุรทิน
ถ้าไม่เป็นให้คืน False
- เส้นไข่ปลา ฟังก์ชัน `day_of_year(d,m,y)` คืนว่าวันที่ `d m y` เป็นวันที่ลำดับที่เท่าใดของปีคริสต์ศักราช `y`
- เส้นประยาว ฟังก์ชัน `days_in_year(y)` คืนจำนวนวันในปีคริสต์ศักราช `y`

► ข้อมูลนำเข้า

มี 2 บรรทัด บรรทัดแรกเป็นวันที่ A ในรูปแบบ `d1 m1 y1` เป็นปีคริสต์ศักราช

บรรทัดที่ 2 เป็นวันที่ B ในรูปแบบ `d2 m2 y2` เป็นปีคริสต์ศักราช

รับประกันว่าวันที่ทั้งสองวันจะเป็นวันที่ถูกต้อง และวันที่ A จะมาก่อนวันที่ B เสมอ และอยู่คนละปีกัน

► ข้อมูลส่งออก

จำนวนวันระหว่างวันที่ A และ B โดยรวมวันที่ A และ B ด้วย

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
1 1 2018 1 1 2020	731
25 12 1999 9 3 2000	76

โปรแกรมเริ่มต้น

```
d1,m1,y1 = [int(e) for e in input().split()]
d2,m2,y2 = [int(e) for e in input().split()]
```

คำนวณว่าวันที่ A เป็นวันที่เท่าไรของปี

```
D = [31,28,31,30,31,30,31,31,30,31,30,31]
```

```
if y1%4==0 and (y1%100!=0 or y1%400==0): # ตรวจสอบปีอธิกสุรทิน
```

```
    D[1] += 1
```

```
A = 0
```

```
for i in range(m1-1):
```

```
    A += D[i]
```

```
A += d1
```

คำนวณว่าวันที่ B เป็นวันที่เท่าไรของปี

```
D = [31,28,31,30,31,30,31,31,30,31,30,31]
```

```
if y2%4==0 and (y2%100!=0 or y2%400==0): # ตรวจสอบปีอธิกสุรทิน
```

```
    D[1] += 1
```

```
B = 0
```

```
for i in range(m2-1):
```

```
    B += D[i]
```

```
B += d2
```

คำนวณจำนวนวันระหว่างปี A และปี B

```
C = 0
```

```
for i in range(y1+1,y2):
```

```
    # บวกด้วยจำนวนวันในปีนั้น ๆ
```

```
    if i%4==0 and (i%100!=0 or i%400==0): # ตรวจสอบปีอธิกสุรทิน
```

```
        C += 366
```

```
    else:
```

```
        C += 365
```

```
print( ?? )
```

```
# (จำนวนวันในปี A) - (ลำดับที่ของวัน A) + 1 + (จำนวนวันระหว่างปีทั้งสอง) + (ลำดับที่ของวัน B)
```

ตัวอย่างการเขียนโปรแกรม

โปรแกรม	คำอธิบาย
ฟังก์ชัน is_leap_year	
จากโปรแกรมในกล่องเส้นประ สามารถเขียนเป็นฟังก์ชันได้ดังนี้ <pre>def is_leap_year(y): if y%4==0 and (y%100!=0 or y%400==0): return True else: return False</pre>	ฟังก์ชัน is_leap_year จะตรวจสอบว่าเป็นปีอธิกสุรทินหรือไม่ ตามเงื่อนไขในกล่องเส้นประ ถ้าตรงตามเงื่อนไขให้คืนค่า True แต่ถ้าไม่ตรงให้คืนค่า False
หรือสามารถเขียนให้สั้นลงได้เป็นแบบนี้ <pre>def is_leap_year(y): if y%4==0 and (y%100!=0 or y%400==0): return True return False</pre>	ปรับปรุง : ในกรณีนี้ เราสามารถตัดคำว่า else ออกได้ เนื่องจากถ้าเงื่อนไขเป็นจริงแล้วจะ return True และจบการทำงานฟังก์ชันทันที จะไม่มาทำงานที่บรรทัด return False อีก
หรือสามารถเขียนให้สั้นลงได้อีกเป็นแบบนี้ <pre>def is_leap_year(y): return y%4==0 and (y%100!=0 or y%400==0)</pre>	ปรับปรุง : เนื่องจากประโยคตรวจสอบเงื่อนไขมีค่าความจริงเป็น True หรือ False อยู่แล้ว เราสามารถคืนผลการเปรียบเทียบนี้โดยตรงได้เลย
ฟังก์ชัน day_of_year	
จากโปรแกรมในกล่องเส้นไขว่ปลา สามารถเขียนเป็นฟังก์ชันได้ดังนี้ <pre>def day_of_year(d,m,y): D = [31,28,31,30,31,30,31,31,30,31,30,31] if y%4==0 and (y%100!=0 or y%400==0): D[1] += 1 A = 0 for i in range(m-1): A += D[i] A += d</pre>	คัดลอกโปรแกรมจากกล่องเส้นไขว่ปลา อย่าลืมเปลี่ยนชื่อตัวแปรด้วย (d1,m1,y1 เป็น d,m,y) แต่ฟังก์ชันนี้ยังไม่ถูกต้องเพราะลืมนับค่า
<pre>def day_of_year(d,m,y): D = [31,28,31,30,31,30,31,31,30,31,30,31] if y%4==0 and (y%100!=0 or y%400==0): D[1] += 1 A = 0 for i in range(m-1): A += D[i] A += d return A</pre>	เติมคำสั่ง return A ทำให้ฟังก์ชันทำงานได้ถูกต้องแล้ว

โปรแกรม	คำอธิบาย
<pre>def day_of_year(d,m,y): D = [31,28,31,30,31,30,31,31,30,31,30,31] if is_leap_year(y): D[1] += 1 A = 0 for i in range(m-1): A += D[i] A += d return A</pre>	ปรับปรุง: นำฟังก์ชัน is_leap_year ที่เขียนไว้แล้วมาใช้ได้
ฟังก์ชัน days_in_year(y)	
<p>จากโปรแกรมในกล่องเส้นประยาว สามารถเขียนเป็นฟังก์ชันได้ดังนี้</p> <pre>def days_in_year(y): if y%4==0 and (y%100!=0 or y%400==0): return 366 else: return 365</pre>	คัดลอกโปรแกรมจากกล่องเส้นประยาว อย่าลืมเปลี่ยนชื่อตัวแปร และคืนค่าให้ถูกต้อง
<pre>def days_in_year(y): if is_leap_year(y): return 366 else: return 365</pre>	ปรับปรุง: นำฟังก์ชัน is_leap_year ที่เขียนไว้แล้วมาใช้
<pre>def days_in_year(y): return day_of_year(31,12,y)</pre>	ปรับปรุง: นำฟังก์ชัน day_of_year ที่เขียนไว้แล้วมาใช้ โดยขอลำดับของวันที่ 31 ธันวาคม ปี y (กรณีนี้ได้โปรแกรมสั้นลง แต่อาจทำให้ฟังก์ชันเข้าใจยากขึ้น)
ส่วนโปรแกรมหลัก	
<pre>d1,m1,y1 = [int(e) for e in input().split()] d2,m2,y2 = [int(e) for e in input().split()] A = day_of_year(d1,m1,y1) B = day_of_year(d2,m2,y2) C = 0 for i in range(y1+1,y2): C += days_in_year(i) print(days_in_year(y1) - A + 1 + C + B)</pre>	เปลี่ยนกล่องต่าง ๆ เป็นการเรียกฟังก์ชันและส่งค่าไปให้แต่ละฟังก์ชันให้ถูกต้อง

ตัวอย่างโจทย์ปัญหา

Four Functions

จงเขียน 4 ฟังก์ชัน ให้ทำงานตามที่เขียนอธิบายกำกับแต่ละฟังก์ชัน ในโครงของโปรแกรมข้างล่างนี้

```
def make_int_list(x):
    # รับสตริง x มาแยกและแปลงเป็น int เก็บใน list แล้วคืนเป็นผลลัพธ์
    # เช่น x = '12 34 5' ได้ผลเป็น [12, 34, 5]

def is_odd(e):
    # คืนค่าจริงเมื่อ e เป็นจำนวนคี่ ถ้าไม่ใช่ คืนค่าเท็จ

def odd_list(alist):
    # คืน list ที่มีค่าเหมือน alist แต่มีเฉพาะตัวที่เป็นจำนวนคี่
    # เช่น alist = [10, 11, 13, 24, 25] จะได้ [11, 13, 25]

def sum_square(alist):
    # คืนผลรวมของกำลังสองของแต่ละค่าใน alist
    # เช่น alist = [1,3,4] ได้ผลเป็น (1*1 + 3*3 + 4*4) = 26

exec(input().strip()) # ต้องมีบรรทัดนี้เมื่อส่งไป Grader
```

► ข้อมูลนำเข้า

คำสั่งในการทดสอบฟังก์ชันที่เขียน

► ข้อมูลส่งออก

ผลที่ได้จากคำสั่งที่ป้อนเป็นข้อมูลนำเข้า

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
<code>print(make_int_list('1 2 3 4 5'))</code>	<code>[1, 2, 3, 4, 5]</code>
<code>print(is_odd(2222))</code>	<code>False</code>
<code>print(odd_list([1,2,3,4,5,6,7]))</code>	<code>[1, 3, 5, 7]</code>
<code>print(sum_square([1,1,2,3]))</code>	<code>15</code>

Recursive C(n,k)

เราสามารถหาค่าของจำนวนวิธีในการเรียงสับเปลี่ยน C(n,k) ได้จากสมการด้านล่าง จงใช้สมการต่อไปนี้ในการเขียนโปรแกรมแบบ recursive เพื่อคำนวณค่า C(n,k)

$$C(n, k) = \begin{cases} C(n-1, k) + C(n-1, k-1), & \text{if } 0 < k < n \\ 1, & \text{if } k = 0 \text{ or } n = k \\ 0, & \text{otherwise} \end{cases}$$

► ข้อมูลนำเข้า

มี 2 บรรทัด ประกอบด้วยจำนวนเต็ม n และ k

► ข้อมูลส่งออก

มี 1 บรรทัด แสดงค่า C(n,k) ที่คำนวณได้

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
6 2	15
10 8	45
0 1	0
3 7	0
10 10	1

Recursive SumList

โจทย์ข้อนี้สั้น ๆ จงเขียนฟังก์ชัน `sumlist(x)` ของโครงโปรแกรมข้างล่างนี้ `sumlist` รับ `x` เป็นลิสต์เก็บจำนวนเต็ม แล้วคืนผลรวมของจำนวนเต็มทุกตัวใน `x` โดย `x` เป็นลิสต์ที่ภายในเป็นลิสต์ซ้อนลิสต์กี่ชั้นก็ได้ ดังตัวอย่างที่แสดงข้างล่างนี้

หมายเหตุ : สามารถใช้คำสั่ง `if type(x) == list` เพื่อทดสอบว่า x เป็นข้อมูลประเภทลิสต์หรือไม่

```
def sumlist( x ):
    # ???

print(eval(input().strip())) # do not remove this line
```

► ข้อมูลนำเข้า

ลิสต์ที่เก็บจำนวนเต็ม (อาจเป็นลิสต์แบบลิสต์ซ่อนลิสต์ก็ขึ้นก็ได้)

► ข้อมูลส่งออก

ผลรวมของจำนวนเต็มทุกตัวในลิสต์ที่เป็นข้อมูลนำเข้า

▶ ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
sumList([],[])	0
sumList([1,1,2,2])	6
sumList([1,[1],[2,[2],0,[0,0]]])	6
sumList([0,[1,[2,[3,[4,5],6],7],8],[9,10]])	55

สรุปเนื้อหา

`import numpy as np`

- NumPy เป็นคลังคำสั่งที่ให้บริการมากมายเกี่ยวกับการคำนวณทางวิทยาศาสตร์ โดยมีที่เก็บข้อมูลที่เรียกว่า อาร์เรย์ n มิติ (ndarray) มีไว้เก็บข้อมูลเพื่อการประมวลผลที่มีประสิทธิภาพมาก ๆ
- อาร์เรย์มีลักษณะคล้ายลิสต์ แต่สร้างแล้วเปลี่ยนขนาดไม่ได้
- ค่าในอาร์เรย์ทุกช่องต้องเป็นประเภทเดียวกันทั้งหมด เช่น เป็น `int` ทุกช่อง หรือ `float` ทุกช่อง (ผสมกันไม่ได้ ถ้าเป็นลิสต์ผสมได้)
- เราสร้าง เวกเตอร์ ได้ด้วยอาร์เรย์ 1 มิติ และสร้าง เมทริกซ์ ได้ด้วยอาร์เรย์ 2 มิติ

การสร้างอาร์เรย์ด้วยฟังก์ชัน

สร้างอาร์เรย์จาก ลิสต์ L	<code>np.array(L)</code>	หาขนาดของ อาร์เรย์ M	<code>M.shape</code>
<code>[1 0 2 3 -1]</code>	<code>np.array([1,0,2,3,-1])</code>	<code>[[2 3] [4 1] [7 5]]</code>	<code>np.array([[2,3],[4,1],[7,5]])</code>
<code>[[0. 0. 0.] [0. 0. 0.]]</code>	<code>np.zeros((2,3))</code>	<code>[[1 1] [1 1] [1 1]]</code>	<code>np.ones((3,2),int)</code>
<code>[[1 0 0 0] [0 1 0 0] [0 0 1 0] [0 0 0 1]]</code>	<code>np.identity(4,int)</code>	<code>[[1. 0. 0.] [0. 1. 0.] [0. 0. 1.]]</code>	<code>np.identity(3)</code>
สร้างอาร์เรย์ที่มีขนาดเท่ากับอาร์เรย์ x และเป็น 0 (int) ทุกช่อง	<code>np.zeros_like(x,int)</code> (x อาจเก็บ int หรือ float ก็ได้)	สร้างอาร์เรย์ที่มีขนาดเท่ากับอาร์เรย์ y และเป็น 1.0 ทุกช่อง	<code>np.ones_like(y,float)</code> (y อาจเก็บ int หรือ float ก็ได้)
<code>[4 5 6 7 8 9]</code>	<code>np.arange(4,10)</code>	<code>[8. 7. 6. 5.]</code>	<code>np.arange(8.0,4.0,-1.0)</code> <code>np.arange(8,4,-1,float)</code>
<code>[2.3 2.33 2.36 2.39 2.42 2.45 2.48]</code>			<code>np.arange(2.3,2.5,0.03)</code>

