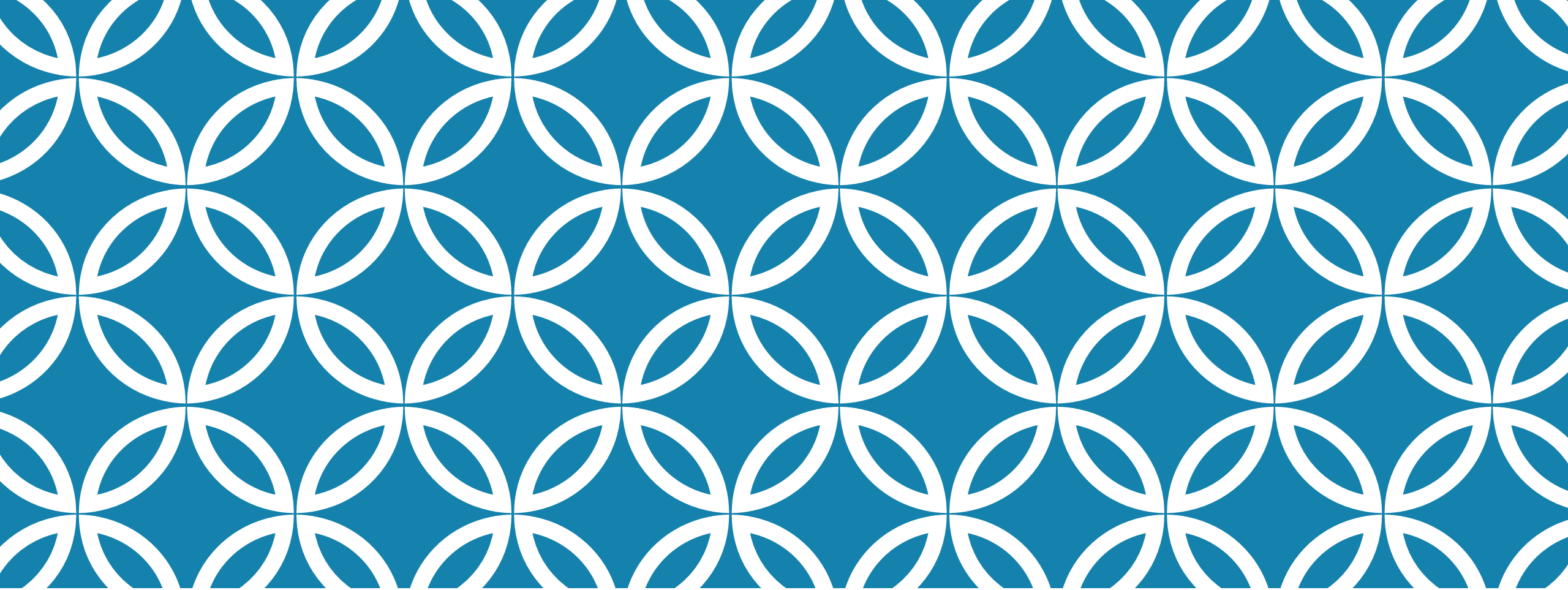


COMPUTATIONAL THINKING

01418112 Fundamentals Programming Concept

COMPUTATIONAL THINKING CHARACTERISTICS

Conceptualizing, not programming	• วิทยาการคอมพิวเตอร์ ไม่ใช่ การเขียนโปรแกรม การคิดตามแนวทางของวิทยาการคอมพิวเตอร์ หมายถึง ความสามารถในการคิดได้หลายระดับ – multiple levels of abstraction
Fundamental, not rote skill	• เป็นทักษะพื้นฐานที่คนในสังคมปัจจุบันต้องรู้ ไม่ใช่ การท่องจำ หรือสิ่งที่ปฏิบัติเป็นประจำในลักษณะของเครื่องจักร ซึ่งใช้ความคิดเพียงเล็กน้อย หรือไม่ใช้เลย
A way that humans, not computers, think	• เป็นแนวทางในการแก้ปัญหา ไม่ใช่ให้คนคิดเหมือนคอมพิวเตอร์ มนุษย์เป็นผู้สร้างระบบการทำงานที่มีคอมพิวเตอร์เป็นเครื่องมือหนึ่งในการแก้ปัญหา และเป็นระบบที่ถูกจำกัดโดยจินตนาการของผู้สร้าง

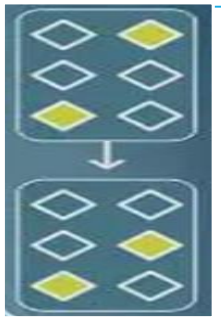
COMPUTATIONAL THINKING CHARACTERISTICS

Complements and combines mathematical and engineering thinking	• วิทยาการคอมพิวเตอร์อยู่บนพื้นฐานของการคิดเชิงคณิตศาสตร์และวิศวกรรมศาสตร์ ช่วยให้เราสามารถสร้างระบบที่มีปฏิสัมพันธ์กับโลกแห่งความเป็นจริง
Ideas, not artifacts	• การคิดเชิงคำนวณไม่ใช่ซอฟต์แวร์หรือฮาร์ดแวร์ที่มีผู้เขียนหรือประดิษฐ์ขึ้น เราใช้แนวคิดเชิงคำนวณเป็นแนวทางในการแก้ปัญหา ใช้ในการดำเนินชีวิต ใช้ในการสื่อสาร และ/หรือ ปฏิสัมพันธ์กับผู้อื่น
For everyone, everywhere	• ทุกคน ทุกวัย สามารถนำการคิดเชิงคำนวณไปใช้ได้ในทุกสถานการณ์

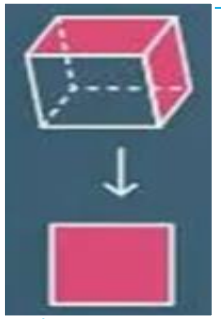
COMPUTATIONAL THINKING

การคิดเชิงคำนวณช่วยพัฒนาทักษะและเทคนิคสำหรับแก้ปัญหาอย่างมีประสิทธิภาพ โดยอาจใช้หรือไม่ใช้คอมพิวเตอร์เป็นเครื่องมือ องค์ประกอบหลักของการคิดเชิงคำนวณ มีดังนี้:

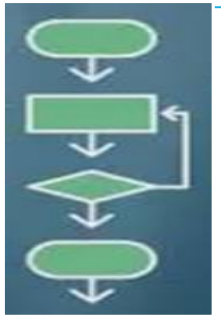
Decomposition: แบ่งปัญหาออกเป็นปัญหาย่อย ๆ ที่มีขนาดเล็กลง และ/หรือ สามารถจัดการได้ง่ายกว่า โดยพิจารณาว่าเราสามารถแบ่งปัญหาหลักออกเป็นปัญหาย่อยได้อย่างไร และแต่ละปัญหาย่อยมีลักษณะเป็นอย่างไร



