

# Hardware, Software, Flowchart

Good students never miss a class !!!

# เนื้อหา

- อุปกรณ์ Hardware
- ประเภทของ Software
- ขั้นตอนการพัฒนาโปรแกรม
- การวิเคราะห์โจทย์ปัญหา
- การเขียนผังงาน
- การ Compile โปรแกรมภาษา C

Don't waste time, rereading 

# 1.1 อุปกรณ์ Hardware

● **คอมพิวเตอร์** หมายถึงเครื่องมือทางอิเล็กทรอนิกส์ที่ใช้ในการประมวลผลข้อมูล

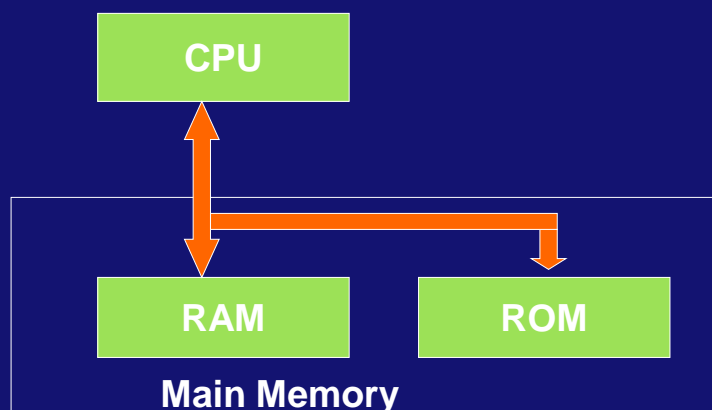
● **ส่วนประกอบหลักของคอมพิวเตอร์**

1. หน่วยประมวลผลกลาง (CPU)
2. หน่วยความจำหลัก (Main Memory)
3. หน่วยความจำสำรอง (Secondary Storages)
4. หน่วยรับและแสดงผล (Input/Output)

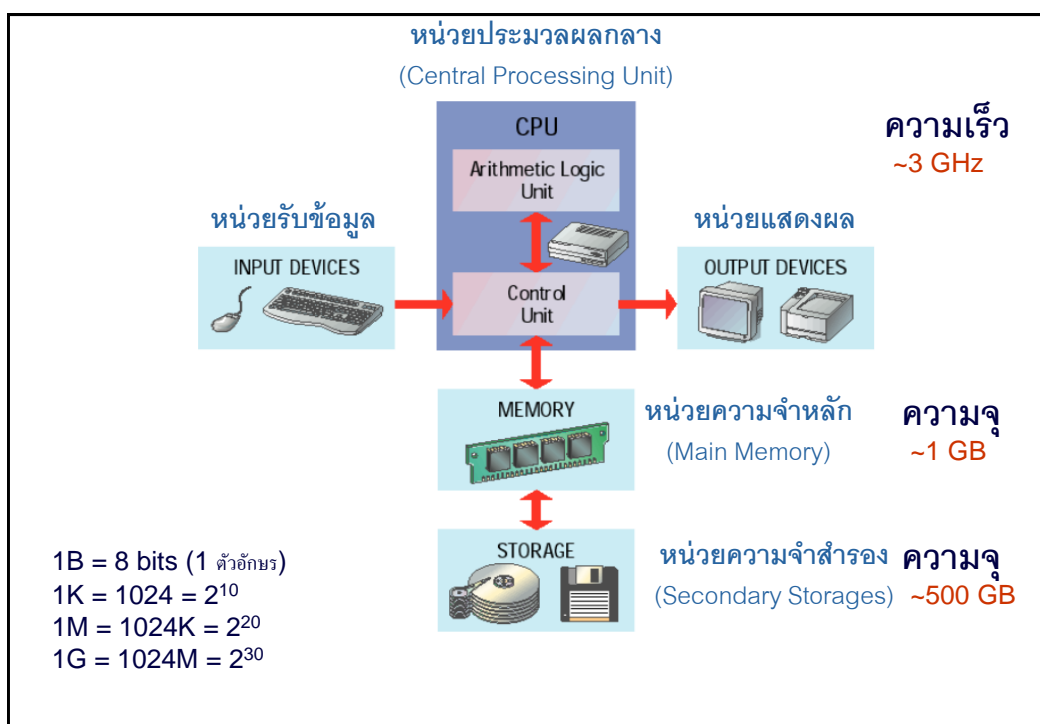
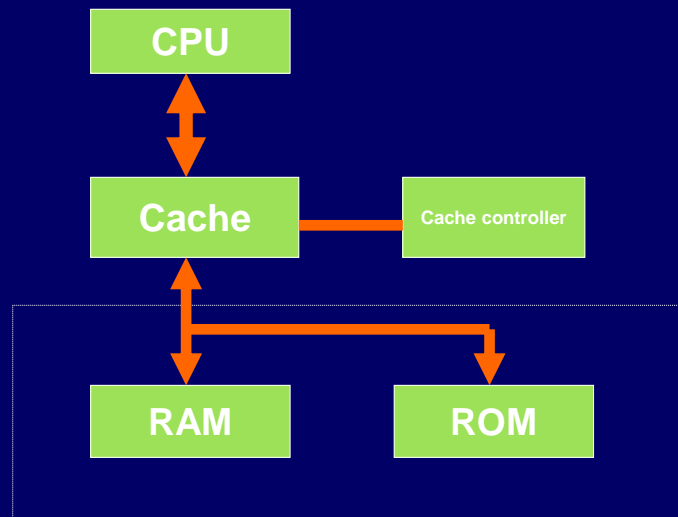