Element-wise operations, Broadcasting และ Indexing

- เป็นสามแนวคิดของ NumPy (ที่ต้องรู้ในวิชานี้) ที่ช่วยให้การประมวลผลอาเรย์ทำได้สะดวก
- การดำเนินการอาเรย์กับอาเรย์ (เช่น $A+B$) การดำเนินการหรือทำฟังก์ชันกับอาเรย์ จะเป็นแบบช่องต่อช่อง (element-wise) เช่น
$$\text{np.array}([1,2,3]) + \text{np.array}([1,1,1]) \text{ ได้ } \text{np.array}([2,3,4])$$
$$1/(\text{np.array}([1,2,4])) \text{ ได้ } \text{np.array}([1.0, 0.5, 0.25])$$
- การดำเนินการอาเรย์กับอาเรย์ที่มีขนาดไม่เท่ากัน อาจมีการขยายอาเรย์ให้มีขนาดเท่ากันก่อน (broadcasting) โดยพิจารณามิติของทั้งสองอาเรย์จากขวาไปซ้าย ให้เป็นไปตามกฎการ broadcast (ขอนำเสนอด้วยตัวอย่าง)
- $A.\text{shape} = (2,3)$, $B.\text{shape} = (3,)$ ดูจากขวามาซ้าย 3 เท่ากัน จึง broadcast B ให้มีขนาดเท่ากับ A ได้
เช่น
$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} + \begin{bmatrix} 9 & 8 & 7 \end{bmatrix} \text{ broadcast แล้วกลายเป็น } \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} + \begin{bmatrix} 9 & 8 & 7 \\ 9 & 8 & 7 \\ 9 & 8 & 7 \end{bmatrix} = \begin{bmatrix} 10 & 10 & 10 \\ 13 & 13 & 13 \end{bmatrix}$$
- มิติไหนขนาดเป็น 1 ถือว่า มิตินั้น broadcast ได้ (แต่มิติอื่นที่ไม่ใช่ 1 ต้องเท่ากัน)
 - $A.\text{shape} = (2,3)$, $B.\text{shape} = (2,1)$ broadcast B ให้เป็น $(2,3)$ มีขนาดเท่ากับ A
 - $A.\text{shape} = (1,3)$, $B.\text{shape} = (2,1)$ broadcast ทั้ง A และ B ให้เป็น $(2,3)$
เช่น
$$\begin{bmatrix} 1 & 2 & 3 \end{bmatrix} + \begin{bmatrix} 6 \\ 7 \end{bmatrix} \text{ broadcast แล้วกลายเป็น } \begin{bmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{bmatrix} + \begin{bmatrix} 6 & 6 & 6 \\ 7 & 7 & 7 \end{bmatrix} = \begin{bmatrix} 7 & 8 & 9 \\ 8 & 9 & 10 \end{bmatrix}$$
 - $A.\text{shape} = (2,3)$, $B.\text{shape} = (1,)$ broadcast B ได้ (ถ้า B เป็นสเกลาร์ ให้ถือว่ามิติเป็น $(1,)$)
เช่น
$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} + 9 \text{ broadcast แล้วกลายเป็น } \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} + \begin{bmatrix} 9 & 9 & 9 \\ 9 & 9 & 9 \end{bmatrix} = \begin{bmatrix} 10 & 11 & 12 \\ 13 & 14 & 15 \end{bmatrix}$$
 - $A.\text{shape} = (3,5)$, $B.\text{shape} = (5,1)$ broadcast B ไม่ได้ (ปรับ 1 ได้ แต่อีกมิติไม่เท่ากัน เลยทำไม่ได้)
- การอ้างอิงข้อมูลในอาเรย์ (Indexing ของ NumPy)
 - การอ้างอิงข้อมูลในอาเรย์ 1 มิติ
 - การเข้าใช้ข้อมูล
 - $V[k]$ เหมือนการใช้ลิสต์ คือเลือกข้อมูลหนึ่งช่อง โดยที่ k เป็นจำนวนเต็ม
 - $V[a:b:c]$ เหมือนการใช้ลิสต์ คือเลือกข้อมูลเป็นช่วง ได้ผลเป็นอาเรย์
 - การใส่ค่าใหม่
 - $V[k] = 99$ เหมือนกับการใช้ลิสต์
 - $V[a:b:c] = d$ แตกต่างจากลิสต์
 - ถ้า d เป็น int, float หรือลิสต์/อาเรย์ขนาด 1 ช่อง จะ broadcast ให้มีขนาดเท่ากับช่วงของ $a:b:c$
 - ถ้า d เป็นลิสต์/อาเรย์ขนาดมากกว่า 1 ช่อง ต้องมีขนาดเท่ากับขนาดของช่วง $a:b:c$ เช่น
 - $V = \text{np.zeros}(5, \text{int})$
 - $V[2:4] = 1$ ทำได้, V เปลี่ยนเป็น $[0 \ 0 \ 1 \ 1 \ 0]$
 - $V[:,2] = 3$ ทำได้, V เปลี่ยนเป็น $[3 \ 0 \ 3 \ 1 \ 3]$
 - $V[:,2] = [9]$ ทำได้, V เปลี่ยนเป็น $[9 \ 0 \ 9 \ 1 \ 9]$
 - $V[:,2] = [8,8,8]$ ทำได้, V เปลี่ยนเป็น $[8 \ 0 \ 8 \ 1 \ 8]$
 - $V[1:5] = [1,2]$ ทำไม่ได้, เพราะขนาดของข้อมูลไม่ตรงกัน
 - $V[7:7] = [1,2]$ ไม่เกิดการเปลี่ยนแปลงใด ๆ

○ การอ้างอิงข้อมูลในอาร์เรย์ 2 มิติ

- สามารถอ้างอิงถึงตัวข้อมูล, อาร์เรย์ 1 มิติที่แทนแถว (row) ของข้อมูล, อาร์เรย์ 1 มิติที่แทนหลัก (column) ของข้อมูล หรืออาร์เรย์ย่อย 2 มิติที่แทนช่วงของแถวและหลัก ได้หลากหลายแบบ ดังนี้

- $M[r,c]$ ข้อมูล ณ แถวที่ r หลักที่ c
- $M[r,:]$ หรือ $M[r]$ อาร์เรย์ 1 มิติของแถวที่ r ทั้งหมด
- $M[:,c]$ อาร์เรย์ 1 มิติของหลักที่ c ทั้งหมด (ไม่ใช่อาร์เรย์ 2 มิติที่มี 1 หลัก)
- $M[r1:r2,:]$ หรือ $M[r1:r2]$ อาร์เรย์ 2 มิติประกอบด้วย แถวที่ $r1$ ถึง $r2-1$
- $M[:,c1:c2]$ อาร์เรย์ 2 มิติประกอบด้วย หลักที่ $c1$ ถึง $c2-1$
- $M[r1:r2,c1:c2]$ อาร์เรย์ย่อย 2 มิติ ในช่วงแถวที่ $r1$ ถึง $r2-1$ และช่วงหลักที่ $c1$ ถึง $c2-1$
- $M[r1:r2:a,c1:c2:b]$ อาร์เรย์ย่อย 2 มิติ ในช่วงแถวและหลักที่กำหนดโดย $r1:r2:a, c1:c2:b$
- เช่น ให้ $M = \text{np.array}([[1,2,3,4], [5,6,7,8], [9,10,11,12]])$
 - $M[1:3]$ ได้ $\text{np.array}([[5,6,7,8], [9,10,11,12]])$
 - $M[1:2, 1:3]$ ได้ $\text{np.array}([[6,7]])$
 - $M[:, 2]$ ได้ $\text{np.array}([3, 7, 11])$
 - $M[1:, :2]$ ได้ $\text{np.array}([[5,6], [9,10]])$
 - $M[0:3:2]$ ได้ $\text{np.array}([[1,2,3,4], [9,10,11,12]])$
 - $M[0:3:2, 1:4:2]$ ได้ $\text{np.array}([[2,4], [10,12]])$
 - $M[:, :-1]$ ได้ $\text{np.array}([[9,10,11,12], [5,6,7,8], [1,2,3,4]])$
- สามารถให้ค่ากับหลาย ๆ ช่องในอาร์เรย์พร้อมกันได้ เช่น
 - $M[2:4, 1:4] = [[-1, 1, -1], [-1, 1, -1]]$ ขนาดเท่ากันพอดี
 - $M[2:4, 1:4] = [-1, 1, -1]$ ขนาดไม่เท่า ระบบ broadcast ให้
 - $M[:, 0] = [1, 2, 3, 4]$ M ต้องมี 4 แถว
 - $M[:, :2, ::3] = 0$ broadcast 0

คำสั่ง dot (ใช้ว่า $x.\text{dot}(y)$ หรือ $\text{np.dot}(x,y)$ ก็ได้)

<u>dot ใช้กับเวกเตอร์ คือการหา dot product</u>	<u>dot ใช้กับเมทริกซ์ คล้ายกับการคูณเมทริกซ์</u>
$x = \text{np.array}([1, 2, 3, 4])$ $y = \text{np.array}([0, -1, 1, 2])$ $x.\text{dot}(y)$ ได้ค่าเท่ากับ $y.\text{dot}(x)$ เท่ากับ $1*0 + 2*(-1) + 3*1 + 4*2 = 9$	$x = \text{np.array}([[1,2], [3,4], [5,6]])$ $x.\text{shape}$ คือ (3,2) $y = \text{np.array}([[-2], [3]])$ $y.\text{shape}$ คือ (2,1) $z = \text{np.array}([[-2,3]])$ $z.\text{shape}$ คือ (1,2) $w = \text{np.array}([-2,3])$ $w.\text{shape}$ คือ (2,)
	$x.\text{dot}(y)$ ได้ $\text{np.array}([[4], [6], [8]])$ มีมิติเป็น (3,2) กับ (2,1) คูณได้ ได้อาร์เรย์ขนาด (3,1) $y.\text{dot}(x)$ ทำไม่ได้ เพราะมิติไม่ถูกต้อง (2,1) กับ (3,2) $x.\text{dot}(z)$ ทำไม่ได้ เพราะมิติไม่ถูกต้อง (3,2) กับ (1,2) $x.\text{dot}(w)$ ได้ $\text{np.array}([4, 6, 8])$ อันนี้แปลก (3,2) กับ (2,) คูณได้ ได้อาร์เรย์ขนาด (3,)

ฟังก์ชันที่ระบุแทนได้, การ transpose และการเปรียบเทียบ

(M = np.array([[1,2,3,4],[5,6,7,8]]))

- ฟังก์ชันที่ระบุแทนได้ เช่น sum, max, min, mean, std, argmax, argmin (คืนตำแหน่งตัวมาก/น้อยสุด)
- np.sum(M) ได้ผลรวมของทุกช่อง np.sum(M) ได้ 36
- np.sum(M,axis=0) ได้ผลรวมตามแนวดิ่ง np.sum(M,axis=0) ได้ np.array([6,8,10,12])
- np.sum(M,axis=1) ได้ผลรวมตามแนวนอน np.sum(M,axis=1) ได้ np.array([10,26])
- matrix transpose ใช้ว่า M.T M.T ได้ np.array([[1,5],[2,6],[3,7],[4,8]])
- M > 3 ได้ array([[False,False,False,True],[True,True,True,True]])
- ใช้ np.sum นับจำนวน element ที่ตรงตามเงื่อนไขได้ np.sum(M > 3) ได้ 5
- ใช้เงื่อนไขในการเลือกบาง element ของอาร์เรย์ได้ M[M%2==0] ได้ np.array([2,4,6,8])

เรื่องพิศบอย

ลืม import numpy	โดยทั่วไปมักใช้ import numpy as np คือการ import คลังคำสั่ง numpy และตั้งชื่อใหม่ว่า np จะได้เขียนสั้น ๆ
ลืมระบุประเภทของข้อมูลที่จะเก็บว่าเป็น int หรือ float	x = np.zeros((3,4)) # ได้เป็น float x = np.zeros((3,4),int) # ได้เป็น int x = np.zeros_like(y) # ขึ้นกับประเภทข้อมูลของ y
ถ้าเก็บค่า float ในเมทริกซ์ที่เก็บ int ระบบจะปัดค่าหลังจุดทศนิยมทิ้ง	x = np.array([1,2,3],int) x[0] = 4.8 # ได้ x เท่ากับ np.array([4,2,3])
ใช้ np.array กับ np.ndarray ผิด	ต้องการสร้างเมทริกซ์ขนาด 4x5 แต่เขียน a = np.array((4,5)) # ได้ [4,5] ต้องการสร้างเวกเตอร์ [2,3] แต่เขียน a = np.ndarray([2,3]) # ได้เมทริกซ์ขนาด 2x3
การทำงานของเวกเตอร์ไม่เหมือนเมทริกซ์ (อาร์เรย์ 1 มิติ vs 2 มิติ)	x = np.array([1,2,3]) ได้อาร์เรย์ 1 มิติ 3 ช่อง y = np.array([[1,2,3]]) ได้อาร์เรย์ 2 มิติ 1 แถว 3 คอลัมน์ x.shape ได้ (3,) y.shape ได้ (1,3) x.T ได้อาร์เรย์เหมือนกับ x x.T.shape ได้ (3,) y.T ได้อาร์เรย์ 3 แถว 1 คอลัมน์ y.T.shape ได้ (3,1) print(x.dot(x.T)) ได้ 14 print(x.T.dot(x)) ก็ได้ 14 print(y.dot(y.T)) ได้ array([[14]]) print(y.T.dot(y)) ได้ np.array([[1,2,3],[2,4,6],[3,6,9]])

broadcasting & element-wise operations	<pre> x = np.array([[1,2,3],[4,5,6]]) y = x+2 # [[3,4,5],[6,7,8]] y = x+[1,2,3] # [[2,4,6],[5,7,9]] y = x+np.array([1,2,3]) # [[2,4,6],[5,7,9]] y = x+np.array([[1,2,3]]) # [[2,4,6],[5,7,9]] y = x+np.array([[1],[2],[3]]).T # [[2,4,6],[5,7,9]] y = x+np.array([[1],[2],[3]]) # ผิด y = x+np.array([[1,2,3]]).T # ผิด y = x+np.array([1,2]).T # [[2,3,4],[6,7,8]] </pre>
---	---



(พยายามเขียนให้สั้น ๆ และทำงานได้เร็วสุด ๆ)

Problem	Code
<p>สมมติว่ามีอาร์เรย์ 2 มิติ M มาให้</p> <p>Input: จำนวนเต็มบวก 1 จำนวนจากแป้นพิมพ์ เก็บใน k</p> <p>Process: เปลี่ยนค่าของช่องใน M ที่หมายเลขแถวและหลักหารด้วย k ลงตัวให้มีค่าเป็น 0</p>	
<p>สมมติว่ามีอาร์เรย์ 2 มิติ M มาให้</p> <p>Input: จำนวนเต็มบวก 1 จำนวนจากแป้นพิมพ์ เก็บใน k</p> <p>Process: เปลี่ยนค่าของช่องใน M ที่หมายเลขแถวและหลักหารด้วย k ลงตัวให้มีค่าเป็น 2 เท่าของค่าเดิม</p>	
<p>สมมติว่ามีอาร์เรย์ 2 มิติ M มาให้</p> <p>ให้คำนวณหาผลต่างของค่ามากที่สุดและค่าน้อยสุดในแต่ละหลัก</p> <p>เช่น ถ้า <code>M = np.array([[3,2],[5,6],[7,1]])</code></p> <p>จะได้คำตอบ A เท่ากับ <code>np.array([4,5])</code></p> <p>(4 มาจาก $7-3$ และ 5 มาจาก $6-1$)</p>	
<p>กำหนดอาร์เรย์ X เก็บความยาวของด้านประกอบมุมฉากของรูปสามเหลี่ยมมุมฉากหลายรูป เช่น</p> <p><code>X = np.array([[3,4],[5,12],[24,7]])</code></p> <p>ให้สร้างอาร์เรย์ Y ซึ่งเก็บความยาวของด้านตรงข้ามมุมฉาก เช่นจาก X ด้านบน จะได้ Y เท่ากับ <code>array([5.,13.,25.])</code></p>	
<p>Input: จำนวนเต็มบวก 1 จำนวนจากแป้นพิมพ์ เก็บใน k</p> <p>Process: สร้างเมทริกซ์ขนาด $k \times k$ ซึ่งเก็บค่า 0 และ 1 เป็นตารางหมากรุก (มุมซ้ายบนเป็น 0) เช่น ถ้า $k = 5$ จะได้</p> <pre> C = np.array([[0,1,0,1,0] [1,0,1,0,1] [0,1,0,1,0] [1,0,1,0,1] [0,1,0,1,0]]) </pre>	