Pattern recognition: มองหาลักษณะร่วมและลักษณะที่แตกต่างกันของปัญหา (ย่อย) ใหม่ กับปัญหา (ย่อย) เดิม ที่เราเคยพบ เพื่อนำแนวทางการแก้ปัญหาเดิมมาใช้ หรือคาดเดารูปแบบที่อาจเกิดขึ้น



Abstraction: แยกรายละเอียดปลีกย่อยออกจากหลักการ/ประเด็นหลักของปัญหา เพื่อนำไปสู่คำตอบที่ต้องการ รวมถึงเป็นแนวทางในการนำไปใช้หาคำตอบกับปัญหาอื่น ๆ ที่มีลักษณะทำนองเดียวกันได้

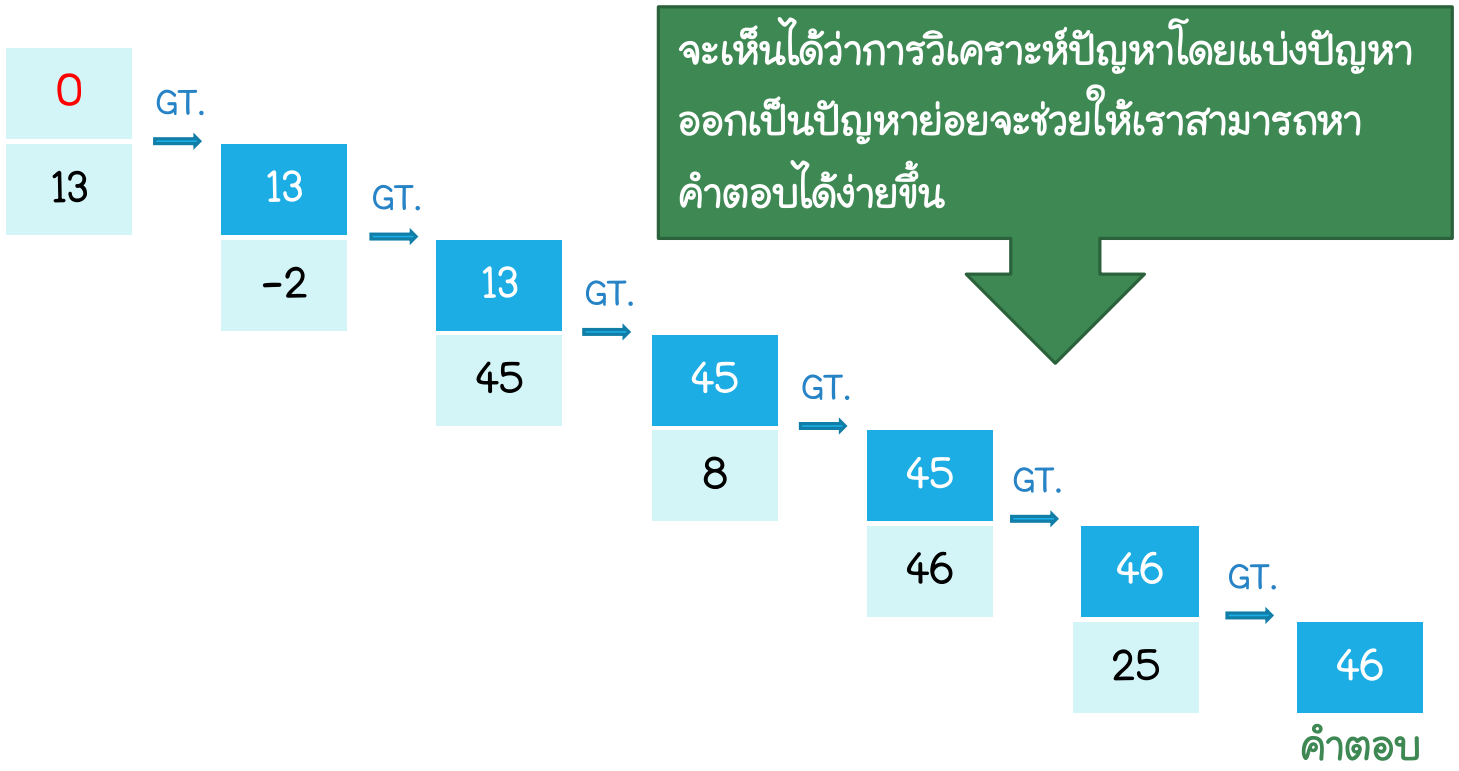


Algorithm design: กำหนดลำดับการทำงานเพื่อใช้ในการแก้ปัญหา เป็นขั้น ๆ ที่สามารถลงมือปฏิบัติตามได้

https://cdn-images-1.medium.com/max/1600/1*g5HrLzOsI5XVsLf0uI0a8g.jpeg

DECOMPOSITION

ตัวอย่างที่ 1 หาค่ามากที่สุดจากชุดข้อมูล 13, -2, 45, 8, 46, 25

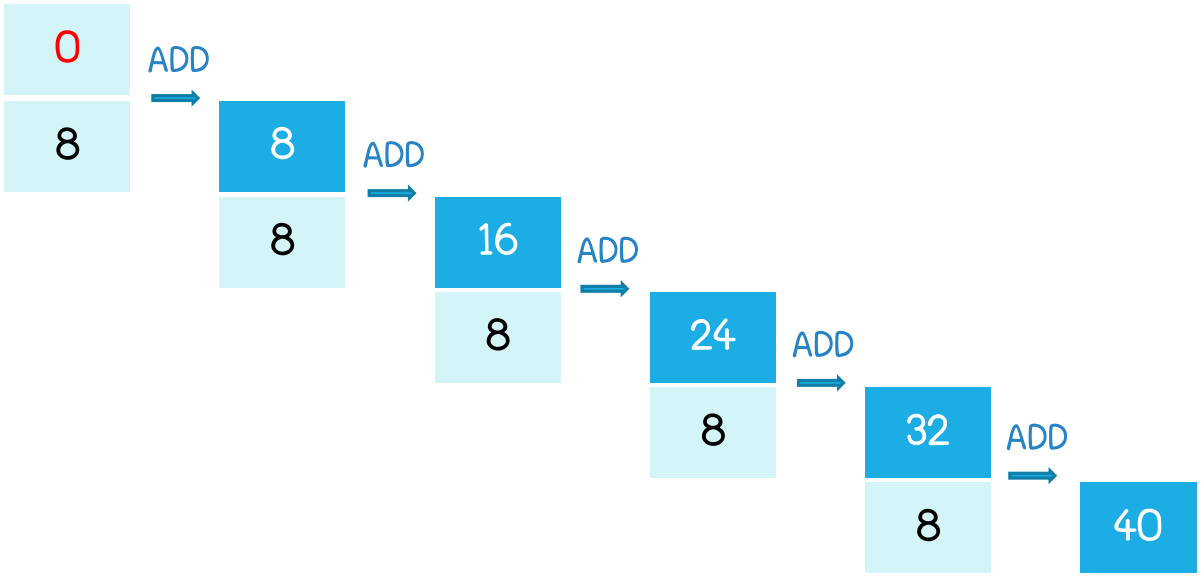


วิเคราะห์:

