

Computer Programming II การเขียนโปรแกรมคอมพิวเตอร์2 LECTURE#1

อ.สถิตย์ ประสมพันธ์

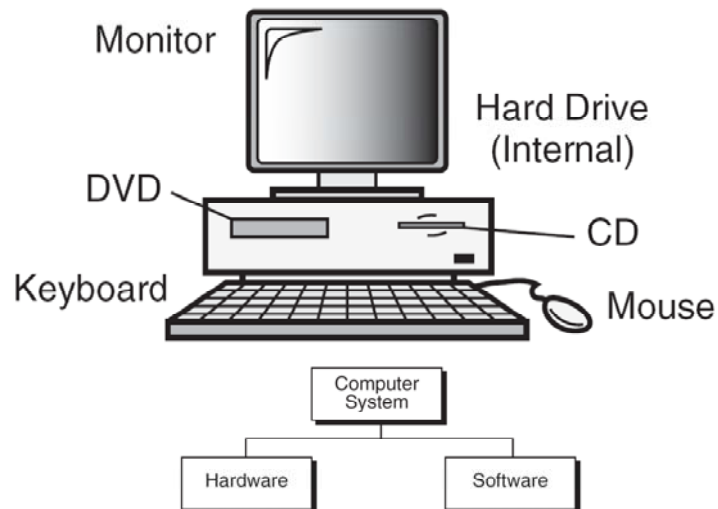
ภาควิชาวิทยาการคอมพิวเตอร์และสารสนเทศ

KMUTNB

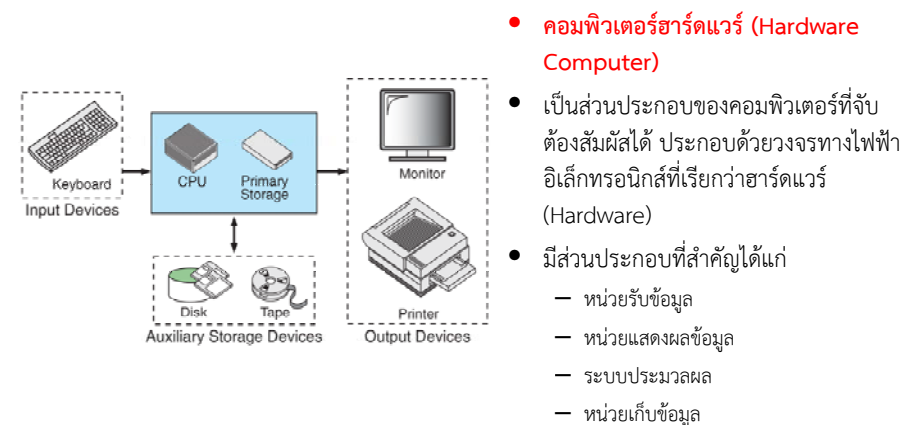
ความหมายของคอมพิวเตอร์

- คอมพิวเตอร์เป็นอุปกรณ์ทางไฟฟ้าชนิดหนึ่งที่สามารถประมวลผลและจำข้อมูลต่าง ๆ ได้ สามารถคิดคำนวณตัวเลข สามารถตอบสนองต่อการกระทำของผู้ใช้ได้และมีความสามารถในการเชื่อมต่อกับอุปกรณ์ไฟฟ้าบางชนิด เพื่อสั่งให้อุปกรณ์นั้นทำงานตามคำสั่งได้ ใช้สำหรับแก้ปัญหาต่าง ๆ ทั้งที่ง่ายและซับซ้อนโดยวิธีทางคณิตศาสตร์