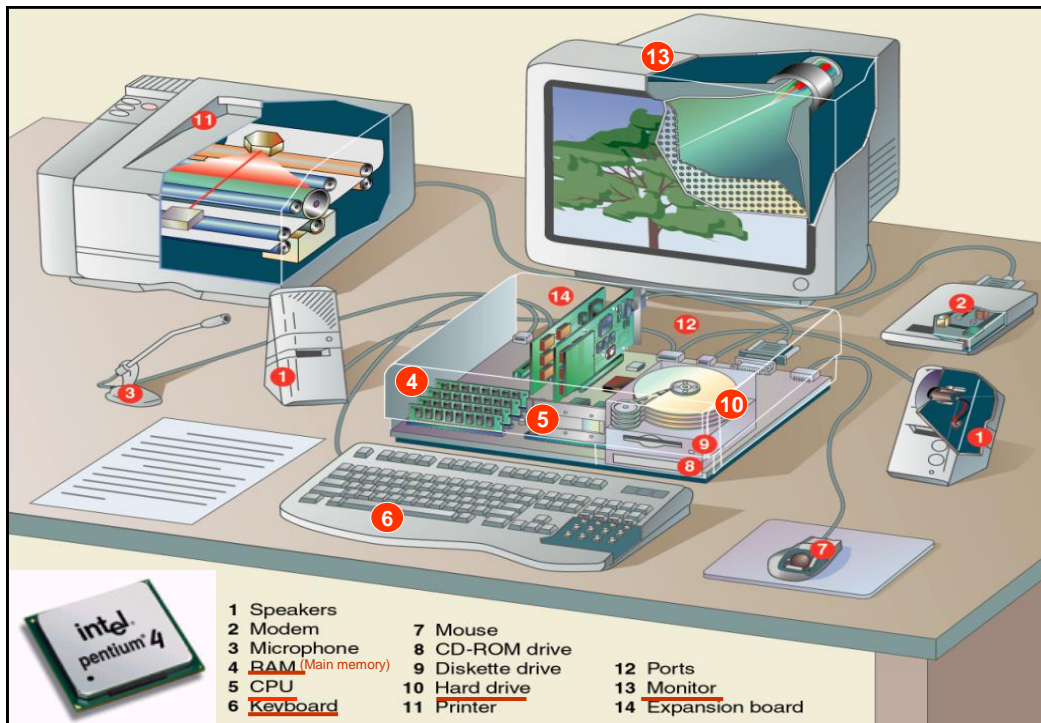
3

## โครงสร้างบนเมนบอร์ด



# โครงสร้างบนเมนบอร์ด





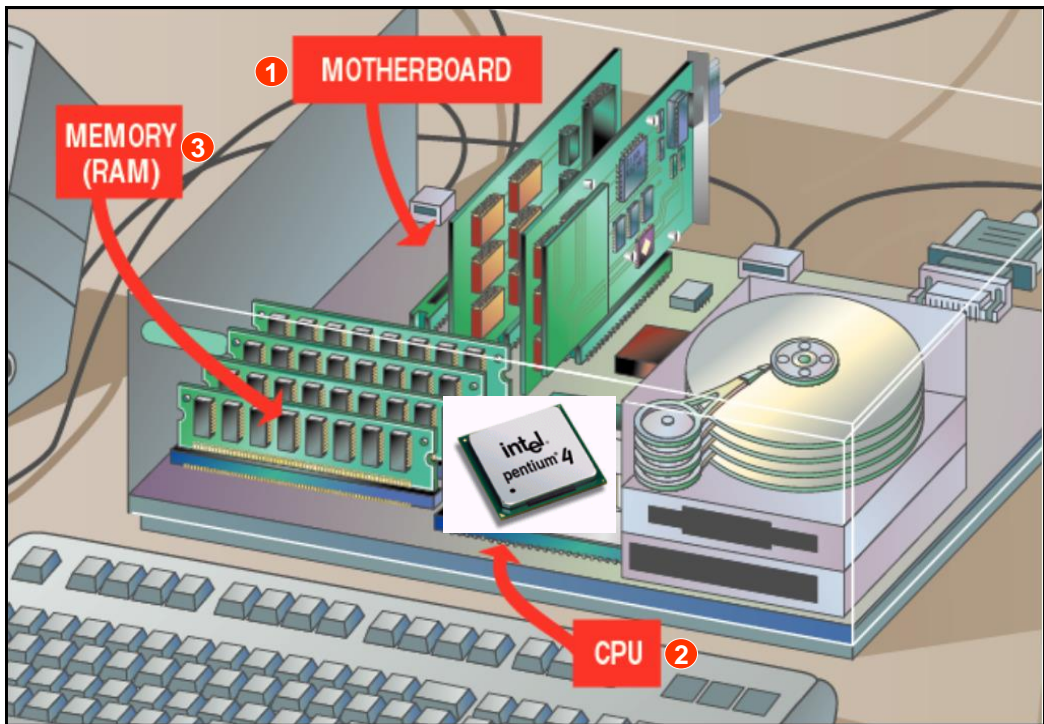
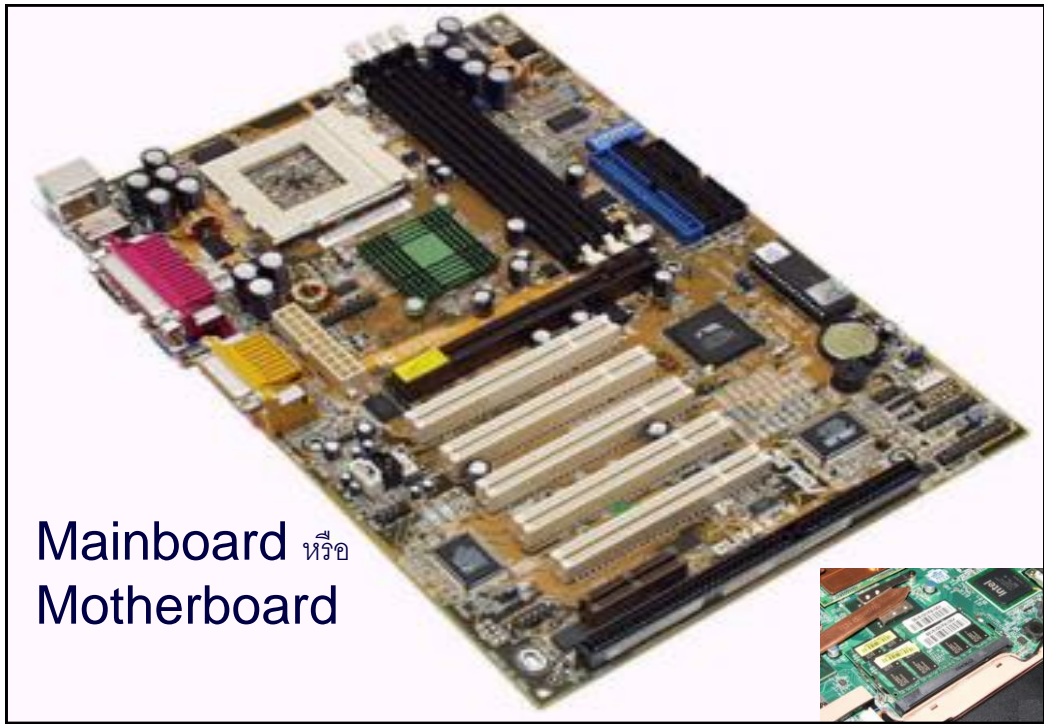
## 1.1.1 CPU