- การดำเนินการพื้นฐานในการหาค่ามากที่สุดของข้อมูล ได้แก่การเปรียบเทียบจำนวน 2 จำนวนเข้าด้วยกัน
- ในกรณีของข้อมูลตัวแรก (หรือมีข้อมูลเพียง 1 ตัว) เพื่อให้ได้ข้อมูลดังกล่าวเป็นค่าที่มากที่สุด เราจึงเปรียบเทียบกับ 0
- จากนั้นจึงนำค่ามากที่สุดที่ได้ไปเปรียบเทียบกับข้อมูลตัวถัดไป
- ดำเนินการจนครบทุกข้อมูล

DECOMPOSITION

ตัวอย่างที่ 2 หาผลลัพธ์ของนิพจน์ 8*5 โดยไม่ใช้การคูณ

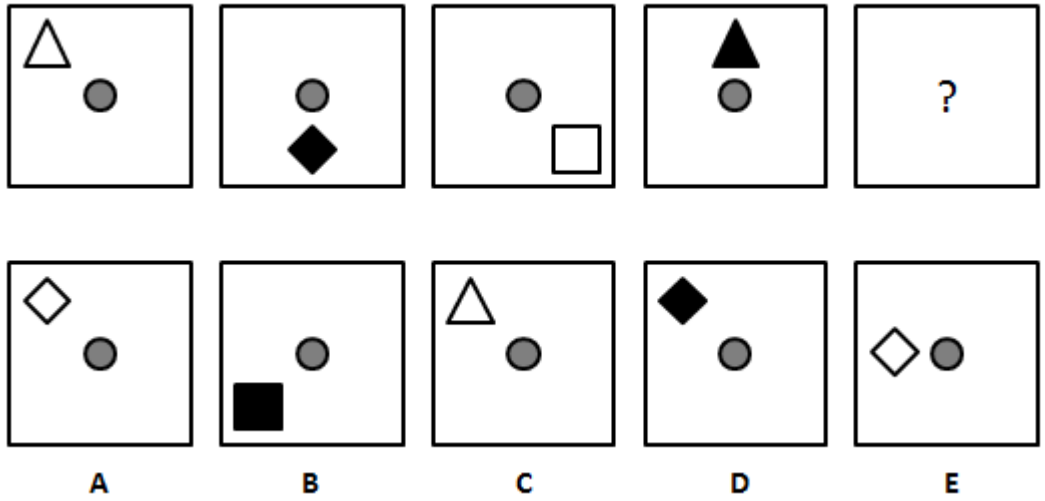


คำถาม: หาผลลัพธ์ของปัญหาการผลคูณของชุดข้อมูล
20, 6, -2, $\sqrt[3]{8}$, $\log_{10}1$, 11

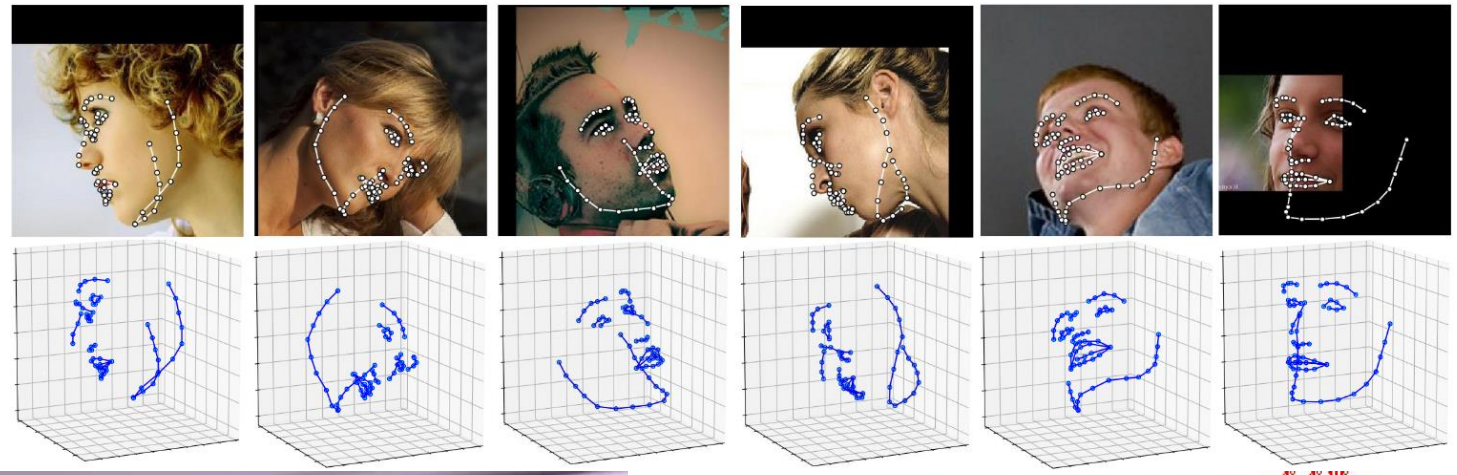
วิเคราะห์:

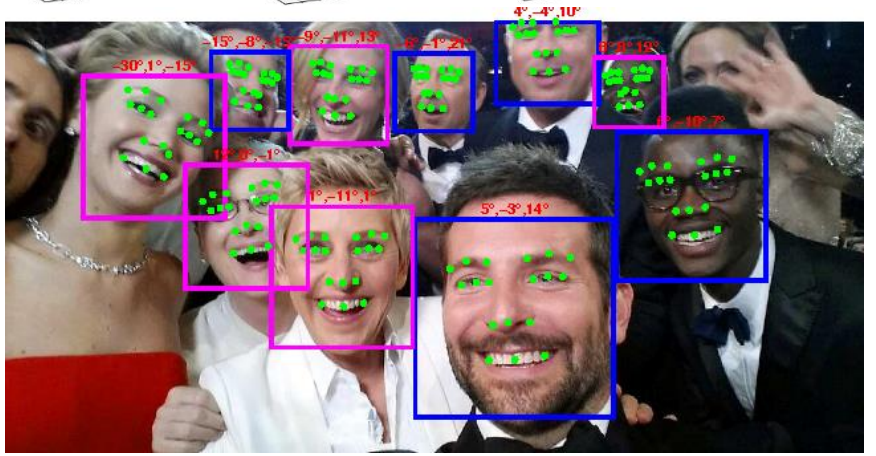
- การคูณเกิดจากการบวกซ้ำ ๆ หรือบวกสะสม ในที่นี้ 8*5 = 8+8+8+8+8
- การบวกเป็นการดำเนินการของ 2 จำนวน
- ในกรณีของข้อมูลตัวแรก (หรือมีข้อมูลเพียง 1 ตัว) เพื่อให้ได้ผลบวกสะสมของจำนวนเดียว เราจึงบวกกับ 0
- จากนั้นจึงนำผลบวกสะสมที่ได้ไปบวกสะสมกับข้อมูลตัวถัดไป
- ดำเนินการจนครบทุกข้อมูล

