



python ๑๐๑



ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์
จุฬาลงกรณ์มหาวิทยาลัย

CONFIDENTIAL

PYTHON ๑๐๑

สิงหาคม ๒๕๖๑

v1.0.2

ผู้อ่านสามารถดาวน์โหลดหนังสือรุ่นล่าสุด
และร่วมแสดงความคิดเห็น/ข้อเสนอแนะเกี่ยวกับหนังสือเล่มนี้ที่

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

ขอบคุณครับ

กิตติคุณ พละการ, กิตติภพ พละการ, สมชาย ประสิทธิ์จตุระกุล, สุกรี สิ้นธุญโญ

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

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

005.133

ISBN 978-616-407-1๐๑-๕

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 การเขียนโปรแกรมคอมพิวเตอร์ให้ตรงตามข้อกำหนดที่ได้รับ การเขียนโปรแกรมเป็นความสามารถที่ต้องลงมือฝึกฝนฝึกปฏิบัติด้วยตนเอง เหมือนกับทักษะอื่นทางวิศวกรรมที่จำเป็นต้องฝึก ๆ ๆ ให้นานาญจึงจะได้ผล ไม่สามารถได้มาด้วยการอ่าน ๆ ๆ 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 นิสิตคณะวิศวกรรมศาสตร์ กว่าหลายพันคนที่ขยันหมั่นศึกษาและฝ่าฟันอุปสรรคในการเรียนวิชาพื้นฐานบังคับที่ค่อนข้างไม่คุ้นเคยนี้จนสำเร็จ *

กิตติภณ พละการ

กิตติภาพ พละการ

สมชาย ประสิทธิ์จูตระกูล

(ผู้เรียบเรียงและรวบรวมเนื้อหา)

สุกรี สิ้นธุริญโย (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 เพ็ชรมาบุญ

อ. ดร. กรวิก ตันเกษร นนท์

อ. ดร. พรรณี แสงแก้ว

รศ. ดร. รัชชา ทวีแสงสกุลไทย

อ. ดร. เชษฐา พันธุ์เครือบุตร

อ. ดร. สุรัฐ ขวัญเมือง

ผศ. ดร. อนุรักษ์ ศรีอริยวัฒน์

ผศ. ดร. จิรวัดน์ ชีวรุ่งโรจน์

ผศ. ดร. ณัฐวุฒิ หนูไพโรจน์

ศิษย์เก่าภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

คุณวโรส โรจนะ

คุณธนวัฒน์ มาลาบุปผา

คุณลิสดา ตรงประกอบ

คุณภัทราวุธ ชื้อสัตยาศิลป์

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

CONFIDENTIAL

00 : Programming in Engineering

ศาสตร์ของวิศวกรรมไฟฟ้า เป็นศาสตร์ที่ครอบคลุมความหลากหลาย มีการเปลี่ยนแปลงและความหวัดอย่างมากในปัจจุบัน ในหลายโอกาส การสร้างต้นแบบเหมือนจริงอาจมีต้นทุนสูง 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 บัณฑิต เราต้องคำนวณและสรุปผลวิเคราะห์ ค่าพารามิเตอร์ทางสิ่งแวดล้อม ค่าเฉลี่ย ไม่ง่าย และไม่ซับซ้อนถึงขนาดต้องเรียน ภาคคอม ฯ การที่นิสิตได้เรียน **Programming** เบื้องต้น จะสามารถช่วยให้นิสิตสามารถ ต่อยอดความรู้ไปสู่งานวิศวกรรม สิ่งแวดล้อมในโลกอนาคตได้อย่างง่ายดาย

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. วรณิ แสงแก้ว
ภาควิชาวิศวกรรมนิวเคลียร์

บทบาทของวิศวกรรมอุตสาหการ คือ การออกแบบ ดำเนินการ ปรับปรุงและสร้างสรรค์นวัตกรรมระบบ ที่ ง่าย เลิศ บริการ และ ธุรกิจ