### ● หน่วยประมวลผลกลาง (Central Processing Unit)

- CPU ทำหน้าที่หลักในการประมวลผลข้อมูล และควบคุมการทำงานทั้งหมดของระบบ
- เป็นเสมือนสมองของคอมพิวเตอร์ในส่วนคำนวณ

### ● หน่วยประมวลผล (CPU) ใน PC

- เป็นหน่วยประมวลผลขนาดเล็กเรียกว่า **Microprocessor** วางอยู่บน **แผงวงจรหลัก (Main Board)**
- ซึ่งประกอบด้วยวงจร ที่เชื่อมต่อระหว่างหน่วยประมวลผล (CPU) กับหน่วยอื่นๆ (Memory, Disk, I/O)





## 1.1.2 หน่วยความจำหลัก

### ● หน่วยความจำหลัก (Main Memory)

- เป็นที่เก็บโปรแกรมและข้อมูลที่อยู่ในระหว่างการประมวลผล
- โดยเก็บใน **RAM** (**R**andom **A**ccess **M**emory) แบบชั่วคราว
- ขนาดของ RAM (0.5-2 GB) แสดงถึงประสิทธิภาพของคอมพิวเตอร์

### ● ความจุของหน่วยความจำ มีหน่วยวัด ดังนี้

- 1 kBs (Kilobytes) =  $2^{10}$  = 1024 Bytes
- 1 MBs (Megabytes) =  $2^{20}$  = 1024 KBs (= 1,048,576 Bytes)
- 1 GBs (Gigabytes) =  $2^{30}$  = 1024 MBs
- 1 TBs (Terabytes) =  $2^{40}$  = 1024 GBs

11

## 1.1.3 หน่วยความจำสำรอง

### ● หน่วยความจำสำรอง (Secondary Storages)

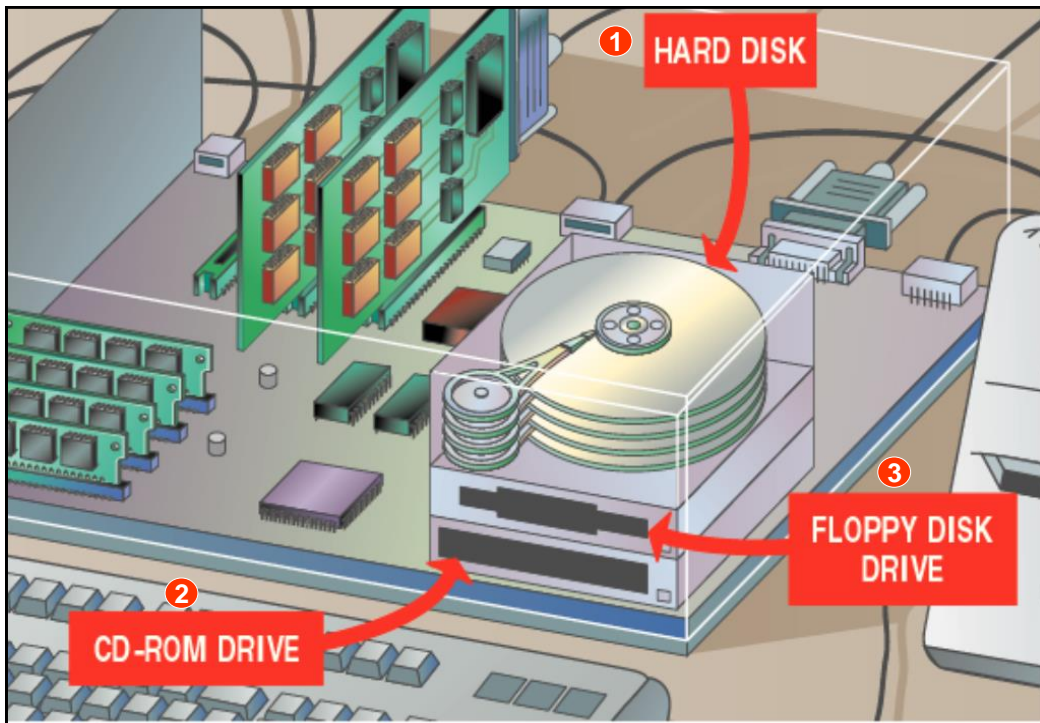
- เก็บโปรแกรมและข้อมูลแบบถาวร
- ทั้งที่กำลังประมวลผลและยังไม่ถูกประมวลผลในขณะนั้น

### ● ชนิดของอุปกรณ์ ที่ใช้เป็นหน่วยความจำสำรอง

- Hard drive
- Floppy disk drive
- CD/DVD-ROM drive, ...