องค์ประกอบของระบบคอมพิวเตอร์



องค์ประกอบของระบบคอมพิวเตอร์



องค์ประกอบของระบบคอมพิวเตอร์

- 1.หน่วยรับข้อมูล(Input Unit)
 - เป็นส่วนที่ใช้รับข้อมูลและคำสั่งจากภายนอกเข้าสู่เครื่องคอมพิวเตอร์เพื่อนำไปประมวลผล
 - อุปกรณ์อินพุตจะเปลี่ยนข้อมูลที่มนุษย์เข้าใจเปลี่ยนเป็นรหัสข้อมูลที่เครื่องคอมพิวเตอร์เข้าใจ
 - อุปกรณ์เหล่านี้จะทำงานได้ ข้อมูลคำสั่งจะต้องถูกเก็บไว้บนสื่อ(Input media) ที่อุปกรณ์นั้น ๆ รู้จักเรียกว่า Input Device
- 2. หน่วยแสดงผลหรือเอาต์พุต(Output Unit)
 - เป็นส่วนที่ใช้แสดงผลลัพธ์จากการประมวลผลออกมาในรูปแบบต่าง ๆ ที่มนุษย์เข้าใจ

องค์ประกอบของระบบคอมพิวเตอร์

- 3. หน่วยประมวลผลกลาง(Central Processing Unit)
 - มีหน้าที่เก็บข้อมูลคำสั่งทำการประมวลผลทางคณิตศาสตร์ เปรียบเทียบข้อมูล เมื่อข้อมูลเข้าสู่ระบบแล้วหน่วยประมวลผลจะทำหน้าที่ประมวลผลตามคำสั่ง หรือโปรแกรมที่กำหนดไว้ โดยโปรแกรมและข้อมูลต่าง ๆ จะถูกเก็บเอาไว้ในหน่วยความจำ เมื่อหน่วยประมวลผลทำงานสำเร็จแล้วจะเก็บข้อมูลลงหน่วยเก็บข้อมูลหรือส่งผลลัพธ์ที่ได้ออกทางหน่วยแสดงผลต่อไป
 - หน่วยประมวลผลกลาง มีหน้าที่ 2 อย่างคือ
 - 1. ทำหน้าที่ประสานการทำงานในระบบคอมพิวเตอร์
 - 2. ทำหน้าที่ประมวลผลทางคณิตศาสตร์และตรรกะของข้อมูล
 - หน่วยประมวลผลกลางแบ่งหน่วยการทำงานออกเป็น 3 หน่วยหลัก คือ
 - หน่วยควบคุม(Control Unit)
 - หน่วยคำนวณและตรรกะ(Arithmetic and Logic Unit)
 - หน่วยความจำหลัก(Main Memory หรือ Primary Storage)

องค์ประกอบของระบบคอมพิวเตอร์

- การวัดขนาดของหน่วยความจำหลัก จะวัดจากจำนวนข้อมูลที่เก็บโดยมีหน่วยของการวัดดังนี้
 - 1 KB(Kilo Byte)=1024 Bytes
 - 1 MB(Mega Byte) =1024 K Bytes
 - 1 GB(Giga Byte) = 1024 M Bytes
 - 1 TB(Tera Byte) = 1024 G Bytes

องค์ประกอบของระบบคอมพิวเตอร์

- 4.หน่วยความจำสำรอง (Auxiliary Memory หรือ Secondary storage)
 - เป็นหน่วยความจำที่อยู่นอกเครื่องคอมพิวเตอร์ มีหน้าที่ช่วยให้อหน่วยความจำหลักทำงานได้มากขึ้น โดยจะเก็บข้อมูลที่รอการประมวลผล และข้อมูลที่ประมวลผลเสร็จแล้ว อุปกรณ์ที่นำมาใช้เป็นหน่วยความจำรอง เช่น Hard Disk Drive เป็นต้น

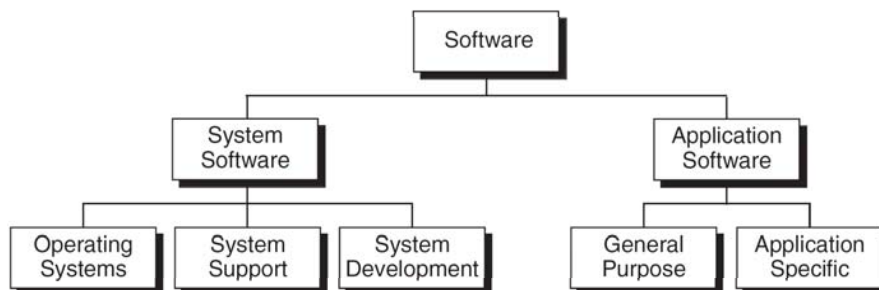
คอมพิวเตอร์ซอฟต์แวร์ (Software Computer)