ทักษะ **Programming** เป็นปัจจัยที่กระตุ้นการคิดอย่างมีเหตุผล อีกทั้งยังเป็นเครื่องมือในการวิเคราะห์ และ ประมวลผลอย่างรวดเร็วและแม่นยำตามตัวแปรและข้อจำกัดในบริบทหนึ่ง 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 พันธุ์ศรีบุญต
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 การใช้ธรณีสัณติเพื่อประกอบการคำนวณปริมาณสำรอง เป็นต้น



ผศ. ดร. จิรวัดน์ ชวรุ่งโรจน์
ภาควิชาวิศวกรรมเหมืองแร่และปิโตรเลียม



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

ผศ. ดร. ณัฐวุฒิ หนูโพธิ์โรจน์
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 ที่เราจะศึกษาในบทนี้ดังต่อไปนี้

int	จำนวนเต็ม	-10 5000011 (Python ห้ามไม่ให้เขียน 0 นำหน้าจำนวนเต็ม เช่น 020)
float	จำนวนจริง	10.0 1.23e59 มีค่าเท่ากับ 1.23×10^{59}
str	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 เพราะไม่รู้ค่า 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)` ำค่าในดั แปร a b และ c มาแสดงต่อกันดัด้วยช่องว่างบนบรรทัดเดียวกัน
- `print(str(a)+str(b)+str(c))` ำค่าใ้ ตัวแปร a b และ c มาเปลี่ยนเป็นสตริงต่อกัน แล้วแสดงบนบรรทัดเดียวกัน

คำสั่งการอ่านข้อมูลจากแป้นพิมพ์

- `a = input()` อ่านข้อความจากแป้นพิมพ์หนึ่งบรรทัด เก็บใส่ตัวแปร `a` (เป็นสตริง)
 - `a = input().strip()` อ่านข้อความจากแป้นพิมพ์หนึ่งบรรทัด ดัดช่องว่างทางซ้ายและขวาออก 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()`
 - `x,y = [int(e) for e in 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

รับข้อมูลจากแป้นพิมพ์แล้วสลับแปลงเป็นจำนวน ก่อนนำไปคำนวณ	$x = \text{input}()$ $y = x**2 + 7$ ผิดเพราะ x เป็นสตริง
ลำดับการทำงานของตัวดำเนินการ $+ - * / // \% **$ ผิด $(** \text{ ทำก่อน } * / // \% \text{ ทำก่อน } + -)$	$y = x / 2*a$ จะได้ $y = (x/2)*a$ ถ้าต้องการคำนวณ $y = \frac{x}{2a}$ ต้องเขียน $y = x/(2*a)$ $y = x**1/3$ จะได้ $(x**1)/3$ ถ้าต้องการหารากที่สามของ x ต้องเขียน $y = x**(1/3)$
สลับใส่ $*$ สำหรับการคูณ	$y = 2x + 1$ ต้องเขียน $y = 2*x + 1$
$10e7$ 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	++ คือ การติดบวกค่าใน k สองครั้ง จึงมีค่าเท่ากับ k ค่าใน k ไม่เปลี่ยน -- k คือการติดลบค่าใน k สองครั้ง จึงมีค่าเท่ากับ k ค่าใน k ไม่เปลี่ยน
ลืม import math เมื่อใช้ฟังก์ชันของ math	$y = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$ จะพ้องว่าไม่รู้จัก math
ใส่วงเล็บเปิดกับปิดไม่ครบ	import math $y = 2 + (x * \text{abs}(y - z / 2))$ วงเล็บปิดมีน้อยไป $y = -b + \sqrt{b^2 - 4ac} / (2a)$ ขาดวงเล็บเปิด
สะกดชื่อตัวแปรผิด หรือผิดเรื่องการใช้ตัวอักษรเล็กกับใหญ่	count = 0 Count = count + 1 Count กับ count เป็นคนละตัว
ตั้งชื่อตัวแปรซ้ำกับชื่อฟังก์ชันมาตรฐานใน IDLE ตัวแปรที่ถูกต้องมีสีดำ เป็นสีอื่นจะสร้างปัญหา	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><u>Input</u>: รับจำนวนจริง 1 จำนวนจากแป้นพิมพ์เก็บใน a</p> <p><u>Process</u>: ให้ x มีค่าเป็น 1 จากนั้นทำซ้ำ</p> $x = (x + a/x)/2$ <p>จำนวน 4 ครั้ง</p> <p><u>Output</u>: ค่า x ที่ได้จากการทำงานข้างบนนี้</p>	
<p><u>Input</u>: 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><u>Process</u>: คำนวณ dot product ของเวกเตอร์ v กับ u</p> <p><u>Output</u>: ค่า dot product ที่คำนวณได้</p>	
<p><u>Input</u>: อ่านจำนวนจริง 4 จำนวนคั่นด้วยช่องว่างจากแป้นพิมพ์เก็บใน x1, y1, x2 และ y2 ค่าของ x1, y1 แทนพิกัดของจุดที่ 1 และ x2, y2 แทนพิกัดของจุดที่ 2 บนระนาบ x-y</p> <p><u>Process</u>: คำนวณระยะทางสั้นสุดระหว่างจุดทั้งสอง</p> <p><u>Output</u>: ระยะทางที่ทำได้</p>	
<p><u>Input</u>: อ่านพิกัดเชิงขั้วของจุดบนระนาบ ซึ่งเป็นจำนวนจริง 2 จำนวนคั่นด้วยช่องว่าง เก็บในตัวแปร r และ theta (เป็นเรเดียน)</p> <p><u>Process</u>: คำนวณค่า x และ y ซึ่งเป็นพิกัดคาร์ทีเซียนของจุด (r, theta) ที่อ่านเข้ามา</p> <p><u>Output</u>: ค่า x และ y (คั่นด้วยช่องว่าง)</p>	
<p><u>Input</u>: อ่านพิกัดคาร์ทีเซียนของจุดบนระนาบ ซึ่งเป็นจำนวนจริง 2 จำนวนคั่นด้วยช่องว่าง เก็บในตัวแปร x และ y</p> <p><u>Process</u>: คำนวณค่า r และ theta (เป็นเรเดียน) ซึ่งเป็นพิกัดเชิงขั้วของจุด (x, y)</p> <p><u>Output</u>: 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

