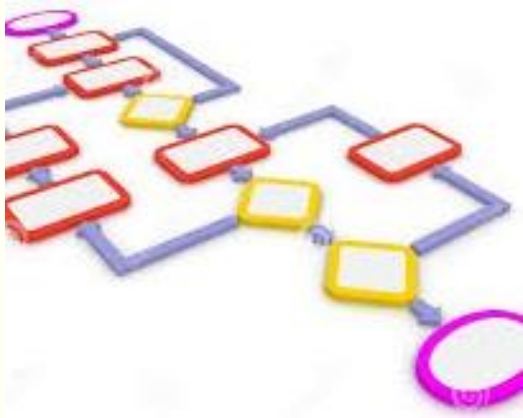
การออกแบบและวิเคราะห์ขั้นตอนวิธี

DESIGN AND ANALYSIS OF ALGORITHMS

02-212-212



อ.ธิดาวรรร คล้ายศรี

บทที่1

Fundamental of Design and Analysis Algorithms

1.1 การแก้ปัญหาและขั้นตอนวิธี (Problems Solving and Algorithms)

1.2 An Introduction to Algorithm Design (บทนำการออกแบบขั้นตอนวิธี)

1.3 คณิตศาสตร์พื้นฐานเพื่อการวิเคราะห์ (Mathematic Preliminaries for Analysis)

1.4 Analysis of Algorithms (การวิเคราะห์อัลกอริธึ่ม)

1.5 Growth of Functions (แนวโน้มการเพิ่มขึ้นของฟังก์ชั่น)

1.1 การแก้ปัญหาและขั้นตอนวิธี (Problems Solving and Algorithms)

- **ขั้นตอนวิธี (Algorithm) = ?**

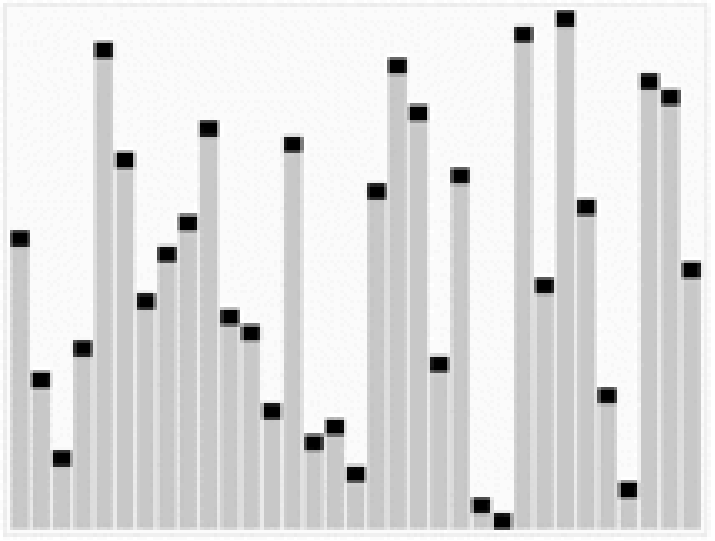
→วิธีการ/เครื่องมือที่ช่วยแก้ปัญหาอย่างเป็นขั้นตอนตามลำดับ เพื่อให้ได้ผลลัพธ์อย่างมีประสิทธิภาพ

บางปัญหาต้องการวิธีการทางคอมพิวเตอร์ (computer algorithm) ช่วยในการแก้ปัญหา ซึ่งแตกต่างจากการแก้ปัญหาแบบสามัญสำนึก (heuristic)

→ขั้นตอนวิธีในการแก้ปัญหาอาจนำเสนอ: natural languages, pseudocode, flowcharts

เมื่อได้เขียนขั้นตอนวิธีแล้วก็จะทำการพัฒนาโปรแกรมด้วยภาษาคอมพิวเตอร์ (High-level programming languages: C++, Java) ให้ได้ผลลัพธ์ตามขั้นตอนวิธีที่ได้ออกแบบไว้

Quick Sort Algorithms)



Natural language

a high-level description English prose, as:
High-level description:

1.If there are no numbers in the set then there is no highest number.

2. Assume the first number in the set is the largest number in the set.
3. For each remaining number in the set: if this number is larger than the current largest number, consider this number to be the largest number in the set.
4. When there are no numbers left in the set to iterate over, consider the current largest number to be the largest number of the set.