- Software คือ ชุดคำสั่งที่มีไว้เพื่อทำงานอย่างใดอย่างหนึ่ง สามารถแบ่งซอฟต์แวร์ตามการทำงานได้ 2 ประเภท คือ
- ซอฟต์แวร์ระบบ (System Software)
 - เป็นซอฟต์แวร์ที่ทำหน้าที่ควบคุมการทำงานของเครื่องเพื่อให้สามารถทำงานต่าง ๆ ได้สะดวก แบ่งออกเป็น
 - โปรแกรมระบบปฏิบัติการ(OS: Operating System)
 - โปรแกรมอรรถประโยชน์ (Utilities Program)
 - โปรแกรมดีไวส์ไดรเวอร์ (Device Driver)

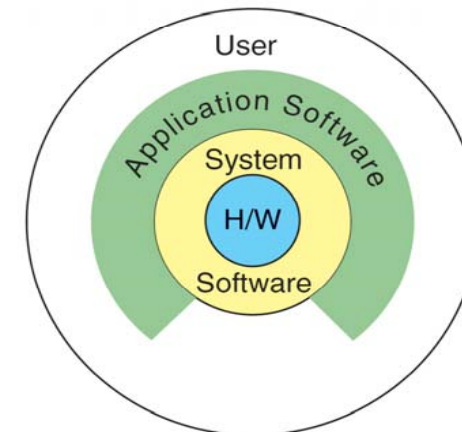
คอมพิวเตอร์ซอฟต์แวร์ (Software Computer)

- ซอฟต์แวร์ประยุกต์ (Application Software)
 - เป็นโปรแกรมที่พัฒนาขึ้นสำหรับงานเฉพาะต่างๆ อาจเป็นโปรแกรมที่เขียนขึ้นเองหรือโปรแกรมที่มีอยู่ทั่วไป Application Software ผลิตขึ้นมาเพื่อให้ผู้ใช้ใช้งานเฉพาะทาง เช่น ซอฟต์แวร์ประมวลผลคำ ซอฟต์แวร์ตารางจัดการ ประเภทของโปรแกรมประยุกต์ที่มองเห็นทั่ว ๆ ไปมีดังนี้
 - ซอฟต์แวร์สำเร็จรูป (Package Software)
 - ซอฟต์แวร์เฉพาะ(Custom Software)
 - ซอฟต์แวร์แบบเปิด (Open Source Software)
 - แชร์แวร์ (Shareware)
 - ซอฟต์แวร์ฟรี(Freeware)

คอมพิวเตอร์ซอฟต์แวร์ (Software Computer)



ความสัมพันธ์ระหว่างคอมพิวเตอร์ฮาร์ดแวร์และคอมพิวเตอร์ซอฟต์แวร์



ภาษาคอมพิวเตอร์(Computer Language)

- ภาษาเครื่อง (Machine Language)
- เป็นภาษาเครื่องเข้าใจคำสั่งได้เลย มีลักษณะคำสั่งเป็นตัวเลขล้วน เป็นภาษาคอมพิวเตอร์ระดับต่ำที่สุด เพราะเป็นตัวเลขฐานสอง แทนข้อมูลและคำสั่งต่างๆ ทั้งหมด

```
1 00000000 0000100 0000000000000000
2 01011110 00001100 11000010 000000000000010
3 11101111 00010110 00000000000000101
4 11101111 10011110 00000000000001011
5 11111000 10101101 11011111 0000000000010010
6 01100010 11011111 0000000000010101
7 11101111 00000010 11110111 0000000000010111
8 11101000 10101101 11011111 0000000000011110
9 00000011 10100010 11011111 000000000100001
10 11101111 00000010 11110111 0000000000100100
11 01111110 11110100 10101101
12 11111000 10101110 11000101 0000000000101011
13 00000110 10100010 11110111 0000000000100001
14 11101111 00000010 11110111 0000000000110100
15 01010000 11010100 0000000000111011
16 00000100 0000000000111101
```