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

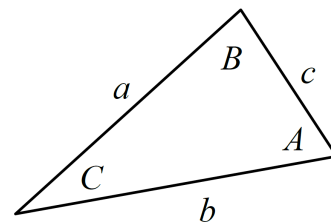
► ข้อมูลนำเข้า

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	ข้อผิดพลาด
<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> $1/2 \times 10 \times 10 \times \text{math.sin}(90) = 1/2 \times 10 \times 10 \times 1$ <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 <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

สูตรในการเปลี่ยนค่าองศาเซลเซียสไปเป็นองศาฟาเรนไฮต์และเคลวินมีดังนี้

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

$$K = C + 273.15$$

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

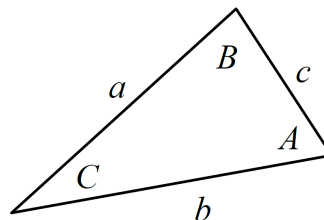
$$c^2 = a^2 + b^2 - 2ab \cos(C)$$

► 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

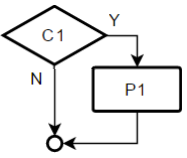
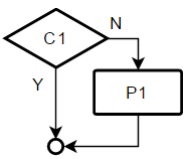
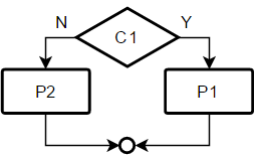
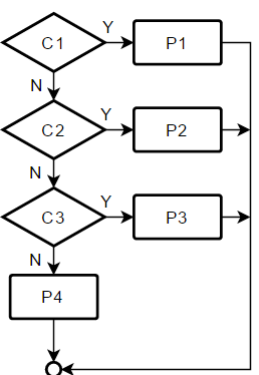
มีบรรทัดเดียว แสดง ISBN ทั้งสิบหลัก รับประกันว่ากรณีทดสอบจะไม่มีกรณีที่ n_{10} เท่ากับ 10

► 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 C1 : P1 else : P2</pre>
	<pre>if C1 : P1 elif C2 : P2 elif C3 : P3 else : P4</pre>