PATTERN RECOGNITION

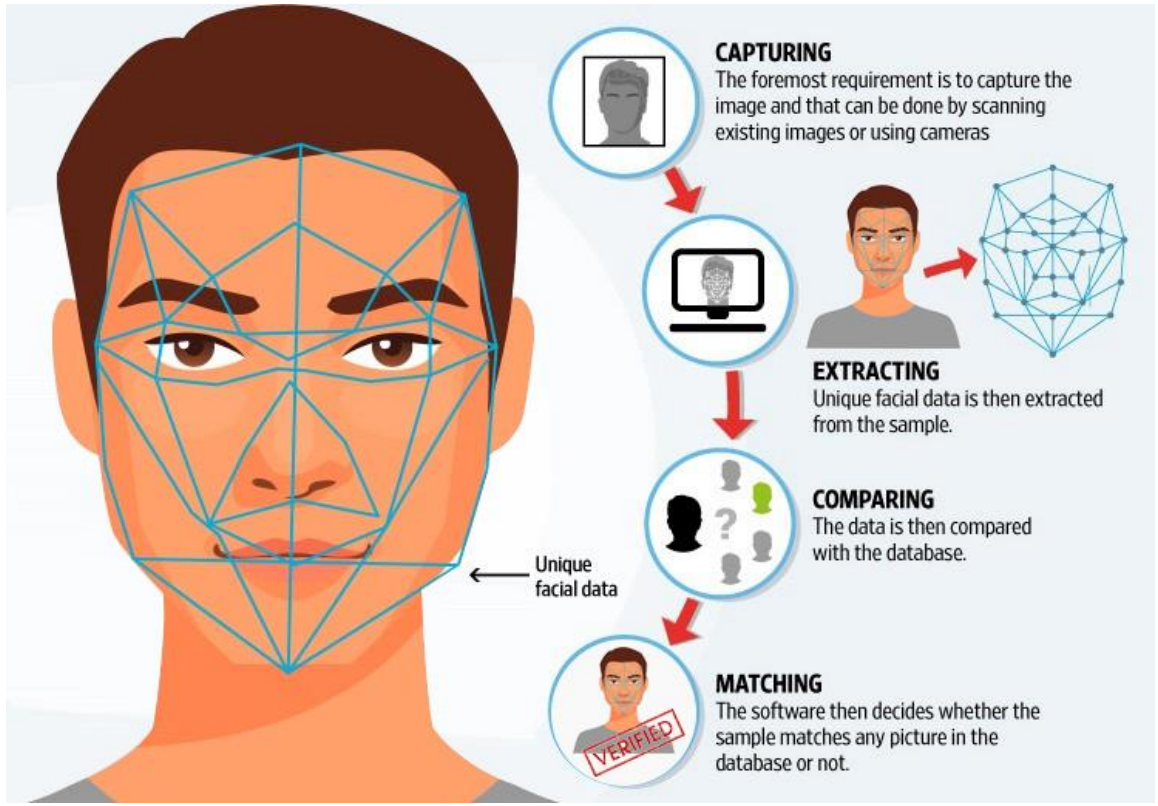


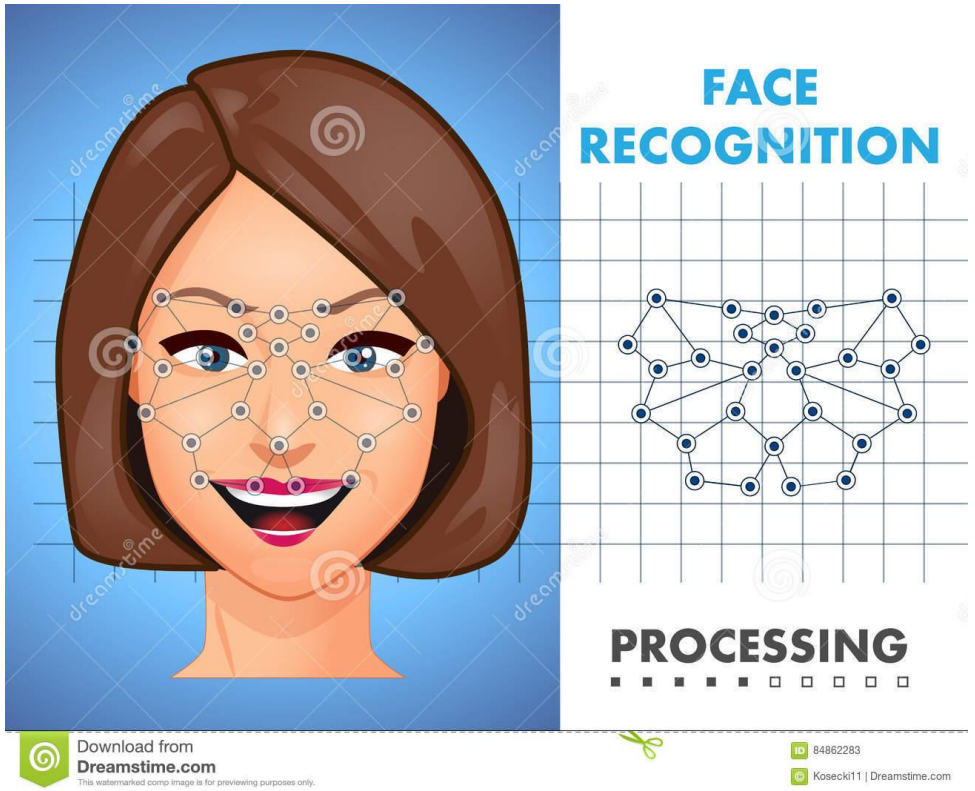
FACE RECOGNITION





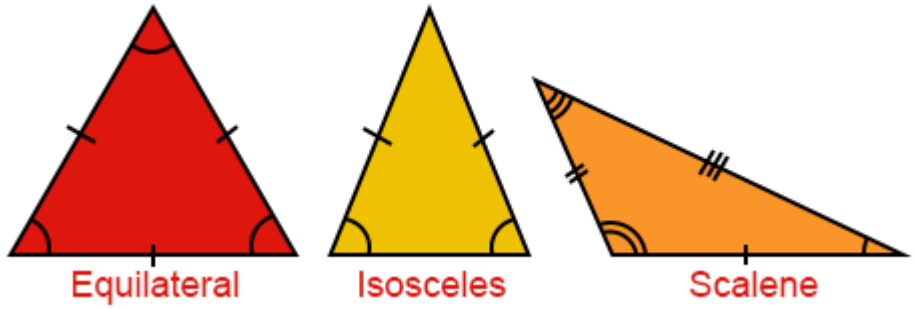
FACE RECOGNITION





ABSTRACTION

ตัวอย่างที่ 3 จงหาพื้นที่ของแต่ละรูปต่อไปนี้



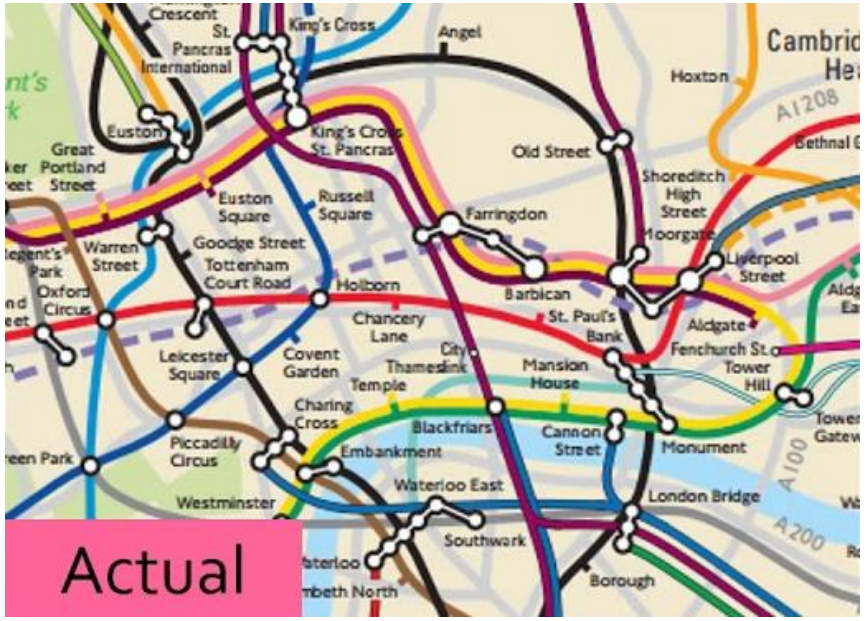
วิเคราะห์: เราพบว่ารูปทั้งสามมีรายละเอียดหลายอย่าง ได้แก่ รูปทรง สี ความยาวด้าน และ ขนาดของมุม

เราแบ่งการมองปัญหาเป็น 2 ส่วน คือ ส่วนรายละเอียดที่แตกต่างกัน (สี ความยาวด้าน ขนาดของมุม) และส่วนรูปทรงสามเหลี่ยม ซึ่งในที่นี้จำเป็นต่อการหาพื้นที่ โดยสูตรในการหาพื้นที่ คือ พื้นที่สามเหลี่ยม = 1/2 * ฐาน * สูง