ภาษาคอมพิวเตอร์(Computer Language)

- ภาษาระดับต่ำ(Low level Language)
- เป็นภาษาที่มีลักษณะใกล้เคียงกับภาษาเครื่อง เพียงแต่มีการใช้สัญลักษณ์หรือตัวอักษรมาแทนคำสั่งในส่วนต่าง ๆ ตัวอย่างของภาษานี้ได้แก่ ภาษาแอสเซมบลี(Assembly) เป็นภาษาคอมพิวเตอร์ที่พัฒนาขึ้นมาเพื่อให้ผู้เขียนโปรแกรมสามารถเขียนโปรแกรมติดต่อกับคอมพิวเตอร์ได้ง่ายกว่าภาษาเครื่อง โดยใช้คำย่อภาษาอังกฤษในการเขียนคำสั่ง

```
1 entry main,"mcr2>
2 sub12 #12,sp
3 jsb C$MAIN_ARGS
4 movab $CHAR_STRING_CON
5
6 pushal ~8(fp)
7 pushal (r2)
8 calli #2,SCANF
9 pushal ~12(fp)
10 pushal 3(r2)
11 calli #2,SCANF
12 mull3 ~8(fp),~12(fp),-
13 pusha 6(r2)
14 calli #2,PRINTF
15 clrl r0
16 ret
```

ภาษาคอมพิวเตอร์(Computer Language)

- ภาษาระดับสูง(High level Language)
- เป็นภาษาที่มีลักษณะใกล้เคียงกับภาษาอังกฤษที่มนุษย์ใช้กันอยู่ เรียนรู้ง่าย เข้าใจได้ง่าย สะดวกในการใช้งาน และใช้ได้กับทุกเครื่อง ตัวอย่างของภาษาระดับสูงได้แก่ ภาษาโคบอล(COBOL) ภาษาปาสคาล (PASCAL) ภาษาซี(C) ภาษาจาวา (JAVA) เป็นต้น

```
1 /* This program reads two integers from the keyboard
2 and prints their product.
3 Written by:
4 Date:
5 */
6 #include <stdio.h>
7
8 int main (void)
9 {
10 // Local Definitions
11 int number1;
12 int number2;
13 int result;
14
15 // Statements
16 scanf ("%d", &number1);
17 scanf ("%d", &number2);
18 result = number1 * number2;
19 printf ("%d", result);
20 return 0;
21 } // main
```

ขั้นตอนการพัฒนาโปรแกรม

- 1. การวิเคราะห์ปัญหา (Problem Analysis)
- 2. เขียนผังงาน (Pseudo Coding)
- 3. เขียนโปรแกรม (Programming)
- 4. ทดสอบและแก้ไขโปรแกรม (Program testing and Debugging)
- 5. ทำเอกสารและบำรุงรักษาโปรแกรม (Program document and Maintenance)

การวิเคราะห์ปัญหา (Problem Analysis)

- เราสามารถวิเคราะห์ปัญหาโดยการวิเคราะห์ส่วนต่าง ๆ ดังนี้
 - วิเคราะห์ข้อมูลนำเข้า (Input Analysis)
 - วิเคราะห์ขั้นตอนการทำงาน (Process Analysis)
 - วิเคราะห์ผลลัพธ์ (Output Analysis)

ตัวอย่างการวิเคราะห์ปัญหา#1

- จงเขียนแนวทางการแก้ปัญหาด้วยคอมพิวเตอร์สำหรับให้คอมพิวเตอร์คำนวณค่าจ้างพนักงานเป็นรายชั่วโมง จากนั้นแสดงค่าจ้างที่คำนวณได้
- **ต้องการอะไร?** ต้องการทราบค่าจ้างของพนักงานแต่ละคน
- **ต้องการผลลัพธ์อย่างไร(Output)** ต้องการผลลัพธ์เป็นค่าจ้างสุทธิของพนักงานทางจอภาพ
- **ข้อมูลเข้า(Input)** รหัสพนักงาน ชื่อพนักงาน จำนวนชั่วโมงทำงาน เก็บในตัวแปรชื่อ Hours ค่าจ้างรายชั่วโมงเก็บในตัวแปรชื่อ PayRate