Flowchart	Code
<pre> graph TD 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] P4 --> End(()) </pre>	<pre> if C1 : P1 if C2 : P2 if C3 : P3 else : P4 </pre>
<pre> graph TD 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 --> J1(()) C4{C4} -- Y --> P5[P5] C4 -- N --> C5{C5} C5 -- Y --> P6[P6] C5 -- N --> J1 P5 --> J1 P6 --> J1 J1 --> P7[P7] P4 --> J2(()) P3 --> J2 J2 --> J1 </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') หรือ 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>	
<code>mx = max(a,b,c,d)</code>	<i>mx เก็บค่ามากที่สุดของ a,b,c และ d หรือเขียนโดยใช้ฟังก์ชัน max</i> <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)	

เข้าใจผิดเกี่ยวกับการเปรียบเทียบสตริง สตริงเปรียบเทียบกันตามลำดับแบบที่เขียนใน พจนานุกรม โดย	'abc' < 'eg' เป็นจริง '1234' < '9' เป็นจริง หมายถึงว่า 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
--	--

เรื่องที่ปรับปรุงได้

คำสั่งที่เหมือนกันทั้งในกลุ่มหลัง if และ กลุ่มหลัง else อาจแยกออกมาข้างนอกก็ได้	<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>
การใช้ if-elif-else ที่ตรวจค่าว่าตกอยู่ใน ช่วงใด สามารถลดการเปรียบเทียบลงได้ ถ้าจัด ลำดับการเปรียบเทียบให้เหมาะสม	<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>



Problem	Code
<p><u>Input</u>: รับจำนวนเต็ม 3 จำนวน คั่นด้วยช่องว่าง</p> <p><u>Process</u>: หาโมยฐานของจำนวนทั้ง 3</p> <p><u>Output</u>: มีฐานที่เท่าใด</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>: ขนาดเสื้อยืดโปโลตามรอบอกที่ได้รับ</p>	

ตัวอย่างการแก้ปัญหา

สลากกินแบ่ง

หากเราซื้อสลากกินแบ่งเรียงหมายเลขตั้งแต่หมายเลข $n1$ ต่อเนื่องไปจนถึงหมายเลข $n2$ (เช่น หมายเลข 10300 ถึง 13999) และงวดนี้รางวัลที่ 1 คือ หมายเลข $p1$ 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 ($n1 \leq p1 \leq n2$) ถ้าใช่ก็เพิ่มเงินรางวัล 10,000 บาท ตามด้วยการตรวจรางวัลเลขท้ายสองตัว แล้วก็สามตัว ในลักษณะเดียวกัน</p> <p>สั่ง run, ใส่ข้อมูลตามตัวอย่าง 01234 11 811 01000 01250, ได้ 10000 บาท ไม่ตรงตามตัวอย่าง ได้แก่รางวัลที่หนึ่ง 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> <pre>if n1%100 <= p2 <= n2%100</pre> <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 += .5*(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 หมายถึง 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></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		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		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 1 ชั่วโมง คิดค่าบริการชั่วโมงละ 10 บาท เศษของชั่วโมงคิดเป็นหนึ่งชั่วโมง
3. จอดรถตั้งแต่ 4 ชั่วโมง ถึง 6 ชั่วโมง คิดค่าบริการชั่วโมงที่ 4-6 ชั่วโมงละ 20 บาท เศษของชั่วโมงคิดเป็นหนึ่งชั่วโมง
4. จอดรถเกิน 6 ชั่วโมงขึ้นไป เหนียวจ่ายวันละ 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

CONFIDENTIAL

วโรส ไรณะ
Intania 85

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

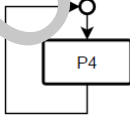
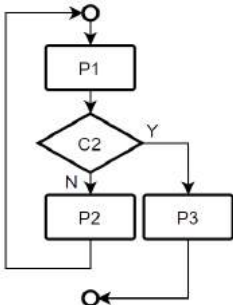
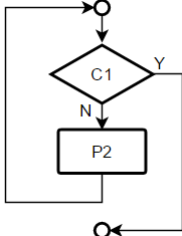
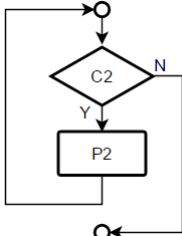