12



## 1.1.4 หน่วยรับ/แสดงผล

### ● หน่วยรับ/แสดงผล (Input/Output)

### ● อุปกรณ์รับข้อมูล (Input Devices)

- เป็นเครื่องมือในการรับข้อมูล และคำสั่งจากผู้ใช้
- เช่น Keyboard, Mouse, Microphone

### ● อุปกรณ์แสดงผลข้อมูล (Output Devices)

- เป็นเครื่องมือในการส่งผลการทำงานกลับมายังผู้ใช้
- เช่น จอภาพ เครื่องพิมพ์ ลำโพง



## 1.2 ประเภทของ Software

● **Software** คือโปรแกรมหรือชุดคำสั่ง ที่ผู้เขียนสร้างขึ้น เพื่อให้คอมพิวเตอร์ทำงานอย่างเป็นขั้นตอน และได้ผลลัพธ์ตามที่ต้องการ

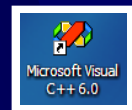
● **Software** แบ่งได้ 2 ประเภท

### 1. โปรแกรมระบบ (Operating System: OS)

- เช่น DOS, Windows, UNIX, ...

### 2. โปรแกรมประยุกต์ (Application Program)

- **Software package** เช่น Word, Spreadsheet, ...
- **Developed program** เช่น โปรแกรมใช้งานเฉพาะ ถูกพัฒนาโดยใช้โปรแกรมภาษา (Programming language เช่น Pascal, C, ...)



15

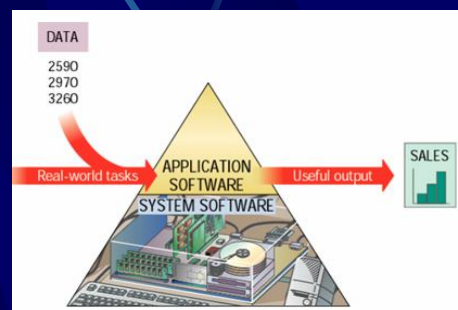
## 1.2.1 OS

● **โปรแกรม OS** ทำหน้าที่ในการควบคุมการทำงานของระบบคอมพิวเตอร์ เพื่อให้อุปกรณ์ต่างๆทำงานร่วมกันอย่างมีประสิทธิภาพ ตั้งแต่เริ่มเปิดเครื่อง

- เป็นโปรแกรมสื่อกลางระหว่างอุปกรณ์ **Hardware** และโปรแกรมประยุกต์ **Software**

● **OS ที่นิยมใช้** เช่น

- **Windows** (สำหรับ PCs), **UNIX** (สำหรับ Minicomputers)



16



## 1.2.2 โปรแกรมภาษา

- **โปรแกรมภาษา** ใช้ในการพัฒนาโปรแกรมประยุกต์สำหรับงานเฉพาะตามที่ใช้ต้องการ

- **ประเภทของโปรแกรมภาษา**

1. **ภาษาระดับต่ำ** (Low-Level Language)  
เช่น ภาษาเครื่อง (Machine language)
2. **ภาษาระดับกลาง** (Middle-Level Language)  
เช่น ภาษาแอสเซมบลี (Assembly Language)
3. **ภาษาระดับสูง** (High-Level Language)  
เช่น Pascal, Fortran, C, JAVA, ...

17

### 1.2.2.1 ภาษาเครื่อง

- **ภาษาเครื่อง** (Machine Language)

เป็นภาษาที่คอมพิวเตอร์เข้าใจ ซึ่งเขียนเป็นรหัสเลขฐาน 2 (0/1) และคำสั่งมีความเกี่ยวข้องกับอุปกรณ์ของคอมพิวเตอร์โดยตรง

- แต่มนุษย์เข้าใจ**ภาษาเครื่อง**ได้ยาก ดังนั้นการเขียนโปรแกรมด้วยภาษาเครื่องจึงยากมาก

18

00010100101101010101010101010101010100010  
111011010101010101010101110010100010110  
0010100101010010111101011101011101010  
1001010010110101010101010101010101010110  
0110100100110010111101011101010100010  
0001000101011101010101000101010111010  
1010100101010010101101011101011101011  
000101001011010101010101010101010100010