Problem	Code
<p>Input: จำนวนเต็มบวก 1 จำนวนจากแป้นพิมพ์ เก็บใน k</p> <p>Process: สร้างเมทริกซ์ขนาด $k \times k$ ซึ่งเก็บค่าเป็นตารางหมากรุกอีกแบบหนึ่ง (มุมซ้ายบนเป็น 1) โดยแทน 1 ด้วยเลขแถว (ให้เลขแถวเริ่มที่ 1) เช่น ถ้า $k = 5$ จะได้</p> <pre>C = np.array([[1,0,1,0,1] [0,2,0,2,0] [3,0,3,0,3] [0,4,0,4,0] [5,0,5,0,5]])</pre>	



Weighted Score

รายการประกวดร้องเพลงกำลังหาผู้ชนะจากการแข่งขัน โดยคะแนนของผู้เข้าแข่งขันมาจาก 3 ส่วน คือ คะแนนของกรรมการ คะแนนจากผู้ชมในห้องส่ง และคะแนนจากผู้ชมที่บ้าน ทางรายการได้กำหนดน้ำหนักของคะแนนแต่ละส่วนมาให้แล้ว ให้คำนวณคะแนนรวมของผู้เข้าแข่งขันแต่ละคน

► ข้อมูลนำเข้า

บรรทัดแรกคือจำนวนผู้เข้าแข่งขัน n เป็นจำนวนเต็ม

n บรรทัดถัดมา รับจำนวนเต็ม 3 จำนวน คือคะแนนของกรรมการ คะแนนจากผู้ชมในห้องส่ง และคะแนนจากผู้ชมที่บ้าน บรรทัดสุดท้ายคือน้ำหนักของคะแนนแต่ละส่วน เป็นจำนวนทศนิยม 3 จำนวน

► ข้อมูลส่งออก

มี n บรรทัด แต่ละบรรทัดแสดงคะแนนของผู้เข้าแข่งขันแต่ละคน

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
4 10 15 10 20 5 15 14 8 7 12 12 12 0.5 0.25 0.25	11.25 15.0 10.75 12.0
2 10 30 20 20 10 30 0.6 0.3 0.1	17.0 18.0

ตัวอย่างการเขียนโปรแกรม

โปรแกรม	คำอธิบาย
<pre> n = int(input()) S = [] for i in range(n): S.append([int(e) for e in input().split()]) W = [float(e) for e in input().split()] for i in range(n): score = [S[i][j]*W[j] for j in range(3)] print(sum(score)) </pre>	<p>โปรแกรมส่วนบนเป็นส่วนสำหรับรับข้อมูล ได้ S เป็นลิสต์ซ้อนลิสต์ของคะแนนผู้เข้าแข่งขันแต่ละคน และ W เป็นลิสต์ของน้ำหนัก เช่น</p> <p>S = [[10,30,20],[20,10,30]]</p> <p>W = [0.6,0.3,0.1]</p> <p>โปรแกรมด้านขวาทำงานถูกต้อง แต่ถ้ามีจำนวนข้อมูลมาก จะช้า เพราะใช้ลิสต์ในการประมวลผล</p>
<pre> n = int(input()) S = [] for i in range(n): S.append([int(e) for e in input().split()]) W = [float(e) for e in input().split()] score = [sum([S[i][j]*W[j] \ for j in range(3)]) \ for i in range(n)] for i in score: print(i) </pre>	<p>ยุบ for วงล่างให้เป็น list comprehension ทำงานถูกต้อง แต่ก็ยังช้าอยู่ ลองเปลี่ยนมาใช้ numpy</p>
<pre> n = int(input()) S = [] for i in range(n): S.append([int(e) for e in input().split()]) W = [float(e) for e in input().split()] S = np.array(S) W = np.array(W) for i in range(n): print(sum(S[i]*W)) </pre>	<p>เปลี่ยนลิสต์ S และ W มาใช้ numpy array สั่ง run, ใส่ข้อมูลตามตัวอย่างแรก, ผิดที่บรรทัดที่ 7</p> <p>NameError: name 'np' is not defined</p> <p>แปลว่า ระบบไม่รู้จักคำว่า np เนื่องจากไม่ได้</p> <p>import numpy as np</p>
<pre> import numpy as np n = int(input()) S = [] for i in range(n): S.append([int(e) for e in input().split()]) W = [float(e) for e in input().split()] S = np.array(S) W = np.array(W) for i in range(n): print(sum(S[i]*W)) </pre>	<p>ทำงานได้ถูกต้อง (ใช้การคูณแบบ element-wise) แต่ไม่ได้เร็วกว่าของเดิมสักเท่าไหร่ เพราะยังใช้คำสั่งของ numpy ได้ไม่เต็มที่</p>

โปรแกรม	คำอธิบาย
<pre>import numpy as np n = int(input()) S = [] for i in range(n): S.append([int(e) for e in input().split()]) W = [float(e) for e in input().split()] S = np.array(S) W = np.array(W) S *= W total_score = np.sum(S, axis = 0) for i in total_score: print(i)</pre>	<p>เราสามารถเขียน $S * W$ ได้เลย เพราะ numpy จะ broadcast อาร์เรย์ W ให้โดยอัตโนมัติ</p> <p>สั่ง run, ใส่ข้อมูลตามตัวอย่างแรก, ได้</p> <p>28</p> <p>9</p> <p>9</p> <p>ไม่ตรงกับที่แสดง พบว่าจำนวนตัวเลขไม่ครบ 4 ตัว น่าจะผิดที่ axis</p>
<pre>import numpy as np n = int(input()) S = [] for i in range(n): S.append([int(e) for e in input().split()]) W = [float(e) for e in input().split()] S = np.array(S) W = np.array(W) S *= W total_score = np.sum(S, axis = 1) for i in total_score: print(i)</pre>	<p>เปลี่ยนเป็น axis = 1</p> <p>สั่ง run, ใส่ข้อมูลตามตัวอย่างแรก, ได้</p> <p>10</p> <p>14</p> <p>10</p> <p>12</p> <p>ไม่ตรงกับที่แสดง พบว่าสิ่งที่แสดงเป็นจำนวนเต็ม แสดงว่าน่าจะมีปัญหาที่การเก็บค่า พบว่าคำสั่ง $S * W$ จะเก็บผลคูณในอาร์เรย์ S ซึ่งเก็บจำนวนเต็ม ดังนั้น ผลลัพธ์จึงเป็นจำนวนเต็ม ซึ่งไม่ถูกต้อง</p>
<pre>import numpy as np n = int(input()) S = [] for i in range(n): S.append([int(e) for e in input().split()]) W = [float(e) for e in input().split()] S = np.array(S) W = np.array(W) T = S*W total_score = np.sum(T, axis = 1) for i in total_score: print(i)</pre>	<p>เปลี่ยนเป็น $T = S*W$</p> <p>สั่ง run, ใส่ข้อมูลตามตัวอย่างแรก, ได้</p> <p>11.25</p> <p>15.0</p> <p>10.75</p> <p>12.0</p> <p>ถูกต้อง</p>
<pre>import numpy as np n = int(input()) S = [] for i in range(n): S.append([int(e) for e in input().split()]) W = [float(e) for e in input().split()] S = np.array(S) W = np.array(W) for i in S.dot(W): print(i)</pre>	<p>อีกวิธีที่ทำได้ และง่ายกว่าคือ ใช้คำสั่ง $S.dot(W)$ ซึ่งได้ผลลัพธ์เป็นอาร์เรย์ 1 มิติ มีข้อมูลแต่ละตัวคือ คะแนนรวมที่ถ่วงน้ำหนักแล้ว</p>

ตัวอย่างโจทย์ปัญหา

ค่าเช่าหนังสือ

จงเขียนโปรแกรมคำนวณหาราคาเช่าหนังสือของร้านเช่าหนังสือแห่งหนึ่ง ที่มีประเภทหนังสือให้เช่า 4 ประเภท คือ นิยาย สารคดี ทองเที่ยว และการตูน

เจ้าของร้านเช่าหนังสือต้องการทราบว่าในหนึ่งสัปดาห์ (คิด 5 วันจันทร์ถึงศุกร์) วันใดให้เช่าหนังสือเป็นจำนวนเล่มมากที่สุด เป็นจำนวนกี่เล่ม และค่าเช่าหนังสือรวมทุกประเภทในแต่ละวันเป็นจำนวนเท่าไร

► ข้อมูลนำเข้า

บรรทัดที่ 1 เป็นจำนวนเต็ม 4 จำนวนคั่นด้วยช่องว่าง แทนค่าเช่า นิยาย สารคดี ทองเที่ยว และการตูน

บรรทัดที่ 2 เป็นจำนวนหนังสือนิยายที่ถูกเช่าในวัน จ. อ. พ. พท. และ ศ. (คั่นด้วยช่องว่าง)

บรรทัดที่ 3 เป็นจำนวนหนังสือสารคดีที่ถูกเช่าในวัน จ. อ. พ. พท. และ ศ. (คั่นด้วยช่องว่าง)

บรรทัดที่ 4 เป็นจำนวนหนังสือทองเที่ยวที่ถูกเช่าในวัน จ. อ. พ. พท. และ ศ. (คั่นด้วยช่องว่าง)

บรรทัดที่ 5 เป็นจำนวนหนังสือการ์ตูนที่ถูกเช่าในวัน จ. อ. พ. พท. และ ศ. (คั่นด้วยช่องว่าง)

► ข้อมูลส่งออก

บรรทัดแรกแสดง ชื่อวันที่มีจำนวนหนังสือถูกเช่ารวมมากที่สุด และจำนวนหนังสือรวมนั้น (ให้ถือว่ามียอดเงินเท่านั้นที่ให้เช่ามากที่สุด)

บรรทัดที่สองแสดงค่าเช่ารวมของหนังสือทุกประเภทในแต่ละวัน (เรียงตั้งแต่วันจันทร์ถึงศุกร์ คั่นด้วยช่องว่าง)

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
50 30 40 20 20 50 10 15 20 30 40 20 65 35 75 30 42 70 45 40 25 35 22 55	Thu 172 5700 5400 3480 5940 4950

ขั้นตอนการทำงานของโปรแกรม

1. x = ลิสต์ที่สร้างจากข้อมูลในบรรทัดแรกจากแป้นพิมพ์ เป็นจำนวนเต็ม 4 จำนวนแทนค่าเช่าหนังสือแต่ละประเภท
2. `rentalrates` = สร้าง numpy array ที่มีค่าเริ่มต้นจากลิสต์ x
3. `sales` = สร้าง numpy array ขนาด 4 แถว 5 คอลัมน์ (แถวแทนประเภทหนังสือ คอลัมน์แทนวัน)
4. วงวนทำซ้ำ 5 โดยเปลี่ยนค่าของตัวแปร $k = 0, 1, 2, 3$ (k แทนหมายเลขประเภทหนังสือ)
5. `sales[k,] = list` ที่สร้างจากข้อมูลหนึ่งบรรทัดจากแป้นพิมพ์ เป็นจำนวนเต็ม 5 จำนวน แทนจำนวนหนังสือประเภทที่ k ของแต่ละวันที่ขายได้
6. `totalsales` = ผลรวมของจำนวนหนังสือที่ถูกเช่าในแต่ละวันคำนวณจาก `sales` (ในข้อ 3)
7. d = ตำแหน่งใน `totalsales` ที่มีค่ามากที่สุด ($d = 0$ แทนวันจันทร์, 1 แทนวันอังคาร, ..., 4 แทนวันศุกร์)
8. หาชื่อย่อวัน จาก d และ tuple ('Mon', 'Tue', 'Wed', 'Thu', 'Fri')
9. แสดง ชื่อย่อวัน ตามด้วย `totalsales[d]`
10. `salesvalues` = ค่าเช่าหนังสือรวมของหนังสือทุกประเภทในแต่ละวัน (นำ `rentalrates` มา dot กับ `sales`)
11. แสดงรายการของยอดเงินที่ขายได้จาก `salesvalues` มาแสดง (คั่นด้วยช่องว่าง)

BMI

ฟังก์ชัน `read_height_weight()` ข้างล่างนี้ อ่านข้อมูลความสูง (หน่วยเป็นเซนติเมตร) และน้ำหนัก (หน่วยเป็นกิโลกรัม) มาสร้าง `numpy array` แบบสองมิติ ดังตัวอย่างในตารางข้างล่างนี้ (บรรทัดแรกคือจำนวนข้อมูล บรรทัดที่ตามมาคือ ความสูงกับน้ำหนัก)

Input	ผลที่ได้จาก <code>read_height_weight()</code>
4 160 60 155 62 170 54 180 55	<code>array([[160, 60], [155, 62], [170, 54], [180, 55]])</code>

จงเขียนฟังก์ชัน `cm_to_m(x)` และ `cal_bmi(hw)` ในโปรแกรมข้างล่างนี้ ที่มีข้อกำหนดของพารามิเตอร์ และผลลัพธ์ที่ได้ตามตารางนี้

Function	Input parameter	Return value
<code>cm_to_m(x)</code>	array หนึ่งมิติ เก็บความสูงหน่วยเป็นเซนติเมตร เช่น <code>array([160, 155, 170, 180])</code>	array หนึ่งมิติเก็บความสูงหน่วยเป็นเมตร เช่น <code>array([1.6, 1.55, 1.7, 1.8])</code>
<code>cal_bmi(hw)</code>	array สองมิติ ขนาด n แถว 2 คอลัมน์ แต่ละแถว แทนข้อมูลหนึ่งคู่ คอลัมน์ 0 เก็บความสูง (เซนติเมตร) คอลัมน์ 1 เก็บน้ำหนัก (กิโลกรัม) เช่น <code>array([[160, 60], [155, 62], [170, 54], [180, 55]])</code>	array หนึ่งมิติเก็บ bmi ที่คำนวณจากความสูงและน้ำหนักใน Input parameter ที่ได้รับ เช่น <code>array([23.4375, 25.80645161, 18.68512111, 16.97530864])</code>

และเขียนคำสั่ง

- หาค่าเฉลี่ยของ bmi ทั้งหมดที่คำนวณได้ เก็บใส่ตัวแปร `avg_bmi` และ
- นับจำนวน bmi ที่คำนวณได้ที่มีค่าน้อยกว่า 18.5

```

import numpy as np

def read_height_weight():
    list_hw = []
    for k in range(int(input())) :
        h,w = input().split()
        list_hw.append((int(h),int(w)))
    return np.array(list_hw)

def cm_to_m(x):
    # ???

def cal_bmi(hw):
    # ???

def main():
    hw = read_height_weight()
    bmi = cal_bmi(hw)
    avg_bmi = -----
    count_underweight = -----
    print('average bmi =', avg_bmi)
    print('#bmi < 18.5 =', count_underweight)

exec(input().strip())