เพื่อหาคำตอบของปัญหาเราแยกส่วนที่ไม่เกี่ยวกับการหาพื้นที่ และพิจารณาเฉพาะส่วนที่จำเป็นเท่านั้น

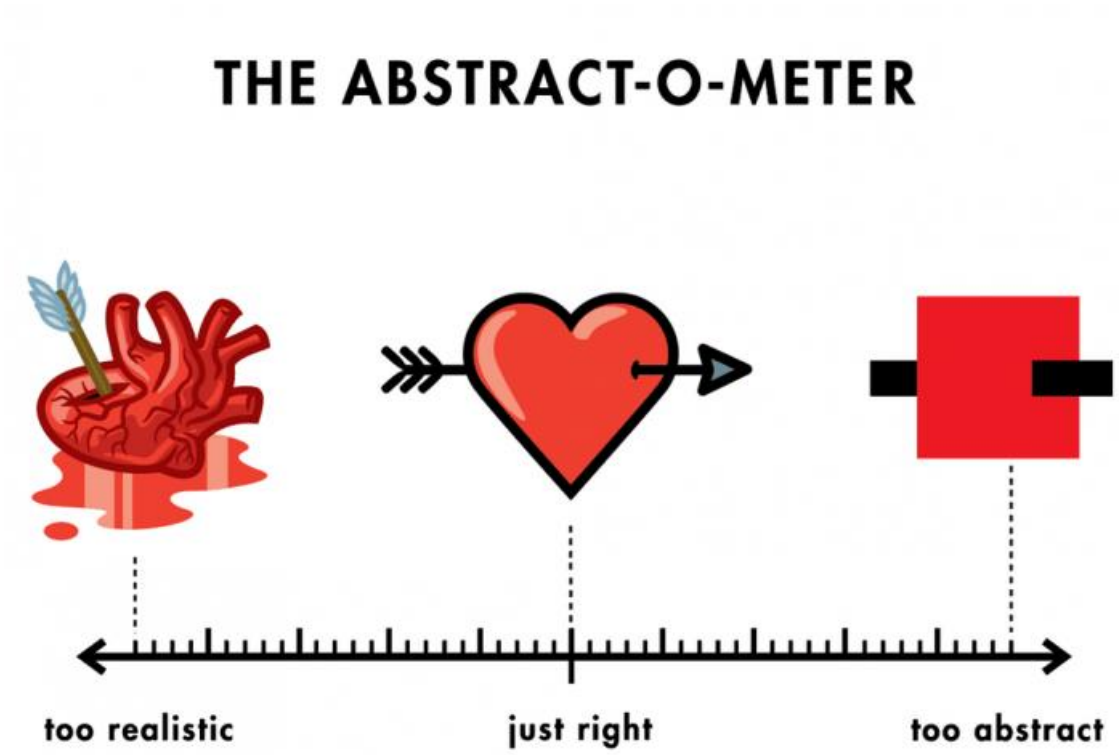
เราสามารถหาพื้นที่ของรูปทั้งสามได้ โดยการหาความยาวฐานและส่วนสูงของสามเหลี่ยมแต่ละรูป แล้วนำมาคำนวณ

EXAMPLES OF ABSTRACTION



https://stackoverflow.com/questions/19291776/whats-the-difference-between-abstraction-and-generalization

EXAMPLES OF ABSTRACTION



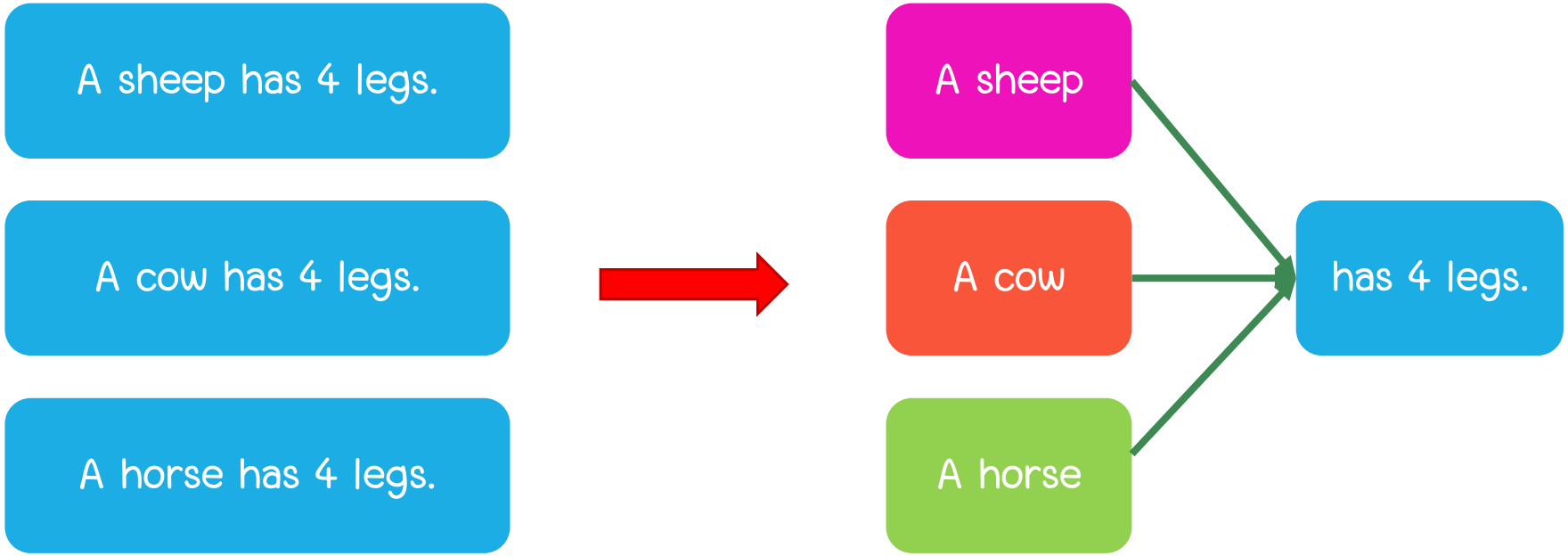
https://computersciencewiki.org/index.php/Abstraction