Programming จำเป็นมากในการเริ่มกิจการด้าน IT เพราะกิจการส่วนใหญ่ที่สามารถอยู่รอดได้ ผู้ก่อตั้งซึ่งเป็น
ผู้ที่เข้าใจผลิตภัณฑ์ที่จะสร้างมากที่สุด ต้องมีส่วนร่วมในการพัฒนาระบบเอง ถึงจะสามารถถ่ายทอดความคิด และ
ผลักดันผลิตภัณฑ์ที่ออกมาได้ เรียกว่ามีเงินมากแค่ไหน ถ้าต้องไปจ้างให้คนอื่นมาเขียนโปรแกรมแทนให้ยังงี้ก็สู้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

นอกจากนี้ **Programming** ยังเป็นการฝึกวิธีคิดแก้ปัญหาอย่างเป็นขั้นตอน ทำให้ผู้เรียนมีระบบการคิดซึ่งสามารถ
ประยุกต์ไปแก้ปัญหาได้ในหลายแขนงอย่างที่คุณเรียนคาดไม่ถึงอีกด้วย (เทพในหลาย ๆ วงการจบวิศวะ 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> while C1 : P1 while C2 : P2 P3 </pre>

ให้สังเกตว่า ภายในวงวน while ควรมีคำสั่งที่เปลี่ยนแปลงเงื่อนไขของ while ไม่งั้นมันจะวนทำงานไม่สิ้นสุด เช่น

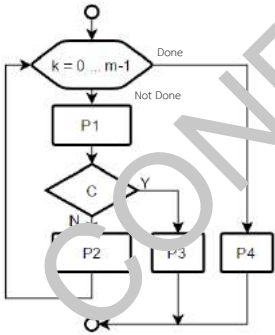
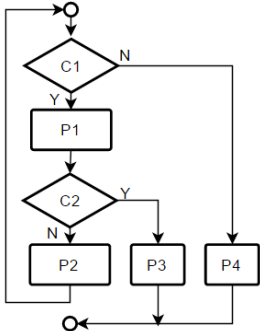
```
while i < j :
```

```
...
```

```
i += 2
```

เงื่อนไข $i < j$ ข้างบนนี้จะเป็เท็จได้ (เพื่อให้ออกจากวงวน) ก็ด้วยการที่ค่าของ i เพิ่มขึ้น หรือค่าของ j ลดลง
คำสั่ง $i += 2$ ในตัวอย่างข้างบนสร้างความมั่นใจว่า วงวนนี้ทำงานแล้วจะมีจุดสิ้นสุดและออกจากวงวน

Flowchart	Code	Flowchart	Code
	<pre> k = 0 while k < n : P1 k += 1 </pre>		<pre> for k in range(n) : P1 </pre>
			<pre> for p in range(m) : for k in range(n) : P1 </pre>

Flowchart	Code
	<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> 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) : ไม่ทำสักรอบ เพราะ step เป็นบวก และ start >= stop
- for k in range(9,10,-1) : ไม่ทำสักรอบ 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 for k in range(n)</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>เมื่อต้องการนำค่าที่สร้างจากรange(start, stop, step) มาใช้ในการประมวลผล</p> <p>ใช้ for k in range(...)</p>	<p>เช่น ต้องการหาว่า q เป็นจำนวนเฉพาะหรือไม่ ใช้วงวน for 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 while หรือใช้ if break 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 for y in range(x,n) 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>

เรื่องฝึกบ่อย

เข้าใจผิดเรื่องตัวสุดท้ายของ range	for k in range(1,5) k = 1,2,3,4 (ไม่รวม 5)
<p>ใช้วงวน ทำอะไรบางอย่างเพื่อสรุปว่า เป็น A หรือ เป็น B โดยจะ เป็น 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>ลิมปราคาของตัวแปรที่ใช้ในเงื่อนไขของ while หรือไม่ก็ปราคาผิต ความผิตพลาตแบบนี้อาจทำใหวทำงานไมลันสุด</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>หลังจากวงทำงานใน 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) # ได้ผลตามทางขวาบน 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	Cruce
<p><u>Input</u>: บรรทัดแรกรับจำนวนเต็มเก็บในตัวแปร n (n จะมีค่ามากกว่า 0) และอีก n บรรทัด รับจำนวนเต็ม บรรทัดละจำนวน</p> <p><u>Process</u>: หาผลต่างของค่ามากที่สุดกับค่าที่น้อยสุด และหาว่ามีกี่จำนวนที่เป็นเลขลบ</p> <p><u>Output</u>: ผลต่าง และจำนวนเลขลบ ที่ทำได้</p>	
<p><u>Input</u>: จำนวนเต็มเก็บในตัวแปร n</p> <p><u>Process</u>: หาจำนวนเต็มบวก w, x, y และ z ทั้งหมดที่ $w^3 = x^3 + y^3 + z^3$ โดยที่ $1 \leq x \leq y \leq z \leq n$</p> <p><u>Output</u>: ค่าของ w, x, y และ z ที่ทำได้</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