[Wikipedia]

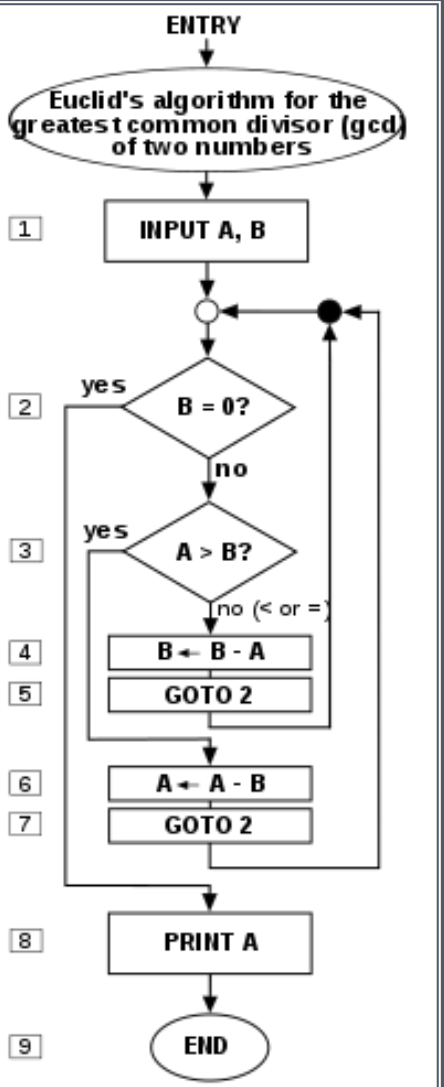
Algorithms

Pseudocode

```
Algorithm LargestNumber
  Input: A list of numbers L.
  Output: The largest number in the list L.


  if L.size = 0 return null
  largest ← L[0]
  for each item in L, do
    if item > largest, then
      largest ← item
  return largest
```

Flowchart



[Wikipedia]

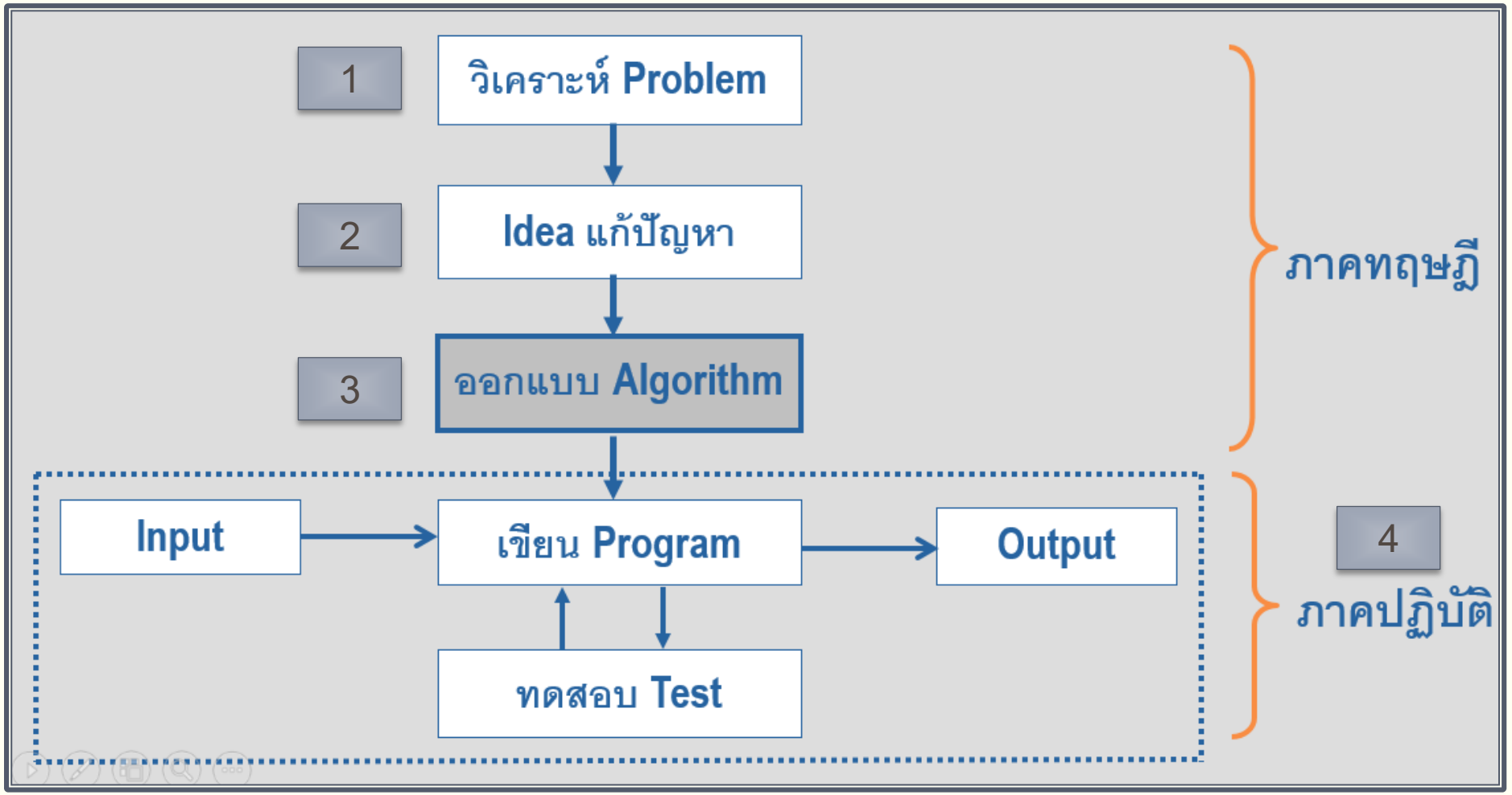
1.1 การแก้ปัญหาและขั้นตอนวิธี (Problems Solving and Algorithms)