SIX COMPUTATIONAL THINKING CONCEPTS

Logic คาดการณ์และวิเคราะห์	• Logic เป็นการศึกษาเกี่ยวกับการให้เหตุผล ใช้สำหรับตรวจสอบความจริง
Algorithms กำหนดขั้นตอน/กฎ	• อัลกอริทึมเป็นลำดับขั้นตอน/คำสั่งหรือชุดของกฎ สำหรับดำเนินการอย่างใดอย่างหนึ่ง
Decomposition การแบ่งงานออกเป็นงานย่อย	• การแบ่งปัญหาออกเป็นส่วนย่อย ๆ หลายระดับ เพื่อช่วยให้สามารถแก้ปัญหาที่ซับซ้อนได้ง่ายขึ้น
Patterns ระบุรูปแบบ/ลักษณะที่เหมือนกัน	• ใช้สำหรับคาดการณ์ กำหนดกฎ ช่วยให้แก้ปัญหาได้อย่างกว้างขวาง หรืออาจเรียกว่าเป็นการทำ generalization
Abstraction มองข้ามรายละเอียดที่ไม่เกี่ยวข้อง	• สนใจประเด็นหลักที่เกี่ยวกับการแก้ปัญหา ลดความซับซ้อนของปัญหา ทำให้การแก้ปัญหาง่ายขึ้น
Evaluation การประเมินผล	• การประเมินผล/การตัดสินใจอย่างเป็นระบบ เป็นไปตามวัตถุประสงค์ ข้อเท็จจริง เงื่อนไข ที่กำหนด