## Object code

### หุ่นยนต์ของฉัน

00	เดินหน้า
01	ยกมือขึ้น
10	ส่งเสียง
11	ถอยหลัง

นำมาทำเป็นโปรแกรม



00  
00  
01  
10

## 1.2.2.2 ภาษาแอสเซมบลี

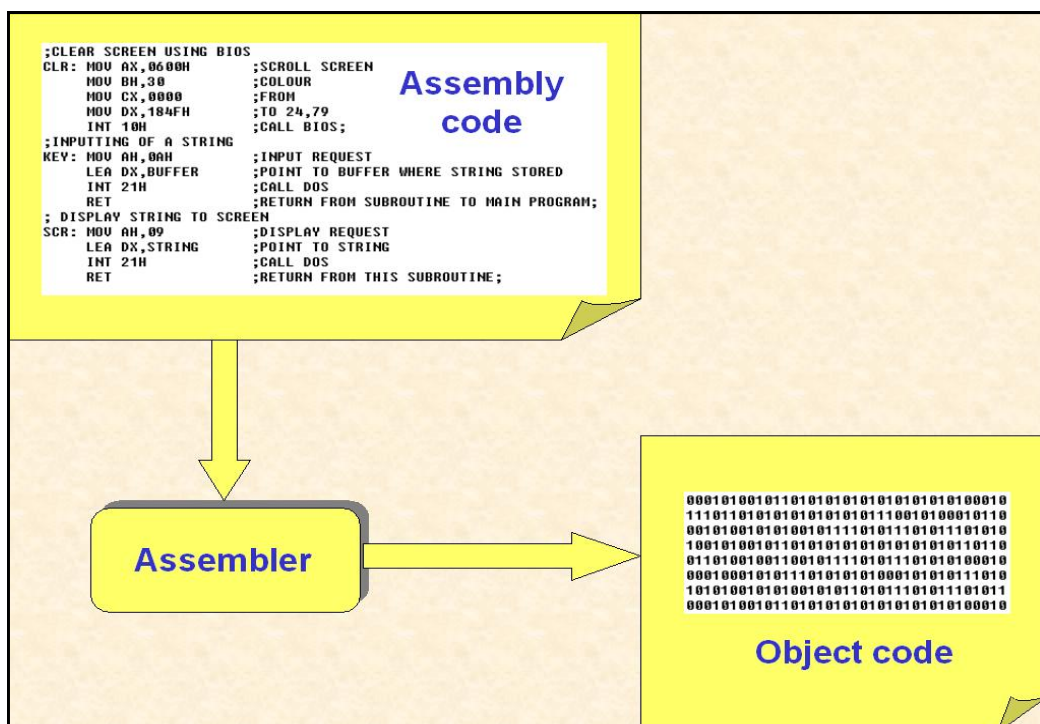
### ● ภาษาแอสเซมบลี (Assembly Language)

เป็นภาษาที่เขียนโดยใช้คำสั่งที่มนุษย์เข้าใจ (English-like statements) แทนการใช้รหัสเลขฐาน 2

- แต่ออกแบบมาเฉพาะสำหรับคอมพิวเตอร์แต่ละแบบ
- และผู้เขียนโปรแกรมยังต้องทราบข้อมูลที่เกี่ยวข้องกับอุปกรณ์ของคอมพิวเตอร์

● ใช้ **แอสเซมเบอ์ (Assembler)** ในการแปลภาษาแอสเซมบลี ให้เป็นภาษาเครื่อง

21



### 1.2.2.3 ภาษาระดับสูง

## ● ภาษาระดับสูง (High-Level Languages)

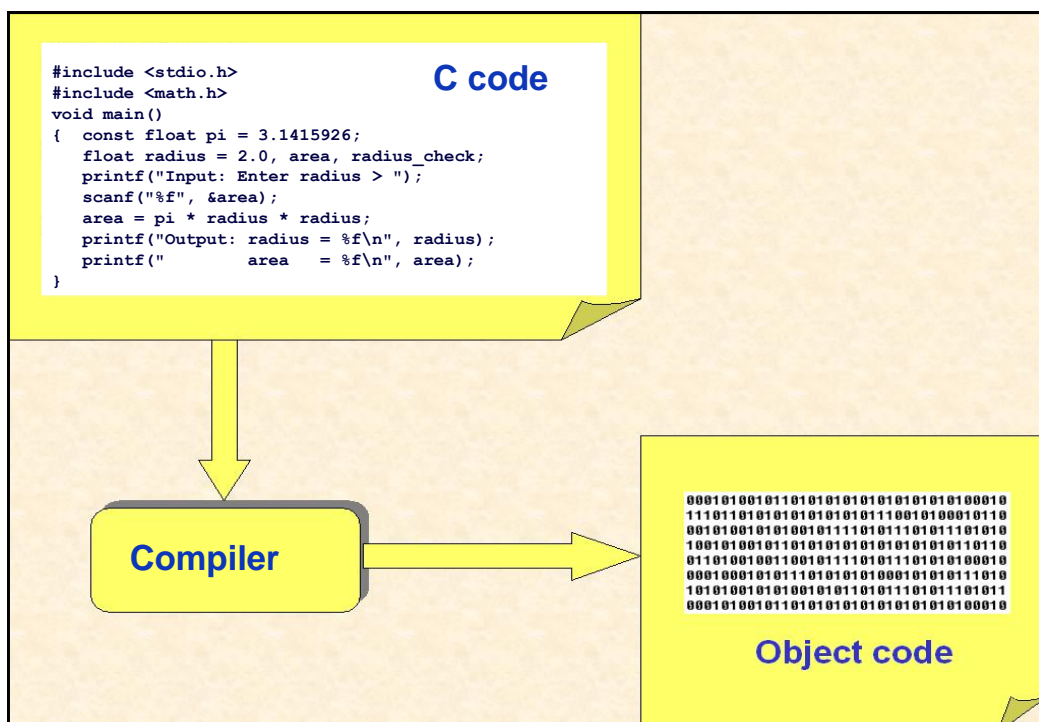
ใช้ภาษาที่มนุษย์เข้าใจ (English-like language)

เช่น Basic, Pascal, C, JAVA, ...