```

► ข้อมูลนำเข้า

คำสั่งในการทดสอบฟังก์ชันที่เขียน

► ข้อมูลส่งออก

ผลที่ได้จากคำสั่งที่ป้อนเป็นข้อมูลนำเข้า

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
<code>x=np.array([160,150,140]); print(cm_to_m(x)); print(x)</code>	<code>[1.6 1.5 1.4]</code> <code>[160 150 140]</code>
<code>d=np.array([[100,30],[120,36]]); print(cal_bmi(d))</code>	<code>[30. 25.]</code>
<code>main()</code> 4 160 60 155 62 170 54 180 55	average bmi = 21.2260953405 #bmi < 18.5 = 1

การคำนวณจำนวนฟีโบนัคชีโดยใช้การยกกำลังเมทริกซ์อย่างรวดเร็ว

0, 1, 1, 2, 3, 5, 8, ... เป็นลำดับของจำนวนฟีโบนัคชี ($F_0 = 0, F_1 = 1, F_2 = 1, \dots$) วิธีหนึ่งในการหา F_n คือคำนวณผลการยกกำลัง n ของเมทริกซ์ A ที่แสดงด้านล่าง จะได้ผลเป็นเมทริกซ์ขนาด 2×2 ที่มี F_n อยู่ที่ยูนิตบนขวาของเมทริกซ์ เช่น

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$$

$$\text{หา } F_3 \text{ คำนวณ } A^3 = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}^3 = \begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix} \text{ ได้ } F_3 = 2 \quad \text{หา } F_4 \text{ คำนวณ } A^4 = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}^4 = \begin{bmatrix} 2 & 3 \\ 3 & 5 \end{bmatrix} \text{ ได้ } F_4 = 3$$

ถ้าคิดดูดี ๆ จะพบว่าการหาด้วยวิธีข้างต้นนี้คือการหาค่ายกกำลัง ซึ่งเราก็ไม่น่าหาแบบค่อย ๆ คูณไปทีละครั้ง เช่น การหา A^{10} ก็ไม่น่าใช้วิธีที่เริ่มด้วยเมทริกซ์เอกลักษณ์ I แล้วคูณด้วย A ไป 10 ครั้ง น่าจะใช้วิธีการหา A^5 แล้วจับมาคูณกับตัวเองก็ได้ A^{10} นั่นคือ

$$A^n = \begin{cases} I & n = 0 \\ (A^{\lfloor n/2 \rfloor})^2 & n \text{ is even} \\ A(A^{\lfloor n/2 \rfloor})^2 & n \text{ is odd} \end{cases} \quad A = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \quad I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

จงเขียนฟังก์ชัน `fib(n,k)` เพื่อคำนวณ $F_n \% k$ ด้วยวิธีข้างต้นนี้ โดยใช้คำสั่งของ `numpy` เพื่อคูณเมทริกซ์ (หมายเหตุ: หลังการคูณเมทริกซ์ทุกครั้งให้นำผลที่ได้มา `% k` `numpy` จะทำ `% k` แบบ `element-wise` ในเมทริกซ์)

```
import numpy as np

def fib(n,k):
    # ???

n,k = [int(e) for e in input().split()]
print( fib(n,k) )
```

► ข้อมูลนำเข้า

จำนวนเต็ม 2 ค่า n กับ k ($0 \leq n \leq 10^{13}, 0 \leq k \leq 100000$)

► ข้อมูลส่งออก

แสดงค่า $F_n \% k$

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
0 10	0
2 10	1
89 10	9
11111 111	55
1234567890 1234	162
10000000000000 999	600

สรุปเนื้อหา

- คลาส ใช้สร้างประเภทข้อมูลใหม่ที่ต้องการ โดยเรากำหนดได้ว่า จะเก็บข้อมูลย่อยอะไรบ้าง และทำงานอะไรได้บ้าง
- คลาส คือ ประเภทข้อมูล, อ็อบเจกต์ คือ ตัวข้อมูล เช่น `b1 = Book(...)` ได้ว่า `b1` เป็นอ็อบเจกต์ของคลาส `Book`
- เมทอด คือ ฟังก์ชันซึ่งเป็นบริการของคลาส

ตัวอย่าง: คลาส `Book`

- คลาส `Book` ข้างล่างนี้ ใช้สร้างอ็อบเจกต์ที่เก็บข้อมูลของหนังสือ (ชื่อ ราคา และคำสำคัญต่าง ๆ ของหนังสือ)
- ทุกเมทอดของคลาสมต้องมีตัวแปร `self` เป็นพารามิเตอร์แรก ซึ่งแทนอ็อบเจกต์ที่จะใช้บริการ เมื่อเรียกใช้ตัวแปรประจำอ็อบเจกต์ภายในคลาส จะต้องมี `self`. นำหน้าเสมอ
- เมทอด `__init__` (เรียกว่า constructor) ใช้สร้างอ็อบเจกต์ของคลาส โดยบอกว่าอ็อบเจกต์จะเก็บข้อมูลอะไรบ้าง และจะถูกเรียกเมื่อมีการสร้างอ็อบเจกต์ โดย `self` จะแทนอ็อบเจกต์ที่เพิ่งสร้าง
- เมทอด `__str__` คืนค่าสตริงของอ็อบเจกต์ จะถูกเรียกเมื่อใช้งานฟังก์ชัน `str` หรือ `print`
- เมทอด `__lt__` ใช้สำหรับเปรียบเทียบอ็อบเจกต์ของ `Book` ในที่นี้จะเปรียบเทียบโดยใช้ราคาก่อน ถ้าราคาเท่ากัน จะใช้ชื่อหนังสือเป็นตัวเปรียบเทียบ เมทอดนี้จะถูกเรียกเมื่อเปรียบเทียบอ็อบเจกต์ด้วย `<` หรือใช้ฟังก์ชัน `sort`
- นอกจากนี้ยังมีบริการ ปรับราคา และขอคำสำคัญร่วมของหนังสือ 2 เล่ม ผ่านเมทอด `update_price` และ `get_common_keywords` (สามารถเขียนเมทอดอื่น ๆ เพิ่มเติมตามต้องการ เป็นการเพิ่มความสามารถของอ็อบเจกต์)
- เมทอดอาจจะคืนค่าหรือไม่ก็ได้ โดยทั่วไปเมทอดที่ไม่คืนผลลัพธ์มักเป็นเมทอดที่มีการเปลี่ยนแปลงค่าภายในอ็อบเจกต์

```
class Book:
    def __init__(self, title, price, keywords):
        self.title = title; self.price = price; self.keywords = set(keywords)

    def __lt__(self, rhs):
        if self.price != rhs.price: return self.price < rhs.price
        else: return self.title < rhs.title

    def __str__(self):
        return self.title + ' ($' + str(self.price) + ')'

    def update_price(self, new_price):
        self.price = new_price

    def get_common_keywords(self, other):
        return self.keywords & other.keywords

b1 = Book('Python', 99, ['code','computer']) # using __init__
b2 = Book('Calculus', 199, ['maths'])
b3 = Book('Physics', 99, ['science','maths'])

b1.update_price(199)
print(Book.get_common_keywords(b2,b3)) # {'maths'}
if b3 < b2: print(b1) # using __lt__ & __str__
books = [b1,b2,b3]
books.sort() # using __lt__
print(books[0],',',books[1],',',books[2]) # using __str__
# 'Physics ($99) , Calculus ($199) , Python ($199)'
```

- การเรียกเมทอดทำได้ 2 แบบ
 - เรียกผ่านชื่อคลาส เช่น `Book.get_common_keywords(b2,b3)`
 - เรียกผ่านอ็อบเจกต์ เช่น `b2.get_common_keywords(b3)`
 (b2 จะถูกแทนใน `self` และ b3 จะถูกแทนใน `other` โดยอัตโนมัติ และทั้งสองคำสั่งนี้ทำงานเหมือนกัน)
- `str(b1)` เหมือนกับการเรียก `Book.__str__(b1)` หรือ `b1.__str__()`
- `print(b1)` เหมือนกับการเรียก `print(str(b1))`
- `b1 < b2` เหมือนกับการเรียก `Book.__lt__(b1,b2)` หรือ `b1.__lt__(b2)`
- การเรียก `books.sort()` จะเรียงลำดับอ็อบเจกต์ของ `Book` จากน้อยไปมาก โดยเปรียบเทียบกับ `__lt__`

เรื่องพิศบอย

ลืม <code>self</code> ในเมทอด ไม่ได้ประกาศ <code>self</code> เป็นพารามิเตอร์ ไม่มี <code>self</code> . นำหน้าตัวแปรของอ็อบเจกต์	<pre>class A: def __init__(self, x, y): d = dict() # แก้เป็น self.d = dict() d[x] = y # แก้เป็น self.d[x] = y def total(): # แก้เป็น def total(self): return sum(d.values()) # แก้เป็น return sum(self.d.values())</pre>
การกำหนดค่าอ็อบเจกต์ให้กับตัวแปรด้วย เครื่องหมายเท่ากับ จะทำให้ตัวแปรนั้นชี้ไปที่ อ็อบเจกต์เดียวกัน	<pre>class B: def __init__(self, b): self.b = b b1 = B(10) b2 = b1 # b1, b2 อ้างอิงอ็อบเจกต์เดียวกัน b2.b = 5 print(b1.b) # 5</pre>
indent ผิด	<pre>class C: def __init__(self, c): self.c = c def double(self): # indent ผิด กลายเป็นฟังก์ชันทั่วไป self.c *= 2 # เรียก double(c1) ได้ # แต่จะเรียก c1.double() ไม่ได้ # เรียก C.double(c1) ก็ไม่ได้</pre>



1. เติมเมทอดของคลาสนิสิตตาม comment ที่กำหนดให้สมบูรณ์

```
class Nisit:
    def __init__(self, name, year, faculty):
        # n = Nisit('Krit', 4, 'Engineering')

    def __str__(self):
        # คืนสตริงของนิสิต เช่น 'Krit (year 4) Engineering'

    def __lt__(self, rhs):
        # เรียงลำดับนิสิตด้วยคณะตามพจนานุกรม ถ้าอยู่คณะเดียวกัน ให้เรียงลำดับด้วยชั้นปีจากน้อยไปมาก
        # ถ้าอยู่คณะและชั้นปีเดียวกัน ให้เรียงลำดับด้วยชื่อตามพจนานุกรม
        # เช่น Nisit('Krit', 4, 'Engineering') < Nisit('Boy', 3, 'Science')
        #       Nisit('Prim', 2, 'Engineering') < Nisit('Krit', 4, 'Engineering')
        #       Nisit('Joey', 2, 'Engineering') < Nisit('Prim', 2, 'Engineering')
```

2. เติมเมทอดของคลาสรถยนต์ตาม comment ที่กำหนดให้สมบูรณ์

```
class Car:
    def __init__(self, license, brand, color):
        # c = Car('AA1234', 'Honda', 'White')
        # มีตัวแปร report สำหรับเก็บข้อมูลประวัติการซ่อมบำรุง โดยกำหนดค่าเริ่มต้นเป็นลิสต์ว่าง

    def __str__(self):
        # คืนสตริงของรถยนต์ เช่น 'AA1234 - White Honda'

    def __lt__(self, rhs):
        # เปรียบลำดับรถยนต์โดยเปรียบเทียบป้ายทะเบียนรถแบบสตริง

    def add_report(self, new_report):
        # เพิ่มประวัติการซ่อมบำรุง โดยไม่ต้องคืนค่า
        # ตัวแปร new_report เก็บ tuple (วันที่, คำอธิบาย, ราคา)
        # เช่น c.add_report( ('25 May 2017', 'change tires', 1500) )

    def total_payment(self):
        # คืนค่าใช้จ่ายทั้งหมดที่ใช้ในการซ่อมบำรุงที่ผ่านมา

    def max_payment(self):
        # คืนลิสต์ของประวัติการซ่อมบำรุง (วันที่, คำอธิบาย, ราคา) ทุกรายการ ที่มีค่าใช้จ่ายมากที่สุด
        # กรณีที่รถยนต์ไม่มีประวัติการซ่อมบำรุงเลย ให้คืนค่าลิสต์ว่าง
```

3. จาก class Book ให้เติมเมทอดของ class ShoppingCart สำหรับการซื้อหนังสือผ่านเว็บไซต์ ดังนี้

```
class ShoppingCart:
    def __init__(self, id):
        self.id = id
        self.books = []
        # books เก็บลิสต์ของหนังสือในตะกร้าพร้อมจำนวน เช่น [[b1,2],[b3,7]]

    def add_book(self, book, n):
        # เพิ่มข้อมูลการซื้อหนังสือ book เพิ่มอีก n เล่ม โดยไม่ต้องคืนค่า
        # หากไม่มีหนังสือเล่มนี้ในตะกร้า ให้เพิ่มลิสต์ [book, n] ต่อท้าย books
        # หากเคยมีข้อมูลหนังสือเล่มนี้ในตะกร้าแล้ว ให้เพิ่มจำนวนที่ซื้ออีก n เล่ม
        # เช่น ถ้า books = [[b1,2]] และเราสั่ง add_book(b1,3) จะได้ books = [[b1,5]]

    def delete_book(self, book):
        # ลบข้อมูลการซื้อหนังสือ book ออกจากตะกร้า โดยไม่ต้องคืนค่า
        # ถ้าในตะกร้าไม่มีหนังสือ book ไม่ต้องทำอะไร

    def get_total(self):
        # คืนค่าราคารวมของหนังสือทั้งหมดในตะกร้า

    def __lt__(self, rhs):
        # ตะกร้าที่มีราคารวมของหนังสือน้อยกว่า จะเป็นตะกร้าที่น้อยกว่า
```


4. ข้างล่างนี้แสดงคลาส Station และคลาส BTScard (อ่านคำอธิบายของแต่ละคลาสจาก comment ที่เขียน)