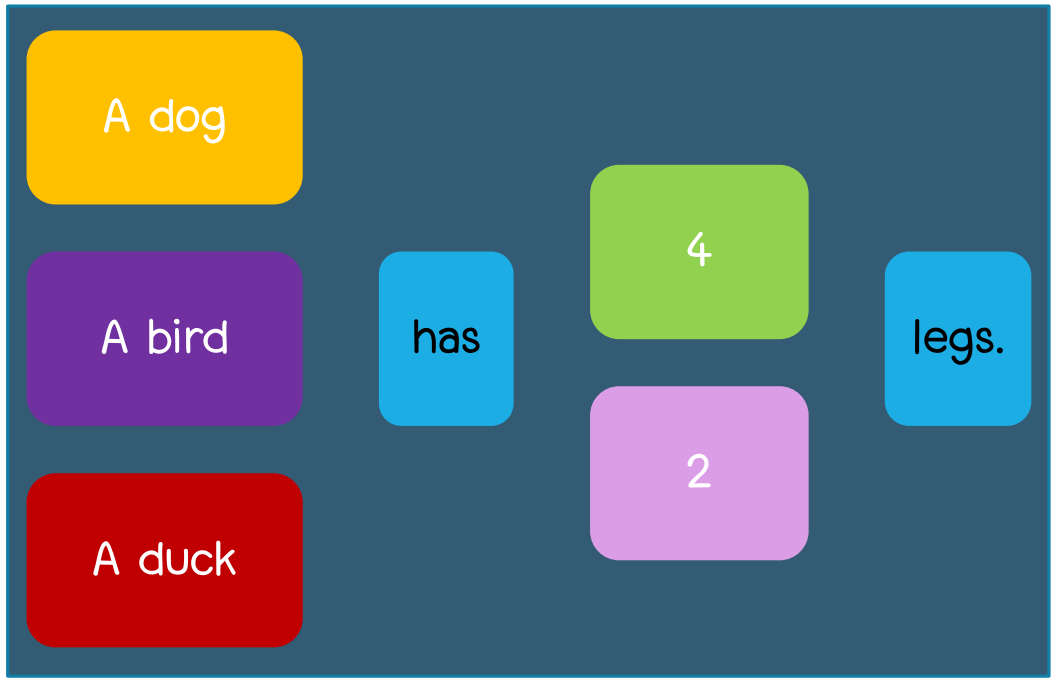
GENERALIZATION

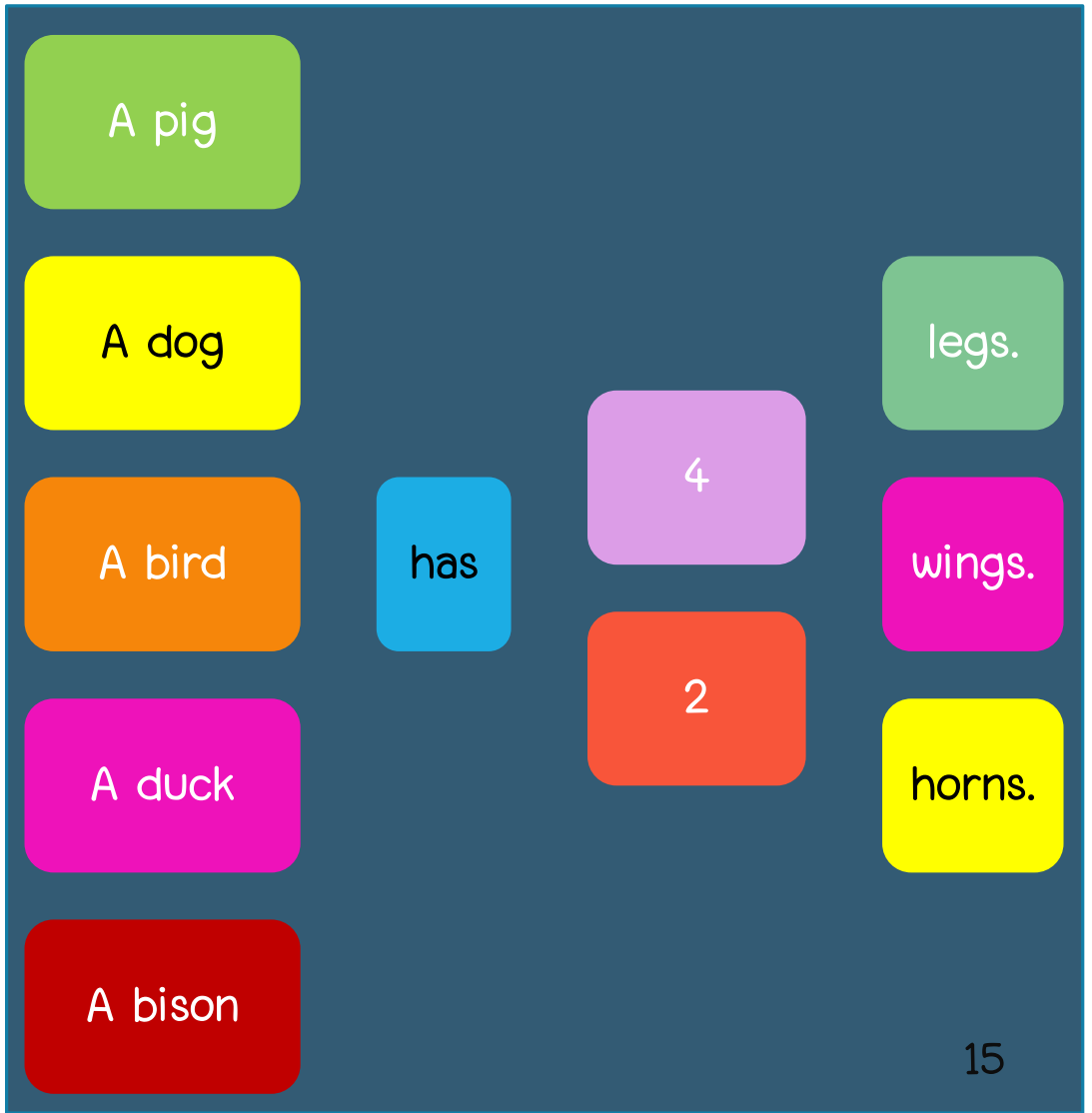
ตัวอย่างที่ 4 พิจารณาข้อความด้านซ้าย เราสามารถสร้างรูปแบบของข้อความตามรูปด้านขวา เช่น



GENERALIZATION

รูปแบบของข้อความมีได้หลากหลาย



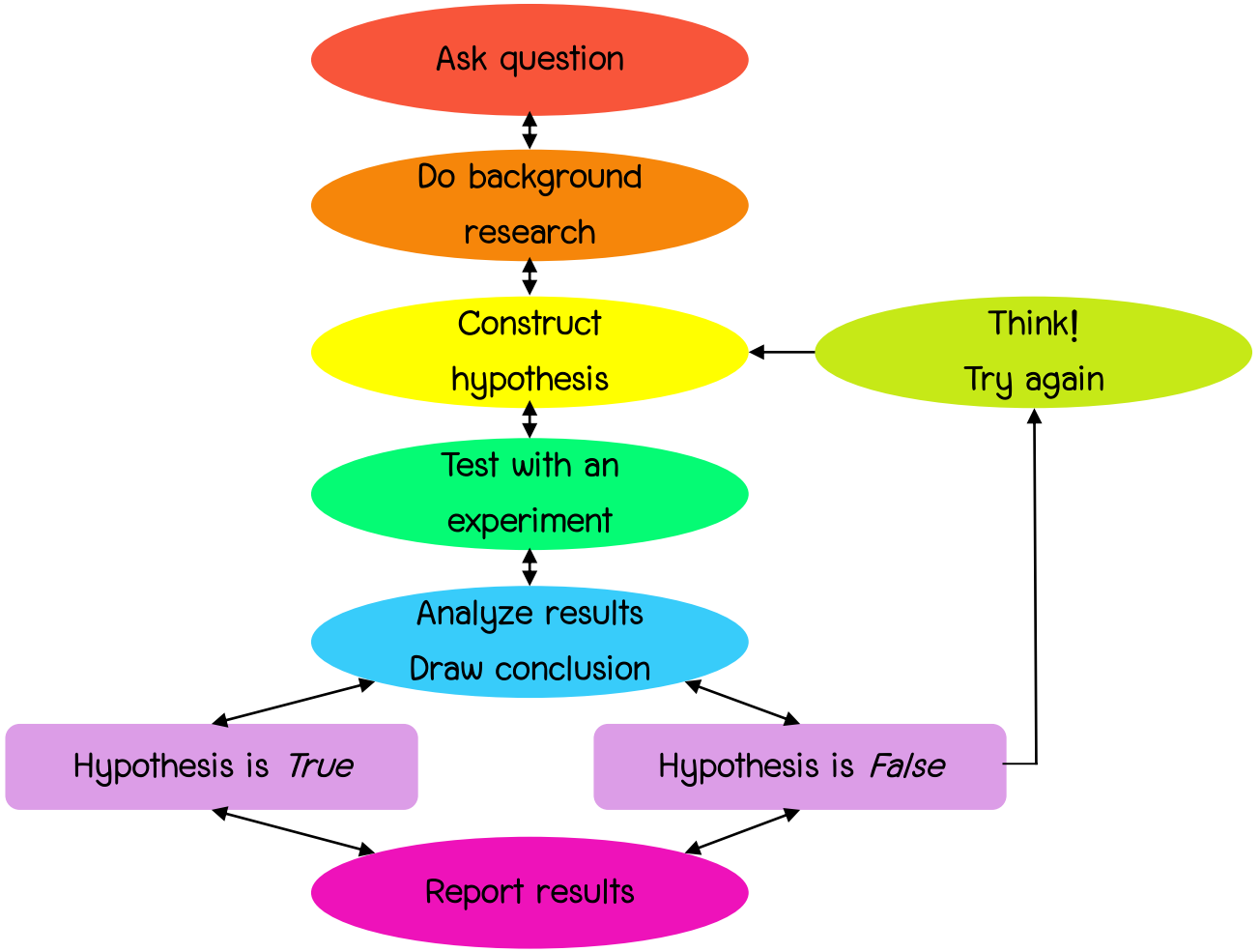


COMPUTATIONAL THINKING SKILLS

Computational Thinking ประกอบด้วยทักษะหลากหลายที่ใช้สำหรับแก้ปัญหา

Logical thinking การคิดเชิงตรรกะ	• เป็นการพิจารณาโดยใช้เหตุผลเป็นขั้นเป็นตอนตามลำดับ เพื่อสามารถคาดการณ์ หาข้อสรุปใหม่จากข้อมูลที่มีอยู่
Algorithmic thinking การคิดอย่างเป็นขั้นเป็นตอน	• ช่วยแก้ปัญหาที่มีลักษณะเดียวกัน โดยกำหนดคำตอบที่สามารถใช้ได้กับทุกสถานการณ์
Efficient solution คำตอบที่มีประสิทธิภาพ	• แก้ปัญหาโดยใช้ทรัพยากรที่มีอยู่จำกัดเท่าที่จำเป็น • ทรัพยากรที่สำคัญ คือ เวลา
Scientific thinking การคิดเชิงวิทยาศาสตร์	• ทักษะในการประเมินว่าสิ่งที่พบเป็นจริงหรือไม่