- ใช้ คอมไพเลอร์ (Compiler) หรือ Interpreter ในการแปลภาษาระดับสูงให้เป็นภาษาเครื่อง

- **Compiler** แปลทั้งโปรแกรม (เช่น Pascal, C, ...)
- **Interpreter** แปลทีละบรรทัด (เช่น Basic, ...)

23

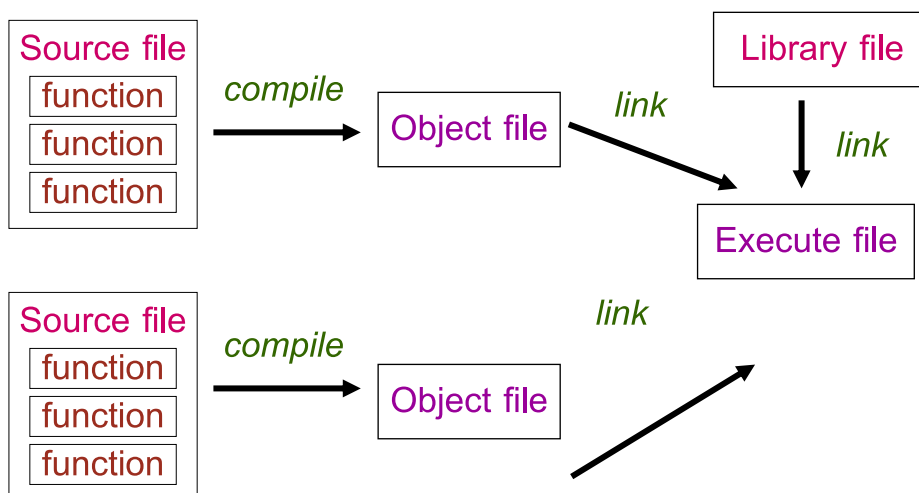


## ภาษาสำหรับนักพัฒนาโปรแกรม

- วิชวลเบสิก (VISUAL Basic)  
พัฒนาโดยบริษัทไมโครซอฟต์ ปัจจุบันใช้เทคโนโลยี .net
- ภาษาจาวา (JAVA)  
พัฒนาโดยบริษัท Sun Microsystem
- ภาษาซีชาร์ป (C#)  
พัฒนาโดยไมโครซอฟต์ ปัจจุบันใช้เทคโนโลยี .net



### ขั้นตอนการสร้างโปรแกรมด้วยภาษา C





## 1.3 การพัฒนาโปรแกรม

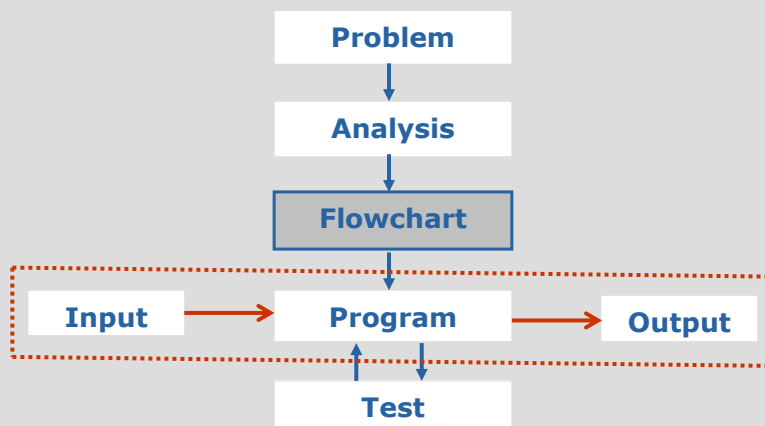
●การพัฒนาโปรแกรม ประกอบด้วย 5 ขั้นตอน

1. กำหนดและวิเคราะห์ปัญหา (State problem & Problem analysis)
2. เขียนผังงาน (Flowchart)
3. เขียนโปรแกรม (Program Development)
4. ทดสอบและแก้ไขโปรแกรม (Testing & debugging)
5. ทำเอกสารและบำรุงรักษาโปรแกรม (Document & maintenance)

27

## ขั้นตอนพัฒนาโปรแกรม

●สรุป 5 ขั้นตอนในการพัฒนาโปรแกรม



## 1.4 กำหนด-วิเคราะห์ปัญหา

- กำหนดขอบเขตของปัญหา ให้ชัดเจนว่า

จะให้คอมพิวเตอร์ทำอะไร (What?)

- วิเคราะห์ปัญหา (Problem analysis) กำหนด

- Input : ลักษณะของข้อมูลเข้า
- Process : วิธีการประมวลผล (How?)
- Output : ลักษณะของผลลัพธ์ที่ต้องการ



29

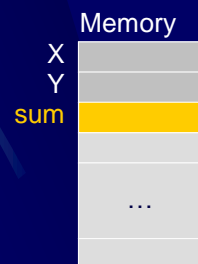
### ตัวอย่าง 1.1

- ออกแบบโปรแกรมให้คอมพิวเตอร์ทำงานเป็นเครื่องคิดเลขอย่างง่าย โดยรับข้อมูล 2 ค่า (X, Y) และแสดงผลบวกทางจอภาพ

- **Problem:** คำนวณผลบวกของ 2 ค่า

- **Problem Analysis**

1. **Input:** รับข้อมูล (X, Y) จากคีย์บอร์ด
2. **Process:** คำนวณ  $sum = X + Y$
3. **Output:** แสดงผลบวก (sum) ทางจอภาพ