สถานีรถไฟฟ้าเป็นอ็อบเจกต์ของคลาส Station และบัตรโดยสารแบบเติมเงินแต่ละใบเป็นอ็อบเจกต์ของคลาส BTScard จงเติมคำสั่งในเมทอด add_value, enter, leave และ __lt__ ของคลาส BTScard ให้ทำงานตาม comment ที่เขียน (เมทอดอื่นที่ได้เขียนคำสั่งไว้ ทำงานถูกต้องแล้ว) ดูตัวอย่างการใช้งานข้างล่างนี้ประกอบ

```
s1 = Station(1,'Siam'); s2 = Station(3,'Mo Chit'); s3 = Station(5,'Asok')
c1 = BTScard(123, 5); c2 = BTScard(999, 10)
# -----
c1.add_value(100)      # c1 มีเงินในบัตร 105 บาท
p = c1.enter(s1)       # p = True
p = c1.enter(s3)       # p = False (แตะเข้าสถานีหลังจากแตะเข้าไปแล้ว)
p = c1.leave(s2)       # c1 เหลือเงินในบัตร 95 บาท โดย p = (95, 0)
# -----
p = c2.enter(s3)       # p = True
p = c2.leave(s1)       # c2 มีเงินในบัตร 10 บาทไม่พอจ่ายค่าโดยสาร โดย p = (10, -1)
c2.add_value(50)       # c2 มีเงินในบัตร 60 บาท
p = c2.leave(s1)       # c2 เหลือเงินในบัตร 40 บาท โดย p = (40, 0)
p = c2.leave(s2)       # p = (40, -2) (ยังไม่ได้แตะเข้าสถานี จึงไม่มีสถานีต้นทาง)
p = c2.enter(s2)       # p = True
p = c1 < c2            # p = False
```

```
class Station:
    # คลาสของสถานีรถไฟฟ้า
    def __init__(self, id, name):
        # สร้างสถานีที่มีหมายเลข (id) และชื่อสถานี (name)
        self.sid = int(id)
        # กำหนดให้หมายเลขสถานีเป็นจำนวนเต็ม โดยสถานีที่ติดกัน
        self.name = name
        # มีค่าห่างกัน 1

    def get_price(self, other):
        # คำนวณค่าโดยสารระหว่างสถานี self และ other
        return abs(self.sid - other.sid)*5

# -----

class BTScard:
    # คลาสของบัตรโดยสารแบบเติมเงิน
    def __init__(self, id, value):
        # สร้างบัตรโดยสารที่มีเลขบัตร (id) และเงินเริ่มต้น (value)
        self.cid = id
        # self.station เก็บว่าสถานีต้นทางคือสถานีอะไร
        self.value = value
        # โดยถ้าบัตรไม่ได้อยู่ระหว่างการเดินทาง จะเก็บค่า
        self.station = ''
        # สถานีต้นทางนี้เป็นสตริงว่าง ๆ

    def __str__(self):
        return '('+str(self.cid)+','+str(self.value)+')'

    def add_value(self, x):
        # เพิ่มเงินในบัตรโดยสารเท่ากับ x โดยไม่ต้อง return
```

```
def enter(self, station):
    # แตะบัตรเพื่อเข้าสู่สถานีรถไฟฟ้า ให้ใช้คำว่า บัตรนี้ไม่ได้แตะเข้าที่สถานีอื่นมาก่อน
    # ถ้าไม่มีการแตะเข้ามาก่อน ให้เปลี่ยนค่าสถานีต้นทางเป็น station แล้ว return True
    # แต่ถ้ามีการแตะเข้าสถานีอื่นมาก่อน ให้ return False โดยไม่เปลี่ยนข้อมูลสถานีต้นทางของบัตรโดยสาร
```

```
def leave(self, station):
    # แตะบัตรเพื่อออกจากสถานีรถไฟฟ้า ให้ใช้คำว่า บัตรนี้มีข้อมูลสถานีต้นทางอยู่
    # ถ้าไม่มีข้อมูลสถานีต้นทาง ให้ return tuple ของเงินในบัตรและ -2
    # ถ้ามีสถานีต้นทาง แต่จำนวนเงินในบัตรไม่พอจ่ายค่าโดยสาร ให้ return tuple ของเงินในบัตรและ -1
    # ถ้ามีสถานีต้นทาง และจำนวนเงินในบัตรพอจ่ายค่าโดยสาร ให้ลบค่าโดยสารออกจากจำนวนเงินในบัตร
    # เปลี่ยนสถานีต้นทางเป็นสถานีจริงแล้ว return tuple ของเงินในบัตรหลังหักค่าโดยสารและ 0
```

```
def __lt__(self, rhs): # บัตรโดยสารที่มีเงินในบัตรน้อยกว่า จะถือว่าน้อยกว่า
```

ตัวอย่างการแก้โจทย์ปัญหา

Bus

ให้เขียนคลาสของรถเมล์ซึ่งมีเมทอดดังนี้

1. `__init__` สร้างรถเมล์ 1 คัน รับพารามิเตอร์ จำนวนคนบนรถ `people` และค่าโดยสาร `fare`
2. `__str__` คืนค่าสตริงซึ่งบอกจำนวนคนบนรถและค่าโดยสาร
3. `__lt__` เปรียบเทียบรถเมล์โดยพิจารณาค่าโดยสารรวมของรถ (จำนวนคนบนรถคูณค่าโดยสารต่อคน)
4. `people_in` เพิ่มจำนวนคนบนรถ `k` คน ไม่คืนค่า
5. `people_out` ลดจำนวนคนบนรถ `k` คน (หากจำนวนคนน้อยกว่า 0 จะต้องแก้ไขจำนวนคนเป็น 0) ไม่คืนค่า
6. `change_fare` เปลี่ยนค่าโดยสารเป็นค่าโดยสารใหม่ `new_fare` ไม่คืนค่า

▶ ตัวอย่าง

<pre>b1 = Bus(10, 5) b2 = Bus(8, 7) if b1 < b2: print(b1) else: print(b2) b1.people_in(3) b1.people_out(6) b1.change_fare(12) print(b1)</pre>	<pre># b1 has 10 people with fare = 5 # b2 has 8 people with fare = 7 # b1 < b2 is True (10*5 < 8*7) this bus has 10 people with fare = 5 # b1 has 13 people with fare = 5 # b1 has 7 people with fare = 5 # b1 has 7 people with fare = 12 this bus has 7 people with fare = 12</pre>
---	--

ตัวอย่างการเขียนโปรแกรม

โปรแกรม	คำอธิบาย
<pre>class Bus: def __init__(people, fare): people = people fare = fare def __str__(): return 'this bus has ' + str(people) \ + ' people with fare = ' + str(fare) def __lt__(rhs): return people*fare < \ rhs.people*rhs.fare def people_in(k): return people += k def people_out(k): return people -= k def change_fare(new_fare): return fare = new_fare</pre>	<p>ทำงานไม่ถูกต้อง มีจุดผิดดังนี้</p> <ul style="list-style-type: none">- ไม่มีการใช้ <code>self</code>- เมทอด <code>people_in</code>, <code>people_out</code> และ <code>change_fare</code> ต้องไม่คืนค่า- เมทอด <code>people_out</code> ไม่ได้ตรวจสอบว่าจำนวนคนน้อยกว่าศูนย์หรือไม่

โปรแกรม	คำอธิบาย
<pre> class Bus: def __init__(self, people, fare): self.people = people self.fare = fare def __str__(self): return 'this bus has ' + \ str(self.people) + \ ' people with fare = ' + \ str(self.fare) def __lt__(self, rhs): return self.people*self.fare < \ rhs.people*rhs.fare def people_in(self, k): self.people += k def people_out(self, k): self.people = max(0,self.people-k) def change_fare(self, new_fare): self.fare = new_fare </pre>	<p>ทำงานถูกต้อง สังเกตการใช้งาน</p> <p><code>self.people = max(0,self.people-k)</code> ว่าทำการแก้ไขจำนวนคนหากน้อยกว่าศูนย์ให้แล้ว</p>
<pre> class Bus: def __init__(self, people, fare): self.fare = fare self.total = people*fare def __str__(self): return 'this bus has ' + \ str(self.total//self.fare) + \ ' people with fare = ' + \ str(self.fare) def __lt__(self, rhs): return self.total < rhs.total def people_in(self, k): self.total += self.fare*k def people_out(self, k): self.total = max(0,self.total- \ self.fare*k) def change_fare(self, new_fare): self.total = \ self.total//self.fare*new_fare self.fare = new_fare </pre>	<p>เราสามารถเขียนคลาส Bus แบบอื่นได้ เช่น</p> <p>แทนที่จะเก็บจำนวนคนและค่าโดยสาร</p> <p>อาจจะเก็บค่าโดยสารต่อคนและค่าโดยสารรวมก็ได้ แต่ก็ต้องปรับการทำงานของคลาสให้ถูกต้อง (ในเมทอด <code>change_fare</code> ถ้าทำการเปลี่ยนค่า <code>self.fare</code> ก่อน จะทำให้เมทอดทำงานผิดได้)</p>

ตัวอย่างโจทย์ปัญหา

เศษส่วน

กำหนดคลาสของเศษส่วน ประกอบด้วยเศษ (numerator) และส่วน (denominator) และมีเมทอด 4 เมทอดคือ เมทอดสำหรับการแสดงผลเป็นสตริง เมทอดการทำเศษส่วนอย่างต่ำ เมทอดการบวก และเมทอดการคูณ ดังนี้ (ขอให้สังเกตการเรียกใช้งานเมทอด ว่าสามารถเรียกได้หลายแบบ)

```
def gcd(x,y):
    if x%y == 0: return y
    return gcd(y,x%y)

class Fraction:
    def __init__(self,a,b):
        self.numerator = _____
        self.denominator = _____

    def __str__(self):
        # ???

    def simplify(self):
        g = gcd(self.numerator,self.denominator)
        return Fraction(self.numerator//g,self.denominator//g)

    def add(self,other):
        # ???

    def multiply(self,other):
        ans_num = self.numerator * other.numerator
        ans_denom = self.denominator * other.denominator
        return Fraction(ans_num,ans_denom).simplify()

a,b,c,d = [int(e) for e in input().split()]
fraction1 = _____
fraction2 = _____

print(fraction1.add(fraction2))
print(Fraction.multiply(fraction1,fraction2))
```

โจทย์ได้เขียนเมทอดการทำเศษส่วนอย่างต่ำและเมทอดการคูณมาให้แล้ว ให้เขียนเติมส่วนอื่น ๆ ให้สมบูรณ์ สำหรับการบวก เมื่อบวกแล้วให้ตอบเป็นเศษส่วนอย่างต่ำด้วย สามารถเรียกใช้เมทอด simplify ได้

► ข้อมูลนำเข้า

มีบรรทัดเดียว เป็นจำนวนเต็มบวก a b c d ซึ่งแทนเศษส่วน a/b และ c/d

► ข้อมูลส่งออก

มี 2 บรรทัด แสดงผลบวกและผลคูณของเศษส่วนที่กำหนดให้

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
1 7 3 7	4/7 3/49
1 2 1 3	5/6 1/6
1 8 3 8	1/2 3/64
2 3 1 2	7/6 1/3

คลาสของไพ่

ข้างล่างนี้แสดงการเรียกใช้คลาส Card ซึ่งแทนไพ่ 1 ใบ ประกอบด้วย ค่าของไพ่ (value) ซึ่งเป็นสตริง ('A', '2', '3', ... , '10', 'J', 'Q', 'K') และ ดอกของไพ่ (suit) ซึ่งเป็นสตริงเช่นกัน ('club', 'diamond', 'heart', 'spade') โปรแกรมข้างล่างนี้รับไพ่เข้ามาหลายใบมาสร้างเป็นลิสต์ของไพ่ (cards) และมีการเรียกใช้เมทอดต่าง ๆ ของคลาส Card ให้นิสิตเขียนเมทอดต่าง ๆ ของคลาส Card ให้สมบูรณ์

```
class Card:
    def __init__(self, value, suit):
        # ???

    def __str__(self):
        # ???

    def getScore(self):
        # ???

    def sum(self, other):
        # ???

    def __lt__(self, rhs):
        # ???

n = int(input())
cards = []
for i in range(n):
    value, suit = input().split()
    cards.append(Card(value, suit))
for i in range(n):
    print(cards[i].getScore())
print("-----")
for i in range(n-1):
    print(Card.sum(cards[i], cards[i+1]))
print("-----")
cards.sort()
for i in range(n):
    print(cards[i])
```

รายละเอียดต่าง ๆ ของคลาส Card และเมทอดของคลาส Card

- เมทอด `getScore` จะคืนค่าคะแนนของไพ่เป็นจำนวนเต็ม ตามกฎดังนี้
 - ไพ่ที่มีค่า A จะมีคะแนน 1 คะแนน
 - ไพ่ที่มีค่า 2 ถึง 10 จะมีคะแนนเท่ากับค่าของไพ่ คือ 2 ถึง 10 คะแนน ตามลำดับ
 - ไพ่ที่มีค่า J, Q, K จะมีคะแนน 10 คะแนน
- เมทอด `sum` จะคืนค่าผลรวมคะแนนของไพ่สองใบและ mod ด้วย 10 เช่น
 - `Card.sum(Card('7', 'club'), Card('2', 'heart'))` ได้ผลลัพธ์เป็น 9
 - `Card.sum(Card('J', 'spade'), Card('5', 'diamond'))` ได้ผลลัพธ์เป็น 5
- การเรียงลำดับของไพ่เป็นดังนี้
 - ค่าของไพ่เรียงตามลำดับดังนี้ $3 < 4 < 5 < \dots < 10 < J < Q < K < A < 2$
 - ดอกของไพ่เรียงตามลำดับดังนี้ `club < diamond < heart < spade`
 - ถ้าไพ่สองใบมีค่าไม่เท่ากัน ไพ่ที่มีค่ามากกว่าจะเป็นไพ่มากกว่า
 - ถ้าไพ่สองใบมีค่าเท่ากัน ไพ่ที่มีดอกสูงกว่าจะเป็นไพ่มากกว่า

► ข้อมูลนำเข้า

บรรทัดแรกมีจำนวนเต็ม n แทนจำนวนไพ่ที่จะรับเข้ามา

n บรรทัดต่อมา แต่ละบรรทัดมีค่าและดอกของไพ่แต่ละใบ คั่นด้วยช่องว่าง

► ข้อมูลส่งออก

มี $3n+1$ บรรทัด

n บรรทัดแรก แสดงคะแนนของไพ่แต่ละใบ ตามด้วยขีดคั่น 1 บรรทัด

$n-1$ บรรทัดถัดมา แสดงคะแนนรวมของไพ่ 2 ใบที่ติดกันในลำดับ ตามด้วยขีดคั่น 1 บรรทัด

n บรรทัดสุดท้าย แสดงไพ่เรียงตามลำดับ

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
5	1
A spade	10
K heart	10
K club	7
7 diamond	2
2 spade	-----
	1
	0
	7
	9

	(7 diamond)
	(K club)
	(K heart)
	(A spade)
	(2 spade)

กระปุกออมสิน

จงเขียนคลาส piggybank เพื่อผลิตอ็อบเจกต์กระปุกออมสิน ที่สามารถหยอดเหรียญมูลค่าอะไรก็ได้ โดยจำกัดจำนวนเหรียญรวมทุกแบบในกระปุกแล้วห้ามเกิน 100 เหรียญ (ถ้าหยอดแล้วเกินไม่รับเพิ่ม) ตามโครงของคลาสและตัวอย่างการใช้งานข้างล่างนี้

โครงของคลาส piggybank	ตัวอย่างการใช้งาน piggybank
<pre>class piggybank: def __init__(self): # มีตัวแปร self.coins เก็บ dict เริ่มต้นให้ว่าง ๆ # มี key เป็นมูลค่าเหรียญ และ value เป็นจำนวนเหรียญ def add(self, v, n): # ถ้าเพิ่มจำนวนเหรียญในกระปุกอีก n เหรียญแล้วเกิน 100 # จะไม่ให้เพิ่ม ให้คืน False แทนว่า เพิ่มไม่สำเร็จ # แปลง v เป็น float ก่อน (เพิ่ม 5 กับ 5.0 จะได้เหมือนกัน) # ถ้ากระปุกไม่เคยมีเหรียญ v ทำ self.coins[v] = 0 # ทำคำสั่ง self.coins[v] += n # คืน True แทนว่าเพิ่มสำเร็จ def __float__(self): # นำค่าของเหรียญคูณกับจำนวนเหรียญ ของเหรียญทุกแบบ # ต้องคืนจำนวนแบบ float เท่านั้น อยากรู้คืนศูนย์ ก็ต้อง 0.0 def __str__(self): # คืนสตริงที่แสดงจำนวนเหรียญแต่ละแบบตามตัวอย่าง # โดยให้เรียงเหรียญตามมูลค่าจากน้อยไปมาก</pre>	<pre>p1 = piggybank() print(float(p1)) # 0.0 p1.add(0.25, 4) # เพิ่มเหรียญ 25 สตางค์ 4 เหรียญ print(float(p1)) # 1.0 p1.add(0.50, 1) # เพิ่มเหรียญ 50 สตางค์ 1 เหรียญ print(float(p1)) # 1.5 p1.add(10, 1) # เพิ่มเหรียญ 10 บาท 1 เหรียญ print(float(p1)) # 11.5 print(p1) # {0.25:4, 0.5:1, 10.0:1} print(p1.add(10, 1)) # True เพิ่มได้ print(float(p1)) # 21.5 print(p1.add(1,94)) # False เพิ่มไม่ได้ เกิน 100 เหรียญ print(float(p1)) # 21.5</pre>