SCIENTIFIC THINKING



https://www.researchgate.net/post/What_is_Scientific_thinking_and_how_it_formed

ตัวอย่างปัญหาการคิดเชิงตรรกะ

ถ้ามีกล่อง 4 ใบอยู่บนโต๊ะ มีกล่อง สีดำ สีแดง สีน้ำเงิน และ สีเขียว และมีกติกา คือ

- ไม่อนุญาตให้หยิบกล่อง สีแดง สีดำ และ สีน้ำเงิน พร้อมกัน
- ไม่อนุญาตให้หยิบกล่อง สีน้ำเงิน สีเขียว และ สีแดง พร้อมกัน

ถ้าต้องการหยิบกล่อง 2 ใบออกจากโต๊ะ จะเป็นกล่องสีอะไร

การคิดเชิงตรรกะ

จากหนังสือ Brain Building ของ Dr. Karl Albrecht เขากล่าวว่า

“การเรียนรู้คณิตศาสตร์เป็นกระบวนการที่เกิดขึ้นแบบตามลำดับ” นั่นคือ ถ้าเราไม่เข้าใจหลักการพื้นฐาน ข้อเท็จจริง หรือกระบวนการบางอย่าง เราจะไม่สามารถเข้าใจสิ่งที่เป็นผลสืบเนื่องจากสิ่งเหล่านั้นได้เลย

ตัวอย่างลำดับความรู้ทางคณิตศาสตร์

- จำนวนนับ
  - การบวก ลบ คูณ หาร จำนวนนับ
    - ตัวประกอบ จำนวนเฉพาะ
      - ตัวคูณร่วมน้อย ตัวหารร่วมมาก
        - เศษส่วน และการบวก ลบ คูณ หาร เศษส่วน
          - ทศนิยม และการบวก ลบ คูณ หาร ทศนิยม

COMPUTATIONAL THINKING AND MATHEMATICAL REASONING

การแก้ปัญหาคณิตศาสตร์ในระดับประถมมี 2 ขั้นตอน:

1. ทำความเข้าใจโจทย์
2. หาคำตอบ

ในการทำความเข้าใจประโยคสัญลักษณ์ 23 + 39 ผู้เรียนจะต้องสามารถเข้าใจความหมายของสัญลักษณ์ที่สอดคล้องกับแนวทางการแก้ปัญหา แล้วคิดลำดับขั้นตอนที่จะสามารถใช้ในการคำนวณหาคำตอบ จากนั้นดำเนินการคำนวณเพื่อหาคำตอบ

พิจารณาโจทย์ปัญหาที่อยู่ในรูปของข้อความ (word problem) เช่น นกซื้อส้มสามลูก ราคาลูกละ 8 บาท นกจ่ายเงินด้วยธนบัตร 50 บาท นกจะได้รับเงินทอนกี่บาท

ขั้นตอนแก้ปัญหาข้างต้น พร้อมกับการทำ abstraction เพื่อสามารถสร้างประโยคสัญลักษณ์ ที่จะนำไปสู่การหาคำตอบ คือ

$$50 - 3 \times 8$$

คำถาม: รายละเอียดใดบ้างในการทำ abstraction ที่ได้ถูกแยกออกจากสาระสำคัญของการหาคำตอบ

POLYA'S PROBLEM SOLVING PRINCIPLES

1. Understand the problem ทำความเข้าใจปัญหา

2. Devise a plan วางแผนแก้ปัญหา เพื่อเชื่อมโยงข้อมูลที่มีอยู่กับคำตอบที่ต้องการหา

3. Carry out the plan ดำเนินการตามแผนเพื่อหาคำตอบ

4. Look back ตรวจสอบคำตอบ

POLYA'S PROBLEM SOLVING PRINCIPLES

Understand the problem

- อยากหาคำตอบอะไร
- มีข้อมูลอะไรบ้าง
- มีเงื่อนไขอะไรบ้าง
- สิ่งที่มีอยู่เพียงพอกับการหาคำตอบของปัญหาหรือไม่

Devise a plan

- เคยพบปัญหาที่มีลักษณะคล้ายคลึงกับปัญหานี้หรือไม่
- เข้าใจปัญหาที่เคยพบอย่างไรบ้าง
  - สามารถใช้คำตอบจากปัญหาที่เคยพบหรือไม่
  - สามารถใช้วิธีการเดียวกับปัญหาที่เคยพบได้หรือไม่
- มีความรู้เกี่ยวกับสูตร/ทฤษฎีที่จะใช้แก้ปัญหานี้หรือไม่
- สามารถหาคำตอบบางส่วนของปัญหาได้หรือไม่

POLYA'S PROBLEM SOLVING PRINCIPLES

Carry out the plan

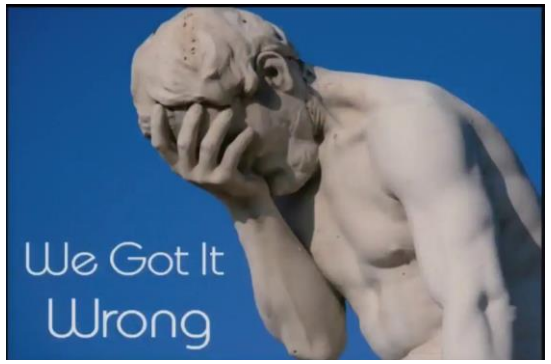
- มั่นใจว่าขั้นตอนที่ใช้แก้ปัญหามีความถูกต้อง
- สามารถพิสูจน์ได้หรือไม่ว่าขั้นตอนมีความถูกต้อง

Look back

- มีวิธีการอื่นในการหาคำตอบหรือไม่
- สามารถใช้คำตอบ หรือวิธีการแก้ปัญหานี้กับปัญหาอื่นได้อีกหรือไม่

PAPERT'S CONCEPT: DO NOT BE AFRAID OF 'BEING WRONG'

Children model of learning



When you program a computer you almost never get it right the first time.

Learning to be a master programmer is learning to become highly skilled at isolating (identifying) and correcting bugs.

The question to ask about the program is not whether it is right or wrong,

but if it is fixable.

REFERENCES

- https://ai2-s2-public.s3.amazonaws.com/figures/2017-08-08/b2cd92d930ed9b8d3f9dfcfff733f8384aa93de8/1-Figure1-1.png
- https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcQJoi8PNP8AA0du8xTmdwxzTJ72Rkbu8h-cIgOHkB-Uk2vU_6-_dQ
- http://www.sara-technology.com/wp-content/uploads/2018/07/face-detect.jpeg
- https://thumbs.dreamstime.com/z/face-recognition-biometric-security-system-concept-84862283.jpg
- https://sayingimages.com/you-got-it-dude-meme/
- https://www.youtube.com/watch?v=OMYcaEyzxBs
- http://games.thinkingmyself.com/
- https://stackoverflow.com/questions/19291776/whats-the-difference-between-abstraction-and-generalization