30

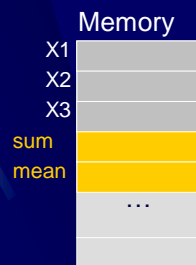
## ตัวอย่าง 1.2

- ออกแบบโปรแกรมให้คอมพิวเตอร์รับข้อมูล 3 ค่า คำนวณค่าเฉลี่ย และแสดงค่าเฉลี่ย ทางจอภาพ
- **Problem:** คำนวณค่าเฉลี่ยของ 3 ค่า  $((X_1 + X_2 + X_3)/3)$

### Problem Analysis

1. **Input:** รับข้อมูล ( $X_1, X_2, X_3$ )
2. **Process:** คำนวณ  $sum = X_1 + X_2 + X_3$   
 $mean = sum/3$

3. **Output:** แสดงค่าเฉลี่ย (mean) ทางจอภาพ



31

## ตัวอย่าง 1.3

- ออกแบบโปรแกรมให้คอมพิวเตอร์รับข้อมูล N ค่า คำนวณค่าเฉลี่ย และแสดงค่าเฉลี่ย ทางจอภาพ

- **Problem:** คำนวณค่าเฉลี่ยของ N ค่า  $(\sum_{i=1}^N X_i/N)$

### Problem Analysis

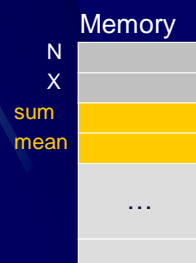
1. **Input:** รับข้อมูล N และ X ( $X_1, ..., X_N$ )
2. **Process:**

```

loop for (i=1 to N)
    read X
    sum = sum + X
end for
mean = sum/N

```

3. **Output:** แสดงค่าเฉลี่ย (mean) ทางจอภาพ



32

## 1.5 การเขียนผังงาน

● **ผังงาน (Flowchart)** เป็นแผนภาพ ที่ใช้อธิบายลำดับการทำงานของโปรแกรมเป็นขั้นตอน

● **ประโยชน์ของ Flowchart**

1. อธิบายลำดับขั้นตอนการทำงานของโปรแกรม
2. ทำให้ตรวจสอบข้อผิดพลาดของโปรแกรมได้ง่าย
3. ทำให้ผู้อื่นสามารถศึกษาการทำงานและแก้ไขโปรแกรมได้ง่าย

33

### 1.5.1 ประเภทของผังงาน

● **ผังงานระบบ (System Flowchart)**

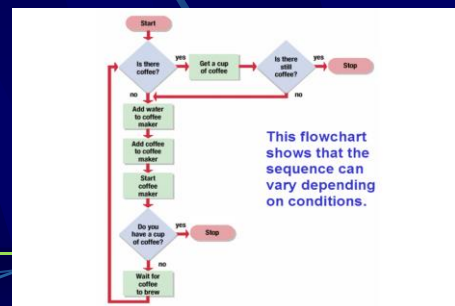
แสดงขั้นตอนการทำงานภายในระบบงาน โดยจะกล่าวอย่างกว้างๆ



● **ผังงานโปรแกรม**




(Program Flowchart)

แสดงขั้นตอนของคำสั่งที่ใช้ในโปรแกรม



## 1.5.2 สัญลักษณ์ในผังงาน

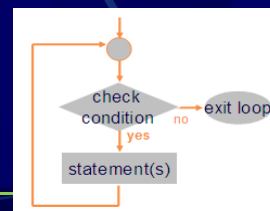
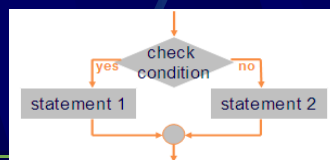
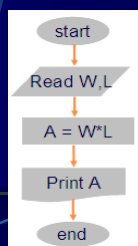
### ● สัญลักษณ์พื้นฐาน ที่นิยมใช้ใน Flowchart

- การเริ่มต้น และการสิ้นสุดการทำงาน
- ↓ ลูกศรแสดงทิศทางการทำงาน และการไหลของข้อมูล
- การประมวลผล หรือการคำนวณ 
- ▱ การรับข้อมูล หรือแสดงผล (ไม่ระบุชนิดอุปกรณ์)
- ⬠ การแสดงผลทางจอภาพ 
- ⬡ การแสดงผลทางเครื่องพิมพ์ (เป็นเอกสาร) 
- ◆ การตรวจสอบเงื่อนไข (เพื่อการตัดสินใจเมื่อมีทางเลือก)
- จุดเชื่อมต่อของผังงาน

35

## 1.5.3 รูปแบบของผังงาน

- แบบลำดับ (Sequence)
- แบบมีทางเลือก (Selection)
- แบบทำซ้ำ (Looping)

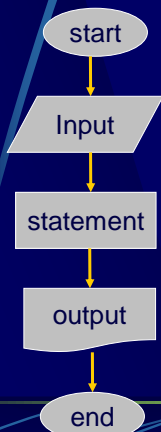


36



## 1.5.3.1 ผังงานแบบลำดับ

### ● Flowchart แบบลำดับ (Sequence)

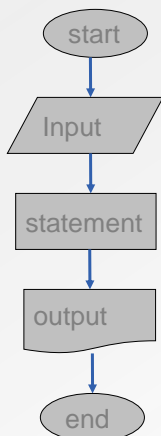


แสดงขั้นตอนการทำงานที่เรียงลำดับ  
(ไม่มีการข้ามขั้น หรือย้อนกลับ)