เมทอด `__float__` ถูกเรียกเมื่อ `float(p)` ทำงาน โดยที่ `p` เป็น `piggybank` ได้ผลลัพธ์เป็น `float` แทนค่าของ `p`

เมทอด `__str__` ถูกเรียกเมื่อ `str(p)` ทำงาน โดยที่ `p` เป็น `piggybank` ได้ผลลัพธ์เป็นสตริงแทนค่าของ `p`

► การส่งตรวจ

ให้นำโปรแกรมข้างล่างนี้ ต่อท้าย `class piggybank` ที่เขียนข้างบนนี้ แล้วจึงส่งให้ Grader ตรวจ

```
cmd1 = input().split(';')
cmd2 = input().split(';')
p1 = piggybank(); p2 = piggybank()
for c in cmd1: eval(c)
for c in cmd2: eval(c)
```

► ข้อมูลนำเข้า

คำสั่งต่าง ๆ เพื่อการทดสอบคลาส

► ข้อมูลส่งออก

ผลการทำงานของโปรแกรมข้างบนที่อาศัยคลาส `piggybank`

► ตัวอย่าง

Input (จากแป้นพิมพ์)	Output (ทางจอภาพ)
<code>p1.add(1.11,2); print(float(p1), p1)</code> <code>print(float(p2), p2)</code>	2.22 {1.11:2} 0.0 {}
<code>p1.add(0.25,1);p1.add(5,1);p1.add(0.25,2);p1.add(5.0,1)</code> <code>print(float(p1), str(p1))</code>	10.75 {0.25:3, 5.0:2}
<code>p1.add(0.25,1); print(p1.add(0.25,100))</code> <code>print(p1.add(0.25,99)); print(float(p1))</code>	False True 25.0



ดร. สุกชัย สุตันทวิบูลย์

Intania 63

Principal Member of Technical Staff
Advanced Micro Devices Inc. (AMD)
California, USA

การใช้คอมพิวเตอร์ช่วยออกแบบ (Computer Aid Design : CAD) มีความสำคัญมากสำหรับการออกแบบ System on Chips ซึ่งประกอบด้วยทรานซิสเตอร์หลายพันล้านตัว วิศวกรออกแบบ chips ที่ประสบความสำเร็จส่วนใหญ่มีความรู้ความสามารถในการใช้โปรแกรม CAD และสามารถเขียนโปรแกรมเสริมเพื่อให้ตนเองทำงานออกแบบได้เร็วขึ้น การเขียนโปรแกรมให้ได้ดีจึงต้องเริ่มด้วยทักษะพื้นฐานด้าน Programming แล้วเสริมด้วยองค์ความรู้เรื่อง Algorithms และ Data Structures จึงจะทำให้เขียนโปรแกรมที่มีประสิทธิภาพ เช่น ทำงานได้เร็ว ใช้หน่วยความจำน้อย และใช้พลังงานต่ำ เป็นต้น

11 : Solutions to Exercises

unth 1

► ข้อที่ 1

```
h = int(input())
m = int(input())
s = int(input())
total = 60*60*h + 60*m + s
print( total )
```

► ข้อที่ 2

```
import math
x = float(input())
y = 2 - x + 3/7*x**2 - 5/11*x**3 + \
    math.log(x,10)
print(y)
```

► ข้อที่ 3

```
a = float(input())
x = 1
x = (x + a/x)/2
x = (x + a/x)/2
x = (x + a/x)/2
x = (x + a/x)/2
print(x)
```

► ข้อที่ 4

```
v1,v2,v3 = [float(e) \
            for e in input().split()]
u1,u2,u3 = [float(e) \
            for e in input().split()]
dotp = v1*u1 + v2*u2 + v3*u3
print(dotp)
```

► ข้อที่ 5

```
import math
x1,y1,x2,y2 = \
    [float(e) for e in input().split()]
d = math.sqrt((x1-x2)**2+(y1-y2)**2)
print(d)
```

► ข้อที่ 6

```
import math
r,theta = [float(e) \
           for e in input().split()]
x = r*math.cos(theta)
y = r*math.sin(theta)
print(x, y)
```

► ข้อที่ 7

```
import math
x,y = [float(e) for e in input().split()]
r = math.sqrt(x**2+y**2)
theta = math.atan2(y,x)
print(r,theta)
```

► ข้อที่ 8

```
a,b,c,d,g = \
    [float(e) for e in input().split()]
avg = (a+b+c+d+g)/5
print(avg)
```

► ข้อที่ 9

```
a,b,c = [e for e in input().split()]
c = int(c)
out = a + b + str(c) + (a + b) * c
print(out)
```

unth 2

► ข้อที่ 1

```
# ex. 3 4 1 --> median is 3
a,b,c = [int(e) for e in input().split()]
if b <= a <= c or c <= a <= b :
    print(a)
elif a <= b <= c or c <= b <= a :
    print(b)
else :
    print(c)
```

► ข้อที่ 2

```
x1,y1,r1 = [float(e) \
            for e in input().split()]
x2,y2,r2 = [float(e) \
            for e in input().split()]
d = (x1-x2)**2 + (y1-y2)**2
sumr2 = (r1+r2)**2
if d < sumr2 :
    print('overlap')
elif d == sumr2 :
    print('touch')
else :
    print('free')
```

► ข้อที่ 3

```
x,y = [float(e) \
        for e in input().split()]
if x == 0 and y == 0 :
    print('At the origin')
elif x == 0 :
    print('On y-axis')
elif y == 0 :
    print('On x-axis')
elif x > 0 and y > 0 :
    print('Quadrant I')
elif x < 0 and y > 0 :
    print('Quadrant II')
elif x < 0 and y < 0 :
    print('Quadrant III')
else :
    print('Quadrant IV')
```

► ข้อที่ 4

```
a,b,c,d,e = [int(e) \
              for e in input().split()]
if a <= b <= c <= d <= e :
    print('True')
else :
    print('False')

# print( a <= b <= c <= d <= e )
```

► ข้อที่ 5

```
a,b,c,d = [int(e) \
           for e in input().split()]
mx = a
if b > mx : mx = b
if c > mx : mx = c
if d > mx : mx = d
mn = a
if b < mn : mn = b
if c < mn : mn = c
if d < mn : mn = d
s = (a+b+c+d) - mx - mn
print(s)

# s = (a+b+c+d)-max(a,b,c,d)-min(a,b,c,d)
```

► ข้อที่ 6

```
a = int(input())
x = int(round(a**(1/3),0))
if x**3 == a :
    print(x)
else :
    print('Not Found')
```

► ข้อที่ 7

```
c = int(input())
if c < 37 :
    s = 'XS'
elif c < 41 :
    s = 'S'
elif c < 43 :
    s = 'M'
elif c < 46 :
    s = 'L'
else :
    s = 'XL'
print(s)
```

บทที่ 3

► ข้อที่ 1

```
k = 1
while 1/k*k == 1 :
    k += 1
print(k)
```

► ข้อที่ 2

```
k = 1
p = 1.0
while (1-p) < 0.5 :
    p *= (365-k)/365
    k += 1
print(k)
```

► ข้อที่ 3

```
p = 0.0
for k in range(1,400000,4):
    p += 1/k
    p -= 1/(k+2)
print(4*p)
```

► ข้อที่ 4

```
a,b = [int(e) for e in input().split()]
s = 0
for i in range(a,b):
    t = 0
    for j in range(i+1,b+1):
        t += (i+j)
    s += (-1)**i * t
print(s)
```

► ข้อที่ 5

```
a,b = [int(e) for e in input().split()]
s = 0
for i in range(a,b):
    for j in range(i+1,b+1):
        s += (-1)**i * (i+j)
print(s)
```

► ข้อที่ 6

```
n = int(input())
a = int(input())
mn = mx = a
c = 0
if a < 0 : c = 1
for k in range(n-1):
    a = int(input())
    if a > mx : mx = a
    if a < mn : mn = a
    if a < 0 : c += 1
print( (mx - mn), c )
```

► ข้อที่ 7

```
n = int(input())
for x in range(1,n+1):
    for y in range(x,n+1):
        for z in range(y,n+1):
            t = x**2+y**2+z**2
            w = int(round(t**(1/3),0))
            if w**3 == t:
                print(w,x,y,z)
```

unที่ 4

► ข้อที่ 1

```
s = input().strip()
t = ''
for e in s:
    t += e*2
print(t)
```

► ข้อที่ 2

```
s = input().strip()
t = ''
s = ' ' + s + ' '
for i in range(1,len(s)-1) :
    t += s[i]
    if s[i-1] != s[i] != s[i+1] :
        t += s[i]
print(t)
```

► ข้อที่ 3

```
s = input().strip()
if s == s[::-1] :
    print('Y')
else:
    print('N')
```

► ข้อที่ 4

```
d,n = [int(e) for e in input().split()]
t = "0"*n + str(d)
t = t[-max(n,len(str(d))):]
print(t)
```

► ข้อที่ 5

```
h = input().strip()
d = '0123456789ABCDEF'.find(h)
if d >= 0 :
    print(d)
else :
    print('invalid hex digit')
```

► ข้อที่ 6

```
t = input().strip()
c = 0
for e in t :
    if e in '13579' :
        c += 1
print(c)
```

► ข้อที่ 7

```
t = input().strip()
c = 0
for k in range(len(t)-1) :
    if t[k] in 'aeiou' and \
       t[k+1] in 'aeiou' :
        c += 1
print(c)
```

► ข้อที่ 8

```
b = input().strip()
b = b[::-1]
d = 0
for i in range(len(b)) :
    d += int(b[i])*2**i
print(d)
```

unที่ 5

► ข้อที่ 1

```
file1 = open( input().strip() )
s = ''
for line in file1 :
    if line[-1] != '\n' : # กรณีบรรทัดสุดท้ายไม่มี \n
        line = line + '\n'
    s = line + s # นำบรรทัดใหม่มาต่อทางซ้าย
file1.close()
print(s[::-1]) # ลบ \n ที่บรรทัดสุดท้าย
```

► ข้อที่ 2

```
file1 = open( input().strip() )
s = ''
for line in file1 :
    if len(line.strip()) > 0 :
        if line[-1] != '\n' :
            line = line + '\n'
        s = line + s
file1.close()
out = open('reverse.txt', 'w')
out.write(s[::-1])
out.close()
```

► ข้อที่ 3

```
file1 = open( input().strip() )
for line in file1 :
    a = line.find("<headline>")
    if a >= 0 :
        j = a+len("<headline>")
        b = line.find("</headline>",j)
        print( line[j:b] )
file1.close()
```

► ข้อที่ 4

```
file1 = open( input().strip() )
file2 = open( input().strip() )
for line1 in file1 :
    line2 = file2.readline()
    if line1 != line2 :
        print( False )
        break
else :
    print( len(file2.readline()) == 0 )
file1.close()
file2.close()
```

หรือแบบสั้น ๆ ใช้ readlines

```
file1 = open( input().strip() )
file2 = open( input().strip() )
print( file1.readlines() == \
        file2.readlines() )
```

บทที่ 6.1

► ข้อที่ 1

```
v1 = [float(e) for e in input().split()]
v2 = [float(e) for e in input().split()]
if len(v1) != len(v2) :
    print( 'Error' )
else :
    dotp = 0
    for k in range(len(v1)) :
        dotp += v1[k]*v2[k]
    print(dotp)
```

► ข้อที่ 2

```
n = int(input())
d = []
for k in range(n):
    d.append( int(input()) )
d.sort()
t = []
for e in d :
    t.append(str(e))
print(','.join(t))
```

► ข้อที่ 3

```
file1 = open( input().strip() )
d = []
for line in file1:
    d.append( int(line) )
file1.close()
c = []
for e in d:
    c.append( d.count(e) )
maxc = max(c)
out = []
for k in range(len(d)):
    if c[k]==maxc and d[k] not in out :
        out.append(d[k])
for e in out :
    print(e)
```

► ข้อที่ 4

```
file1 = open( input().strip() )
h = []
for line in file1 :
    a = line.find("<headline>")
    if a >= 0 :
        j = a+len("<headline>")
        b = line.find("</headline>",j)
        h.append( line[j:b] )
file1.close()
h.sort()
for e in h :
    print(e)
```

► ข้อที่ 5

```
file1 = open( input().strip() )
d = []
for line in file1 :
    d.append( line.strip() )
file1.close()

for k in range(len(d)-1) :
    for i in range(len(d)-1) :
        if len(d[i]) > len(d[i+1]) or \
           len(d[i]) == len(d[i+1]) and \
           d[i] > d[i+1] :
            d[i],d[i+1] = d[i+1],d[i]

for e in d :
    print(e)
```

บทที่ 6.2

► ข้อที่ 1

```
r,c = [int(e) for e in \
      input().split()]
m = []
for k in range(r) :
    m.append( [int(e) for e in \
              input().split()] )
    if len(m[k]) != c :
        m = [[]]
        break
print( m )
```

► ข้อที่ 2

```
infile = open(input().strip())
f = []
for line in infile :
    usernames = line.split()
    f.append( [usernames[0], \
              usernames[1:]] )
infile.close()
print( f )
```

► ข้อที่ 3

```
# สมมติว่ามี f มาแล้ว
nofollowers = []
for [username,followers] in f :
    if len(followers) == 0 :
        nofollowers.append( username )
print( 'No followers :', \
      ','.join(nofollowers) )
```

► ข้อที่ 4

```
n = int(input())
d = []
for k in range(n)
    s = input().strip()
    d.append( [len(s),s] )
d.sort()
for [c,s] in d :
    print(s)
```

บทที่ 6.3

► ข้อที่ 1

```
d = [ e.count(c) for e in x ]
```

► ข้อที่ 2

แบบแรก

```
x = [e for e in x if e >= 0]
```

คือการสร้างลิสต์ใหม่ที่ไม่มีเลขลบ แล้วให้ x อ้างอิงลิสต์ใหม่นี้ ถ้าเขียนอีกแบบ

```
x[:] = [e for e in x if e >= 0]
```

คือการสร้างลิสต์ใหม่ที่ไม่มีเลขลบ แล้วนำค่าในลิสต์ใหม่นี้ไปใส่ในที่เก็บเดิมของลิสต์ x สองวิธีนี้ให้ผลคล้ายกัน ต่างกันตรงที่ ถ้าก่อนหน้านี้มีการทำคำสั่ง `y = x` คือให้ y อ้างอิงลิสต์เดียวกับ x การทำแบบแรกจะทำให้ y อ้างอิงลิสต์เดิม และ x อ้างอิงลิสต์ใหม่ ในขณะที่แบบหลัง y ยังคงอ้างอิงลิสต์เดียวกับ x

► ข้อที่ 3

```
# t = [ [1,2,3], [33], [3,3,3,4] ]
t = sum( [ sum(e) for e in x ] )
```