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</p> <p>สูตรการคำนวณมีลักษณะเป็นการบวกซ้ำ ๆ กันหลาย ๆ พจน์ที่ใช้ค่าของ k ที่เริ่มจาก 0 เพิ่มขึ้นทีละ 1 ไปเรื่อย ๆ ทำให้คิดถึงการใช้วงวน for ปัญหาคือจะต้องหาคำนวนกัรบ กี่พจน์ จึงจะละเอียดสุด ๆ ตามที่โจทย์ต้องการ</p> $\sin(x) = \sum_{k=0}^n (-1)^k \frac{x^{2k+1}}{(2k+1)!}$ <p>ทดลองตั้งให้ n = 30 มีตัวแปร s เก็บผลลัพธ์ คำนวนค่าของพจน์ตรงไปตรงมาตามสูตร ซึ่งมีการยกกำลังด้วย ** และการหาค่าแฟกทอเรียลด้วย 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 แต่จำนวนเต็มมีขนาดใหญ่เกินที่ float จะรับได้ แล้วจะรู้ได้อย่างไรว่า ผิดตรงไหน</p> <p>หาก run ด้วย IDLE เมื่อเกิดข้อผิดพลาดแล้ว เราสามารถขอดูตัวแปรต่าง ๆ ได้ เช่น อยากรู้ค่า k เมื่อเกิดความผิดพลาด ก็ใส่คำสั่ง print(k) ที่ Python shell จะแสดงว่า ผิดตอนที่ k มีค่าเป็น 85 ก็ลองคำนวณค่าต่าง 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 แต่จากที่บอกว่าผิดเพราะแปลงจำนวนเต็มที่ใหญ่เกินไปเป็น float ทำให้วิเคราะห์ได้ว่า ผลของการหาค่าแฟกทอเรียลต้องแปลงเป็น 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>แก้ปัญหานี้โดยไม่ใช้จำนวนเต็มขนาดใหญ่ แต่มองว่า การคำนวณหาพจน์ในแต่ละรอบนั้น สามารถคำนวณได้จากพจน์ในรอบก่อนได้ ไม่จำเป็นต้องคำนวณ $(-1)^k$, x^{2k+1} และ $(2k+1)!$ ทุกรอบ แต่สามารถคำนวณจากความสัมพันธ์ดังนี้</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 แสดงว่าการคำนวณตั้งแต่รอบที่ 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)**k**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

โจทย์นี้สนใจเฉพาะสามเหลี่ยมมุมฉากที่ความยาวด้านทุกด้านเป็นจำนวนเต็ม

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

สนธิวัฒน์ 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 ของอักขระสุดท้ายคือ 0 อักขระแต่ละตัวในสตริงเป็นได้ทั้งตัวอักษร ตัวเลข และสัญลักษณ์พิเศษต่าง ๆ การเขียนสตริงทำได้หลายแบบดังนี้

- เขียนครอบด้วยอักขระประกาศเดี่ยว `'I am "Python".'` `'I\'m Python.'`
- เขียนครอบด้วยอักขระประกาศคู่ `"I'm Python."` `"I am \"Python\"."`
- เขียนครอบด้วยอักขระประกาศเดี่ยวสามตัวติด `'''I'm "Python".'''`
- เขียนครอบด้วยอักขระประกาศคู่สามตัวติด `"""I'm "Python".""""`

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

ตัวอย่างการใช้อักขระและสตริงย่อยในสตริง (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'