ตัวอย่างการวิเคราะห์ปัญหา#1

- **วิธีการประมวลผล(Process)**
- กำหนดวิธีการคำนวณ
 - ค่าจ้างสุทธิ = จำนวนชั่วโมง x อัตราต่อชั่วโมง
- ขั้นตอนการประมวลผล
 - 1. เริ่มต้น
 - 2. รับรหัสพนักงาน ชื่อพนักงาน จำนวนชั่วโมงทำงานเก็บในตัวแปรชื่อ Hours ค่าจ้างรายชั่วโมงเก็บในตัวแปรชื่อ PayRate
 - 3. คำนวณ ค่าจ้างสุทธิ = Hours x PayRate
 - 4. แสดงผลลัพธ์เป็นรหัสพนักงาน ชื่อ ค่าจ้างสุทธิของพนักงานทางจอภาพ
 - 5. จบการทำงาน

ตัวอย่างการวิเคราะห์ปัญหา#2

- จงเขียนโปรแกรมเพื่อรายงานผลสอบของนักศึกษาวิชาคอมพิวเตอร์ โดยให้แสดงคะแนนรวมและเกรดออกมา
- **ต้องการอะไร?** ต้องการพิมพ์คะแนนผลสอบและเกรดของนักศึกษา
- **ต้องการผลลัพธ์อย่างไร(Output)** ต้องการผลลัพธ์เป็นคะแนนรวมและเกรดของนักศึกษาแต่ละคน
- **ข้อมูลเข้า(Input)** รหัสประจำตัวนักศึกษา(ID) ชื่อนักศึกษา(name) คะแนนสอบกลางภาค(mid) คะแนนสอบย่อย(test) คะแนนสอบปลายภาค(final)

ตัวอย่างการวิเคราะห์ปัญหา#2

• วิธีการประมวลผล(Process)

• กำหนดวิธีการคำนวณ

- คะแนนรวม= คะแนนกลางภาค+คะแนนสอบย่อย+คะแนนปลายภาค
- ถ้าคะแนนรวม ≥ 80 ได้เกรด “A”
- ถ้าคะแนนรวม ≥ 70 และ < 80 ได้เกรด “B”
- ถ้าคะแนนรวม ≥ 60 และ < 70 ได้เกรด “C”
- ถ้าคะแนนรวม ≥ 50 และ < 60 ได้เกรด “D”
- ถ้าคะแนนรวม < 50 ได้เกรด “F”

ตัวอย่างการวิเคราะห์ปัญหา#2

• ขั้นตอนการประมวลผล









- 1. เริ่มต้น
- 2. รับค่าตัวแปร ID name mid test final
- 3. คำนวณคะแนนรวมและเกรด
 - Total= mid + test + final
 - ถ้า Total ≥ 80 , Grade= “A”
 - ถ้า Total ≥ 70 และ < 80 , Grade= “B”
 - ถ้า Total ≥ 60 และ < 70 , Grade= “C”
 - ถ้า Total ≥ 50 และ < 60 , Grade= “D”
 - ถ้า Total < 50 , Grade= “F”
- 4. แสดง ID name Total Grade ของนักศึกษา
- 5. กลับไปข้อ 2 เพื่อรับจนครบทุกคน ถ้าครบแล้วไปข้อ 6
- 6. หยุดการทำงาน

การเขียนผังงานของโปรแกรม

• การเขียนผังงานที่ดี

- เขียนตามสัญลักษณ์ที่กำหนด
- ใช้ลูกศรแสดงทิศทางการทำงานจากบนลงล่าง
- อธิบายสั้น ๆ ให้เข้าใจง่าย
- ทุกแผนภาพต้องมีทิศทางเข้าออก
- ไม่ควรโยงลูกศรไปที่ไกล ๆ มาก ถ้าต้องทำให้ใช้สัญลักษณ์ของการเชื่อมต่อแทน