► ข้อที่ 4

```
c = sum( [1 for e in input().split() \
          if int(e) < 0] )
```

► ข้อที่ 5

```
t = ''.join([e for e in input() \
            if 'a' <= e.lower() <= 'z'])
```

► ข้อที่ 6

```
x = [float(e) for e in input().split()]
y = [float(e) for e in input().split()]
z = [x[i]+y[i] for i in range(len(x))]
```

► ข้อที่ 7

```
f = [e for row in m for e in row]
```

► ข้อที่ 8

```
x = [int(e) for e in input().split()]
x.sort()
t = [x[i] for i in range(len(x)-1) \
     if x[i] != x[i+1] ]
t.append(x[-1])
```

เขียนแบบนี้ก็ได้ (แต่ช้ากว่าเยอะ)

```
x = [int(e) for e in input().split()]
t = []
for e in x :
    if e not in t :
        t.append(e)
```


► ข้อที่ 9

```

n = int(input())
x = [j for i in range(2, n//2) \
      for j in range(2*i, n, i)]
x.sort()
c = [x[i] for i in range(len(x)-1) \
      if x[i] != x[i+1] ]
c.append(x[-1])

```

► ข้อที่ 10

```

n = int(input())
x = [j for i in range(2, n//2) \
      for j in range(2*i, n, i)]
x.sort()
c = [x[i] for i in range(len(x)-1) \
      if x[i] != x[i+1] ]
c.append(x[-1])
p = [e for e in range(2,n) if e not in c]

```

unที่ 7

► ข้อที่ 1

- 1) dict มี key คือ studentID,
value คือ grade
- 2) set ของ studentID
- 3) dict มี key คือ dept,
value คือ set ของ studentID
- 4) list ของ phone
- 5) dict มี key คือ เลขท้าย TelNo,
value คือ count

► ข้อที่ 2

```

x = int(input())
t = ()
for i in range(0,x,2):
    # คำสั่งข้างล่างนี้ไม่ได้เปลี่ยน tuple ของ t แต่สร้าง
    # tuple ใหม่ แล้วแทนที่ t ตัวเก่า
    t += (i,) # เหมือน t = t + (i,)
print(t)
หรือ
t = tuple([e for e in range(0,x,2)])
หรือ
t = tuple(range(0,x,2))

```

► ข้อที่ 3

```

x = int(input())
t = ()
while x > 0:
    t = (x%10,) + t
    x //= 10
print(t)
หรือ เปลี่ยนมารับสตริง, นำแต่ละหลักมาเปลี่ยนเป็น int,
เก็บใส่ list, แล้วส่งไปสร้างเป็น tuple
x = input().strip()
print(tuple([int(e) for e in x]))

```

► ข้อที่ 4

```

x = input().strip()
d = {}
for e in x:
    if e not in d: d[e] = 1
    else: d[e] += 1
print(d)

```

► ข้อที่ 5

```

x = input().strip()
y = input().strip()
set_x = set(x)
set_y = set(y)
print(set_x.intersection(set_y))

```

unที่ 8

► ข้อที่ 1

```

def f1(a,b):
    for i in range(b): print(a)

```

► ข้อที่ 2

```

def f2(a,b):
    return [a]*b

```

► ข้อที่ 3

```

def g(m,b,n,c):
    if m==n and b!=c: return 1
    if m==n and b==c: return 2
    x = (c-b)/(m-n)
    y = m*x + b
    return (x,y)

```

► ข้อที่ 4

```

def h(x):
    return [e for e in x if e%2==0]

```

► ข้อที่ 5

```

def a(n):
    if n==0: return 1
    if n==1: return -2
    return a(n-2)*n

```

► ข้อที่ 6

```
def k(n):
    if n==0: return 1
    if n==1: return 2
    if n%2==0:
        x = k(n//2)
        return x + x%10
    return k(n//2-1)*(n//2)
```

► ข้อที่ 7

```
def s(i,k):
    if i>=k: return 0
    return k + t(i+1,k)
def t(j,k):
    if j>=k: return 0
    return j + s(j,k-1)
```

► ข้อที่ 8

```
def is_palindrome(s):
    if len(s) <= 1: return True
    if s[0]!=s[-1]: return False
    return is_palindrome(s[1:-1])
```

บทที่ 9

► ข้อที่ 1

```
k = int(input())
M[:, :k, :k] = 0
```

► ข้อที่ 2

แบบช้า

```
k = int(input())
for i in range(M.shape[0]):
    for j in range(M.shape[1]):
        if i%k==0 and j%k==0:
            M[i][j]*=2
```

แบบนี้ก็ช้าอยู่

```
k = int(input())
M = np.array([[2*M[i][j] if i%k==0 \
               and j%k==0 else M[i][j] \
               for j in range(M.shape[1])] \
               for i in range(M.shape[0])])
```

แบบนี้เร็ว

```
k = int(input())
N = np.zeros_like(M)
N[:, :k, :k] = 1
M += M*N
```

แบบนี้เร็วและสั้น

```
k = int(input())
M[:, :k, :k] *= 2
```

► ข้อที่ 3

```
MAX = np.max(M,axis=0)
MIN = np.min(M,axis=0)
A = MAX-MIN
```

► ข้อที่ 4

```
Y = (X[:,0]**2+X[:,1]**2)**0.5
```

► ข้อที่ 5

```
k = int(input())
C = np.zeros((k,k),int)
C[:,2, 1::2] = C[1::2, ::2] = 1
```

► ข้อที่ 6

```
k = int(input())
C = np.zeros((k,k),int)
C[:,2, ::2] = C[1::2, 1::2] = 1
C = (C*np.arange(1,k+1)).T
# ใช้ C*(np.arange(1,k+1).T) ไม่ได้(เพราะอะไร?)
```

บทที่ 10

► ข้อที่ 1

```
def __init__(self, name, year, faculty):
    self.name = name
    self.year = year
    self.faculty = faculty
```

```
def __str__(self):
    return self.name + \
           ' (year ' + str(self.year) + ') ' + \
           self.faculty
```

```
def __lt__(self, rhs):
    if self.faculty != rhs.faculty:
        return self.faculty < rhs.faculty
    if self.year != rhs.year:
        return self.year < rhs.year
    return self.name < rhs.name
```

► ข้อที่ 2

```
def __init__(self, license, brand, color):
    self.license = license
    self.brand = brand
    self.color = color
    self.report = []

def __str__(self):
    return self.license + ' - ' \
        + self.color + ' ' + self.brand

def __lt__(self, rhs):
    return self.license < rhs.license

def add_report(self, new_report):
    self.report.append(new_report)

def total_payment(self):
    return sum([r[2] for r in self.report])

def max_payment(self):
    if self.report == []: return []
    max_p = max([r[2] for r in self.report])
    return [r for r in self.report \
            if r[2] == max_p]
```

► ข้อที่ 3

```
def add_book(self, book, n):
    for b in self.books:
        if b[0] == book:
            b[1] += n
            break
    else:
        self.books.append([book,n])

def delete_book(self, book):
    self.books = [[b,n] for [b,n] \
        in self.books if b != book]

หรือ
for k in range(len(self.books)):
    if self.books[k][0] == book:
        self.books.pop(k)
        break

def get_total(self):
    return sum([b.price*n for \
        [b,n] in self.books])

def __lt__(self, rhs):
    return self.get_total() < \
        rhs.get_total()
```

► ข้อที่ 4

```
def add_value(self, x):
    self.value += x

def enter(self, station):
    if self.station == '':
        self.station = station
        return True
    else:
        return False

def leave(self, station):
    if self.station == '':
        return (self.value, -2)
    price = Station.get_price \
        (self.station, station)
    if price > self.value:
        return (self.value, -1)
    else:
        self.value -= price
        self.station = ''
        return (self.value, 0)

def __lt__(self, rhs):
    return self.value < rhs.value
```

เว็บไซต์ที่เป็นประโยชน์

- youtu.be/U2l1xgpVsuo?list=PL0ROnCzUGB4ieaQndKybT9xyoq2n9NGq (ภาพยนตร์บรรยายเนื้อหาวิชา)
- en.wikibooks.org/wiki/Python_Programming (เว็บไซต์อ้างอิงสำหรับการเขียนโปรแกรมภาษา Python)
- www.python.org (เว็บไซต์ทางการของภาษา Python)
- docs.python.org/3/tutorial/index.html, docs.python.org/3/library/index.html, docs.python.org/3/reference/index.html (เว็บไซต์ทางการของภาษา Python ส่วนเอกสารอ้างอิง)
- www.numpy.org (เว็บไซต์ทางการของคลังคำสั่ง NumPy)
- www.lfd.uci.edu/~gohlke/pythonlibs (เว็บไซต์สำหรับดาวน์โหลดคลังคำสั่งภาษา Python)
- www.pythontutor.com (เว็บไซต์สำหรับการ visualize โปรแกรมภาษา Python)
- repl.it/languages/python3 (เว็บไซต์สำหรับเขียนโปรแกรมภาษา Python ออนไลน์ สามารถใช้ NumPy ได้)
- openbookproject.net/thinkcs/python/english3e (เว็บไซต์อ้างอิงสำหรับการเขียนโปรแกรมภาษา Python)

Error ที่สามารถพบได้

แบบขึ้นเป็นกล่องข้อความ

ข้อความ error	สาเหตุ	แนวทางการแก้ไข
unexpected EOF while parsing	ใส่วงเล็บไม่ครบ เช่น <code>x = (1+(2**3)</code>	ตรวจสอบวงเล็บให้ครบ
EOL while scanning string literal	ใส่อักขระประกาศจบสตริงไม่ครบ เช่น <code>s = 'hello</code>	เติมอักขระประกาศปิดสตริงให้ครบ
invalid syntax	มีการเขียนโปรแกรมผิดกฎของภาษา เช่น ลืม : หลัง if หรือ while หรือเรียง else มาก่อน if หรือเปรียบเทียบด้วย = ตัวเดียว เป็น if x = 1 : หรืออาจจะมีอักขระพิเศษซ่อนอยู่ ซึ่งมาจากการ copy-paste คำสั่งจากไฟล์ pdf	ตรวจสอบการเขียนโปรแกรมให้เป็นไปตามกฎของภาษา
unexpected indent	มีการเยื้องที่ไม่ตรงกัน เช่น <code>if a>0: a+=1 a+=2</code>	จัดการเยื้องให้ตรงกัน

หมายเหตุ ควรตรวจสอบข้อผิดพลาดทั้งบรรทัดที่เกิด error และบรรทัดก่อนหน้าบรรทัดที่เกิด error

แบบขึ้นเป็นตัวอักษรสีแดงใน shell

ข้อความ error	สาเหตุ	แนวทางการแก้ไข
name 'x' is not defined	ไม่ได้กำหนดค่าตัวแปร x ก่อนใช้งาน เช่น <code>print(x+1)</code> หรือพิมพ์ชื่อฟังก์ชันผิด เช่น <code>printt('cat')</code> หรือลืม <code>import</code>	กำหนดค่าตัวแปรก่อนใช้งานให้ เรียบร้อย ตรวจสอบการพิมพ์ชื่อฟังก์ชัน ทำการ <code>import</code> ให้เรียบร้อย
ZeroDivisionError: division by zero	มีการหารด้วย 0 เช่น <code>print(5/0)</code>	แก้ไขตัวหารให้ไม่เป็น 0
ImportError: No module named 'maht'	พิมพ์ชื่อ module ที่ต้องการ import ผิด เช่น <code>import maht</code>	แก้ไขชื่อ module ที่ต้องการ import ให้ถูกต้อง
AttributeError: module 'math' has no attribute 'arcsin'	module ที่เรียก ไม่มีฟังก์ชันที่ ต้องการ เช่น <code>print(math.arcsin(0))</code>	แก้ไขชื่อฟังก์ชันให้ถูกต้อง เช่นในที่นี้ต้องใช้ <code>math.asin(0)</code>
ValueError: math domain error	มีการใส่ค่าที่ฟังก์ชันไม่รองรับ เช่น <code>print(math.asin(1000))</code> <code>print(math.log(10,0))</code>	ตรวจสอบค่าที่ฟังก์ชันนั้นรองรับ และใส่ค่าที่ถูกต้อง
invalid literal for int() with base 10	มีการใส่ค่าที่แปลงไม่ได้ลงไปใน <code>int()</code> เช่น <code>x = int('cat')</code>	ตรวจสอบค่าที่ต้องการแปลง ว่าเป็นจำนวนเต็มหรือไม่ และแก้ไขให้ถูกต้อง
could not convert string to float	มีการใส่ค่าที่แปลงไม่ได้ลงไปใน <code>float()</code> เช่น <code>x = float('cat')</code>	ตรวจสอบค่าที่ต้องการแปลง ว่าเป็นจำนวนทศนิยมหรือไม่ และแก้ไขให้ถูกต้อง
Can't convert 'int' object to str implicitly	มีการบวก <code>int</code> กับ <code>string</code> เช่น <code>print('cat'+3)</code>	แก้ไขให้ถูกต้อง อาจใช้ <code>comma</code> แทนเครื่องหมาย + หรือเพิ่มอัญประกาศเป็น <code>'3'</code>
<code>sqrt()</code> takes exactly one argument (3 given)	ฟังก์ชัน <code>sqrt()</code> ต้องใส่ข้อมูล 1 ค่า แต่ตอนนี้ใส่ไป 3 ค่า เช่น <code>x = sqrt(4,9,16)</code>	เปลี่ยนให้ใส่ข้อมูลตามจำนวนที่ฟังก์ชัน กำหนด
ValueError: too many values to unpack	ค่าที่มีให้ มากกว่าจำนวนตัวแปร เช่น <code>a,b,c = 1,2,3,4,5</code>	แก้ไขจำนวนทั้งสองฝั่งให้เท่ากัน
ValueError: not enough values to unpack	ค่าที่มีให้ น้อยกว่าจำนวนตัวแปร เช่น <code>a,b,c = 1,2</code>	แก้ไขจำนวนทั้งสองฝั่งให้เท่ากัน
TypeError: unorderable types: int() < str()	มีการเปรียบเทียบจำนวนกับสตริง เช่น <code>if 9 < '9':</code>	แก้ไขประเภทข้อมูลให้ตรงกัน
'float' object cannot be interpreted as an integer	มีการใส่ค่า <code>float</code> ในส่วนที่ควรจะเป็น ค่า <code>int</code> เช่น <code>for i in range(2.5):</code>	แก้ไขค่าให้เป็นจำนวนเต็ม