Algorithm จะประกอบด้วย วิธีการเป็นขั้นๆ และมีส่วนที่ต้องทำงาน

- แบบวนซ้ำ (iterate)
- เรียกตัวเอง (recursive) โดยใช้ตรรกะ (logic) และ/หรือ ในการเปรียบเทียบ (comparison) ในขั้นตอนต่างๆ จนกระทั่งเสร็จสิ้นการทำงาน หรือ
- Algorithms ที่ต่างกันซึ่งมีจำนวนและชุดคำสั่งที่ใช้ต่างกัน

→ เวลา (Time), และขนาดหน่วยความจำ (space) ที่ต้องการต่างกัน หรือเรียกได้อีกอย่างว่ามีความซับซ้อน (complexity) ต่างกัน

1.1 การแก้ปัญหาและขั้นตอนวิธี (Problems Solving and Algorithms)



[ภาพโดย อ. จีรพร วีระพันธุ์]

1.1 การแก้ปัญหาและขั้นตอนวิธี (Problems Solving and Algorithms)

กระบวนการแก้ปัญหาด้วยคอมพิวเตอร์ ประกอบไปด้วย 4 ขั้นตอน

1. การวิเคราะห์ปัญหา (Problem Analysis) → เข้าใจปัญหา/ชัดเจน
2. หาแนวคิดในการแก้ปัญหา (Solution Idea) → มีวิธีใดบ้าง
3. การออกแบบและวิเคราะห์ขั้นตอนวิธีทางคอมพิวเตอร์ (Design and Analysis Computer Algorithm) → แบบวนซ้ำ (iterate)/เรียกตัวเอง (recursive)/ตรรกะ (logic)/การเปรียบเทียบ (comparison)
4. การพัฒนาและทดสอบโปรแกรม (Program Development and Testing)

1.1 การแก้ปัญหาและขั้นตอนวิธี (Problems Solving and Algorithms)

กระบวนการทำงาน (procedure) ใน Algorithm เพื่อใช้แก้ปัญหาให้บรรลุเป้าหมายนั้นจะต้อง

- มีขั้นตอนการดำเนินงานที่ชัดเจน
- กระบวนการทำงานควรไม่ซับซ้อน เพื่อที่จะได้สามารถนำไปปฏิบัติการโดยเครื่องคอมพิวเตอร์
- กระบวนการทำงานมีจุดจบ (finiteness)
- นิยมเขียนในรูปแบบของคำสั่งลำลอง (pseudo code)