การเขียนผังงานของโปรแกรม

สัญลักษณ์	ความหมาย
	จุดเริ่มต้น หรือ สิ้นสุด
	รับข้อมูล (Input) แสดงข้อมูล (Output)
	การคำนวณ (Process)
	การตัดสินใจ (Decision) การเปรียบเทียบ (Compare)
	การส่งออกทางเครื่องพิมพ์ (Printer)
	การทำงานย่อย (SubProgram)
	จุดเชื่อมต่อ (Connection)
	ทิศทาง (Flow)

การเขียนอัลกอริทึมของโปรแกรม

- อัลกอริทึม(Algorithms) หมายถึงลำดับขั้นตอนเชิงคำนวณที่แปลงข้อมูลด้านอินพุตของปัญหาไปเป็นผลลัพธ์ที่ต้องการ
- ขั้นตอนต่าง ๆ ในอัลกอริทึมสามารถเปลี่ยนไปเป็นคำสั่งที่ให้คอมพิวเตอร์ทำงานได้
- ถ้าหากทำตามอัลกอริทึมแล้ว ปัญหาจะต้องถูกแก้ได้สำเร็จและได้คำตอบที่ถูกต้องสำหรับทุกกรณีตามที่กำหนดในอัลกอริทึม
- ดังนั้นเราจะไม่ยอมรับอัลกอริทึมที่ทำงานติดอยู่ใน Loop ไม่มีที่สิ้นสุดหรืออัลกอริทึมที่ทำงานแล้วได้คำตอบถูกบ้างผิดบ้าง

ตัวอย่างการเขียนอัลกอริทึมของโปรแกรม#1

- จงวิเคราะห์ปัญหาและเขียนอัลกอริทึมสำหรับหาค่าเฉลี่ยของอุณหภูมิประจำวันโดยรับค่าอุณหภูมิสูงสุดและอุณหภูมิต่ำสุดเป็นเลขจำนวนเต็มเข้าไปและให้แสดงค่าอุณหภูมิเฉลี่ยออกทางจอภาพ
- **ต้องการอะไร?** ต้องการทราบค่าเฉลี่ยของอุณหภูมิประจำวัน
- **ต้องการผลลัพธ์อย่างไร(Output)** ต้องการผลลัพธ์เป็นค่าเฉลี่ยของอุณหภูมิประจำวันโดยใช้ตัวแปรชื่อ avg_temp
- **ข้อมูลเข้า(Input)** รับค่าอุณหภูมิสูงสุดอยู่ในตัวแปรชื่อ max_temp
อุณหภูมิต่ำสุดอยู่ในตัวแปรชื่อ min_temp

ตัวอย่างการเขียนอัลกอริทึมของโปรแกรม#1

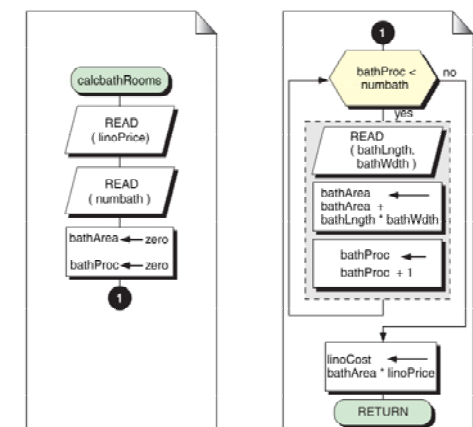
- **วิธีการประมวลผล(Process)**
 - 1. รับค่าอุณหภูมิสูงสุดและอุณหภูมิต่ำสุด
 - 2. หาค่าเฉลี่ยโดยใช้ $\text{avg_temp} = (\text{max_temp} + \text{min_temp}) / 2$
 - 3. แสดงค่า avg_temp ทางจอภาพ
- **อัลกอริทึม(Algorithms)**
Find_avg_temperature
 1. READ max_temp, min_temp
 2. $\text{avg_temp} = (\text{max_temp} + \text{min_temp}) / 2$
 3. Output avg_temp to the screen