ข้อความ error	สาเหตุ	แนวทางการแก้ไข
'int' object is not iterable	มีการใช้คำสั่ง in กับสิ่งที่ไล่ลำดับไม่ได้ (int หรือ float) เช่น for i in 5:	ใช้ in กับสิ่งที่ไล่ลำดับได้ เช่น range, string, list, tuple, set, dict
index out of range	มีการเรียก index เกินจากที่มี เช่น x = 'cat'; print(x[100]) y = [1,2,3]; print(y[100])	ตรวจสอบว่า index ที่ต้องการเรียก ไม่เกินความยาวที่มี
indices must be integers	มีการเรียก index ของสตริงหรือลิสต์ ด้วยค่าที่ไม่เป็นจำนวนเต็ม เช่น x = 'cat'; print(x[2/1]) y = [1,2,3]; print(y['a'])	ตรวจสอบว่า index ที่เรียก เป็นจำนวนเต็มหรือไม่ และแก้ไขให้ถูกต้อง
FileNotFoundError: No such file or directory	ต้องการเปิดไฟล์ แต่ไม่เจอไฟล์นั้น	ตรวจสอบว่ามีไฟล์อยู่จริง และพิมพ์ชื่อไฟล์ถูกต้อง
OSError: Invalid argument: 'D:\x0cile.txt'	พิมพ์ชื่อไฟล์ด้วยเครื่องหมาย \ เช่น f = open('D:\file.txt')	พิมพ์ชื่อไฟล์ด้วย \\ แทน เป็น f = open('D:\\file.txt')
KeyError	dict ไม่มี key ที่ต้องการ เช่น d = {}; print(d[1]) หรือ set ไม่มีสมาชิกที่ต้องการ เช่น s = {1,2,3}; s.remove(4)	ดูว่าใช้ key ถูกต้องหรือไม่ เพิ่ม code เพื่อตรวจสอบว่ามี key อยู่ใน dict หรือไม่
'set' object has no attribute 'delete' (อาจเกิด error กับ list, string, tuple และ dict ได้ด้วย)	เรียกใช้บริการที่ไม่มีกำหนดไว้ เช่น s = {1,2,3}; s.delete(1) ซึ่ง set ไม่มีบริการที่ชื่อว่า delete	แก้ไขการใช้คำสั่งให้ถูกต้อง
TypeError: unhashable type	มีการใช้ list, set, dict เป็น key ของ dict หรือเก็บ list, set, dict ใน set เช่น d = {}; d[{1}] = 'cat'	ตรวจสอบประเภทข้อมูลที่นำมาใช้ว่าไม่ใช่ list, set, dict
unsupported operand type(s) for +=: 'NoneType' and 'int'	มีการนำ None ไปประมวลผล เช่น def func(x): x += 1 print(func(3)+1) ซึ่งอาจเกิดจากการลืม return	return ค่าให้ถูกต้อง
'int' object is not callable	มีการเรียกฟังก์ชันด้วยตัวแปร เช่น a = 3 b = a(5) อาจเกิดจากการตั้งชื่อฟังก์ชันซ้ำกับตัวแปร	ตรวจสอบการตั้งชื่อฟังก์ชันและตัวแปร และแก้ไขให้ถูกต้อง
RecursionError: maximum recursion depth exceeded	มีการเรียกฟังก์ชันตัวเองซ้ำมากเกินไป อาจเกิดจากการลืมกรณีพื้นฐานของการ recursive เช่น def f(n): return n*f(n-1)	เติมกรณีพื้นฐานของการ recursive และเช็คเงื่อนไขการหยุดทำงานให้ถูกต้อง

ศุภเสฏฐ์ ชุชัยศรี

Intania 86

Solutions Engineer, Facebook
และ นายกสมาคมโปรแกรมเมอร์ไทย



ผมคิดว่าการเรียนเขียนโปรแกรมจะเป็นเหมือนการเรียนรู้ภาษาที่ 3 ในอนาคตอันใกล้ ยิ่งเทคโนโลยีแทรกซึมไปในทุกสิ่งเร็วเท่าไร ความต้องการคนที่เข้าใจและควบคุมเทคโนโลยีได้ก็จะมีมากขึ้นเท่านั้น นอกจากนั้นการเรียนเขียนโปรแกรมายังช่วยให้เรารู้จักคิดแก้ปัญหาอย่างเป็นระบบ และให้พลังในการสร้างสรรค์สิ่งต่าง ๆ ได้อย่างไม่รู้จบ

ณัฐยา ลีละศุภกุล

Intania 95

ตลาดหลักทรัพย์แห่งประเทศไทย



ถ้าเทียบการสร้าง software กับการสร้างตึกตึกหนึ่ง **programming** ก็เหมือนเป็นเสาเข็ม ที่รองรับแรงของทั้งตึกให้ตั้งอยู่ได้ และไม่พังทลาย ตราบใดที่ไม่มีเสาเข็ม ก็ไม่สามารถสร้างตึกได้ ดังนั้น **programming** ก็เป็นเหมือนรากฐานที่สำคัญของ software ถ้าเรามีทักษะ **programming** ที่ดี เราก็จะสามารถพัฒนา software ที่มีคุณภาพได้เช่นกัน

คำสั่งพื้นฐาน	import math
<p><code>x = int(input())</code> รับข้อมูลจำนวนเต็ม 1 จำนวน</p> <p><code>x,y,z = [float(e) for e in input().split()]</code> รับข้อมูลจำนวนจริง 3 จำนวนในบรรทัดเดียวกัน คั่นด้วยช่องว่าง</p> <p><code>int(x), float(x), str(x):</code> คืนค่า x ที่ถูกเปลี่ยนประเภทข้อมูลเป็นจำนวนเต็ม จำนวนจริง และสตริง</p> <p><code>abs(n):</code> คืนค่าสัมบูรณ์ของ n</p> <p><code>round(f):</code> คืนค่าจำนวนเต็มที่เกิดจากการปัดเศษจำนวนจริง f โดยถ้าเศษของ f มีค่าตั้งแต่ 0.5 จะปัดขึ้น ถ้าน้อยกว่า 0.5 จะปัดลง</p> <p><code>round(f,d):</code> คืนค่าจำนวนจริงที่เกิดจากการปัดเศษจำนวนจริง f โดยปัดให้มีจำนวนตัวเลขหลังจุดทศนิยม d หลัก</p> <p><code>range(start,stop [,step])</code> หรือ <code>range(stop):</code> คืนค่าเป็นลิสต์ของตัวเลขตามลำดับตั้งแต่ start ถึง stop-step และ เพิ่มขึ้นทีละ step (ถ้าไม่ระบุ start จะมีค่า 0 และ step จะมีค่า 1)</p> <p><code>enumerate(L):</code> คืนลิสต์ของ tuple (index, element) ของแต่ละ ข้อมูลในลิสต์ L</p> <p><code>len(a):</code> คืนค่าเป็นจำนวนข้อมูลใน a ซึ่ง a อาจเป็นลิสต์ ดิกชันนารี เซต ทูเปิล สตริง หรือ numpy array ก็ได้</p> <p><code>max(a), min(a):</code> คืนค่าที่มาก/น้อยที่สุดของข้อมูลใน a ซึ่ง a อาจเป็น ลิสต์ ดิกชันนารี เซต ทูเปิล หรือสตริงก็ได้ (numpy array ใช้ <code>np.max(a), np.min(a)</code>) ถ้า a เป็นดิกชันนารี จะคืนค่ามาก/น้อยที่สุดของ key ของดิกชันนารี</p> <p><code>type(a):</code> คืนค่าประเภทของ a เช่น <code>type([1,2])</code> ได้ <code><class 'list'></code></p> <p><code>list(), dict(), tuple(), set():</code> สร้างลิสต์ว่าง ดิกชันนารีว่าง ทูเปิลว่าง เซตว่าง</p>	<p><code>math.exp(x):</code> คืนค่า e ยกกำลัง x</p> <p><code>math.cos(x):</code> คืนค่า cosine ของ x เรเดียน</p> <p><code>math.sin(x):</code> คืนค่า sine ของ x เรเดียน</p> <p><code>math.sqrt(x):</code> คืนค่ารากที่สองของ x</p> <p><code>math.log(x,base):</code> คืนค่าลอการิทึมของ x ฐาน base</p> <p><code>math.degrees(x):</code> แปลงมุม x จากเรเดียนเป็นองศา</p> <p><code>math.radians(x):</code> แปลงมุม x จากองศาเป็นเรเดียน</p> <p><code>math.pi, math.e:</code> ค่าคงที่ pi และ e</p>
	string s
	<p><code>s.lower():</code> คืนสตริงใหม่ที่มีค่าเหมือน s แต่เป็นตัวพิมพ์เล็กทั้งหมด</p> <p><code>s.upper():</code> คืนสตริงใหม่ที่มีค่าเหมือน s แต่เป็นตัวพิมพ์ใหญ่ทั้งหมด</p> <p><code>s.find(sub):</code> คืน index แรกสุดที่พบ sub ใน s ถ้าไม่พบคืนค่า -1</p> <p><code>s.find(sub,i):</code> คืน index แรกสุดที่พบ sub ใน s โดยเริ่มค้นที่ index i</p> <p><code>s.count(sub):</code> คืนจำนวนครั้งที่ sub ปรากฏในสตริง s</p> <p><code>s.split(sep):</code> คืนลิสต์ของสตริงที่แยกด้วย sep (หรือ space ถ้าไม่ระบุ)</p> <p><code>s.strip():</code> คืนสตริงใหม่ที่มีค่าเหมือน s แต่ตัด spaces หัวท้ายออก</p> <p><code>s.join(L):</code> คืนสตริงที่สร้างจากการนำแต่ละ element ในลิสต์ L มาต่อกัน โดยมี s เป็นตัวคั่นระหว่างข้อมูลที่ต่อกัน (L ต้องเป็นลิสต์ของสตริง)</p>
	import numpy as np
	<p><code>np.array(L):</code> คืนค่า numpy array ที่สร้างจากลิสต์ L</p> <p><code>np.arange(start,stop,step):</code> คืนอาร์เรย์ 1 มิติของจำนวนที่มีค่าตาม start,stop,step</p> <p><code>np.ones(shape):</code> คืนอาร์เรย์ที่มีค่า 1 ทั้งหมด มีขนาดตาม tuple shape</p> <p><code>np.zeros(shape):</code> คืนอาร์เรย์ที่มีค่า 0 ทั้งหมด มีขนาดตาม tuple shape</p> <p><code>np.identity(size):</code> คืนอาร์เรย์ขนาด size x size ซึ่งมีข้อมูลในแนว เส้นทแยงมุมเป็น 1 และค่าในตำแหน่งอื่น ๆ เป็น 0</p> <p><code>np.empty_like(a):</code> คืนอาร์เรย์ใหม่ที่มีขนาดเหมือน a แต่ไม่มีการกำหนด ค่าข้อมูลในอาร์เรย์ใหม่นี้</p> <p><code>np.add(a,b), np.subtract(a,b), np.multiply(a,b), np.divide(a,b):</code> คืนค่าอาร์เรย์ใหม่ที่เป็นผลบวกลบคูณหารแบบช่องต่อช่องของ a และ b</p> <p><code>np.dot(a,b):</code> คืนอาร์เรย์ที่เป็นผลคูณแบบเมทริกซ์ของ a และ b</p> <p><code>np.sin(a), np.cos(a), np.sqrt(a), np.abs(a):</code> คืนอาร์เรย์ที่มีค่าของข้อมูลในแต่ละตำแหน่งเป็นผลจากการเรียกฟังก์ชัน sine, cosine, sqrt, abs กับข้อมูลในอาร์เรย์ a ที่ตำแหน่งเดียวกัน</p> <p><code>np.max(a,axis), np.min(a,axis):</code> คืนอาร์เรย์ของค่ามาก/น้อยที่สุดใน a ตาม axis ที่กำหนด</p> <p><code>np.argmax(a,axis), np.argmin(a,axis):</code> คืนอาร์เรย์ของ index ที่มีค่ามาก/น้อยที่สุดใน a ตาม axis ที่กำหนด ตัวอย่างเช่น <pre>a = np.array([[2, 4, 6], [8, 10, 12]]) np.max(a) คืนค่า 12, np.argmax(a) คืนค่า 5 np.max(a,axis=0) คืนค่า array([8,10,12]) np.argmax(a,axis=0) คืนค่า array([1,1,1]) np.argmax(a,axis=1) คืนค่า array([2,2])</pre> </p> <p><code>np.sum(), np.std(), np.mean():</code> มีการใช้งานเหมือน <code>np.max()</code></p> <p><code>np.ndenumerate(a):</code> คืนลิสต์ของ tuple (position,element) ของ แต่ละข้อมูลใน a โดย position เป็น tuple ที่เก็บตำแหน่งของข้อมูล</p>
list L	
<p><code>L.append(e):</code> เพิ่ม e ไปที่ท้ายลิสต์ L</p> <p><code>L.insert(index,e):</code> เพิ่ม e ไปที่ตำแหน่ง index ในลิสต์ L</p> <p><code>L.pop(index):</code> ลบข้อมูลที่ตำแหน่ง index และคืนค่าข้อมูลที่ถูกลบ</p> <p><code>L.count(e):</code> คืนจำนวนครั้งที่ e ปรากฏในลิสต์ L</p>	
dict D	
<p><code>D.items():</code> คืนลิสต์ของ tuple (key, value) ของดิกชันนารี D</p> <p><code>D.keys():</code> คืนลิสต์ของ key ทั้งหมดของดิกชันนารี D</p> <p><code>D.values():</code> คืนลิสต์ของ value ทั้งหมดของดิกชันนารี D</p> <p><code>D.pop(k):</code> ลบข้อมูลใน D ที่มี key เป็น k และคืนค่า value ของ key นั้น</p> <p><code>D.update(D1):</code> เพิ่มข้อมูลจากดิกชันนารี D1 เข้าไปใน D</p>	
set S	
<p><code>S.add(e):</code> เพิ่ม e ในเซต S</p> <p><code>S.difference(T):</code> คืนเซตใหม่ที่เท่ากับ S-T</p> <p><code>S.discard(e):</code> ลบ e ออกจากเซต S ถ้าไม่มี e ใน S ก็ไม่ทำอะไร</p> <p><code>S.intersection(T):</code> คืนเซตใหม่ที่เท่ากับ S ∩ T</p> <p><code>S.union(T):</code> คืนเซตใหม่ที่เท่ากับ S ∪ T</p> <p><code>S.issubset(T):</code> ทดสอบว่า S ⊆ T หรือไม่</p> <p><code>S.issuperset(T):</code> ทดสอบว่า S ⊇ T หรือไม่</p> <p><code>S.pop():</code> ลบข้อมูลหนึ่งตัวออกจากเซต S และคืนข้อมูลที่ถูกลบ</p> <p><code>S.update(T):</code> ให้ S = S ∪ T</p>	

เปลี่ยนความอยากรู้อยากเห็น มาสู่งานอาชีพ

ในอนาคต ประเทศไทยจำเป็นต้องมีบุคลากรทางด้านวิศวกรรมและเทคโนโลยีเพิ่มขึ้นอีกมาก
เพื่อให้ทันต่อความต้องการ ด้วยเหตุนี้ เอสโซ่ และ บริษัทในเครือเอ็กซอนโมบิลในประเทศไทย
จึงให้ความสนับสนุนโครงการเรียนรู้ที่เกี่ยวข้องกับคณิตศาสตร์และวิทยาศาสตร์
เพื่อเตรียมความพร้อมทั้งด้านศักยภาพและทักษะให้เยาวชนรุ่นใหม่ สำหรับงานในศตวรรษที่ 21
เพราะการส่งเสริมสนับสนุนเยาวชนของเราวันนี้
จะส่งผลถึงความก้าวหน้าของประเทศต่อไปในวันข้างหน้า

Energy lives here™

ExxonMobil

Mobil  Mobil



Find job opportunities at jobs.exxonmobil.com



“
ทุกอย่างใน **python** จะดูซอฟต์
เมื่อเป็นสัฟาสเทล
”

- โปรแกรมเมอร์นิรนาม -

สนับสนุนการพิมพ์โดย

ExxonMobil

ISBN 978-616-407-189-6



9 786164 071896