End

การเขียนอัลกอริทึมของโปรแกรม

Pseudocode for Calculate Bathrooms

```
Algorithm Calculate BathRooms
1 prompt user and read linoleum price
2 prompt user and read number of bathrooms
3 set total bath area and baths processed to zero
4 while ( baths processed < number of bathrooms )
  1 prompt user and read bath length and width
  2 total bath area =
  3   total bath area + bath length * bath width
  4 add 1 to baths processed
5 bath cost = total bath area * linoleum price
6 return bath cost
end Algorithm Calculate BathRooms
```



ตัวอย่างการวิเคราะห์ปัญหาและการเขียนโปรแกรมเพื่อช่วยแก้ปัญหา

- จงเขียนผังงานและโปรแกรมเพื่อรับข้อมูลตัวเลขจำนวนจริงความยาวฐาน (base) และความสูง (height) ของรูปสามเหลี่ยม แล้วให้ทำการคำนวณพื้นที่และแสดงผลในรูปแบบต่อไปนี้
- Enter base value: **10** (กดแป้น Enter)
- Enter height value: **5** (กดแป้น Enter)
- Area is : 25.000

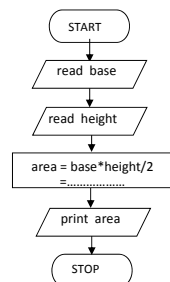
ตัวอย่างการวิเคราะห์ปัญหาและการเขียนโปรแกรมเพื่อช่วยแก้ปัญหา

- **ข้อมูลนำเข้า** ความยาวฐาน และความสูง
- **แสดงผล** พื้นที่
- **กำหนดตัวแปร**

ชื่อตัวแปร	ความหมาย
base	ความยาวฐานของรูปสามเหลี่ยม
height	ความสูงของรูปสามเหลี่ยม
area	พื้นที่ของรูปสามเหลี่ยม

ตัวอย่างการวิเคราะห์ปัญหาและการเขียนโปรแกรมเพื่อช่วยแก้ปัญหา

ขั้นตอนการทำงาน



เขียนโปรแกรม

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    float base, height, area;
    printf("Enter base value: "); /* prompt to input base */
    scanf("%f", &base);          /* input base */
    printf("Enter height value: "); /* prompt to input height */
    scanf("%f", &height);         /* input height */
    area = base*height/2;         /* compute area */
    printf("Area = %7.2f\n", area); /* display result */
    system("PAUSE");
    return 0;
}
```

การเขียนโปรแกรม

- เป็นขั้นตอนของการเขียนโปรแกรม เพื่อให้คอมพิวเตอร์สามารถประมวลผลได้
- การเขียนโปรแกรมจะต้องเขียนตามภาษาที่คอมพิวเตอร์เข้าใจ
- จะใช้ภาษาระดับใดก็ได้ ซึ่งจะต้องเขียนให้ถูกต้องตามหลักไวยากรณ์ (Syntax) ของภาษานั้น ๆ

การทดสอบและแก้ไขโปรแกรม

- หลังจากการเขียนโปรแกรมจะต้องทดสอบความถูกต้องของโปรแกรมที่เขียนขึ้นว่ามีข้อผิดพลาดหรือไม่ ซึ่งเรียกว่า ดีบั๊ก (Debug) ซึ่งโดยทั่วไปข้อผิดพลาด (Bug) มี 2 ประเภท คือ
- **1. Syntax error** คือ การเขียนคำสั่งไม่ถูกต้องตามหลักการเขียนโปรแกรมของภาษานั้น ๆ กรณีที่เกิด Syntax error โปรแกรมจะไม่สามารถทำงานได้
- **2. Logic error** เป็นข้อผิดพลาดทางตรรกะ โปรแกรมสามารถทำงานได้แต่ผลลัพธ์จะไม่ถูกต้อง

การทำเอกสารและบำรุงรักษาโปรแกรม

- 1. **คู่มือการใช้** หรือ User Document หรือ User Guide ซึ่งจะเป็นส่วนที่อธิบายการใช้โปรแกรม
- 2. **คู่มือโปรแกรมเมอร์** หรือ Program Document หรือ Technical Document ซึ่งจะทำให้มีความสะดวกในการแก้ไข และพัฒนาโปรแกรมต่อไปในอนาคต