37

## ผังงานแบบลำดับ

### • Flowchart แบบลำดับ (Sequence)



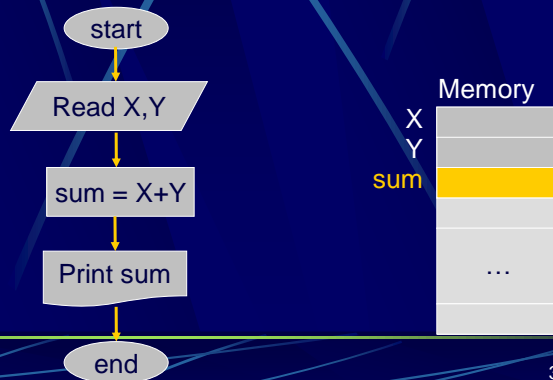
แสดงขั้นตอนการ  
(ไม่มีการข้ามขั้น

```
#include "stdio.h"
#include "conio.h"
main()
{
    int x,y;
    scanf("%d",&x);
    y = x + 1;
    printf("Output = %d\n",y);

    getch();
}
```

## ตัวอย่าง 1.4

- แสดงการออกแบบ **Flowchart** เพื่อให้โปรแกรมทำงานเป็นเครื่องคิดเลขอย่างง่าย โดยรับข้อมูล 2 ค่า (X, Y) คำนวณ **การบวก** (sum) พร้อมแสดงผลบวก

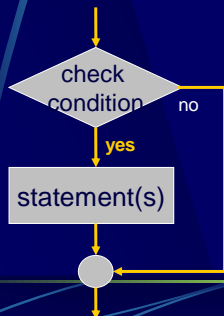


39

## 1.5.3.2 ฟังก์ชันแบบมีทางเลือก

- แสดงการตรวจสอบเงื่อนไขให้โปรแกรมเลือกทำอย่างใดอย่างหนึ่ง ซึ่งมี **3 กรณี**

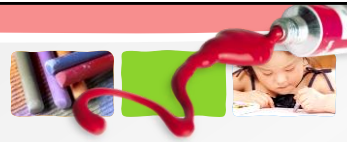
### 1. การเลือกแบบ 1 เส้นทาง



จะทำงานเฉพาะเมื่อเงื่อนไขเป็น**จริง**เท่านั้น

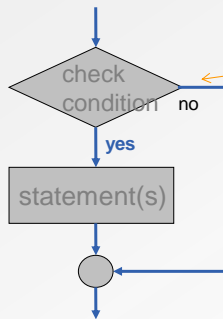
40

## ผังงานแบบมีทางเลือก



- แสดงการตรวจสอบเงื่อนไขให้โปรแกรมเลือกทำอย่างใดอย่างหนึ่ง ซึ่งมี 3 กรณี

### 1. การเลือกแบบ 1 เส้นทาง



ถ้ามี if ไม่ต้องมี else

จะทำงานเฉพาะเมื่อเงื่อนไขเป็นจริงเท่านั้น

41

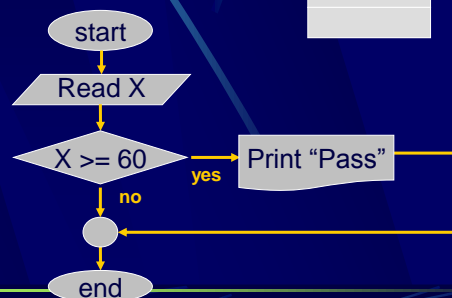
## ตัวอย่าง 1.5

- แสดงการออกแบบ Flowchart เพื่อให้โปรแกรมรับคะแนนนักศึกษา (X) ตรวจสอบและแสดงผล ถ้าผ่าน (Pass) โดยมีเงื่อนไขดังนี้

**เงื่อนไข**

คะแนน  $\geq 60$  ผ่าน (Pass)

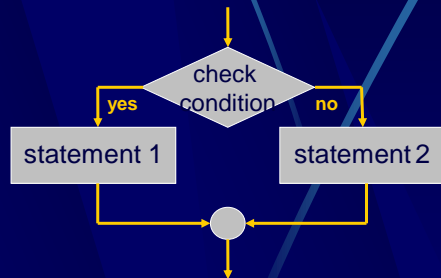
คะแนน  $< 60$  ตก (Fail)



42

## ผังงาน-ทางเลือก

### 2. การเลือกแบบ 2 เส้นทาง



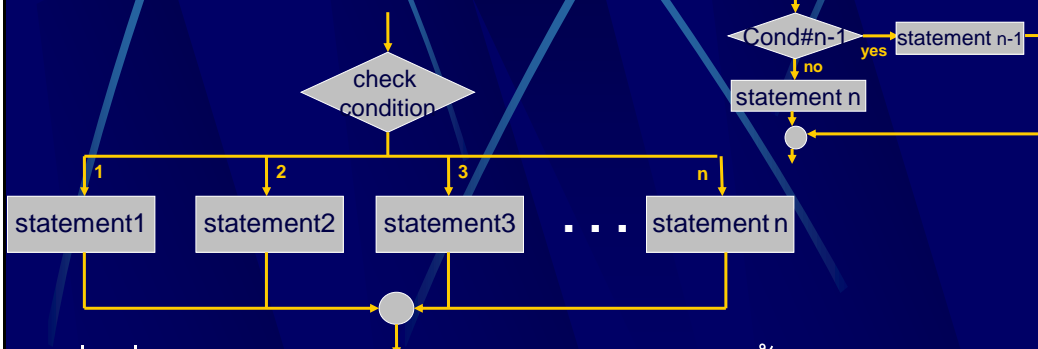
เมื่อเงื่อนไขเป็น**จริง**จะทำการอย่างหนึ่ง

เมื่อเงื่อนไขเป็น**เท็จ**จะทำอีกอย่างหนึ่ง

43

## ผังงาน-ทางเลือก

### 3. การเลือกแบบหลายเส้นทาง (n)



เมื่อเงื่อนไขเท่ากับทางเลือกใดจะทำตามทางนั้น

44

## ตัวอย่าง 1.6

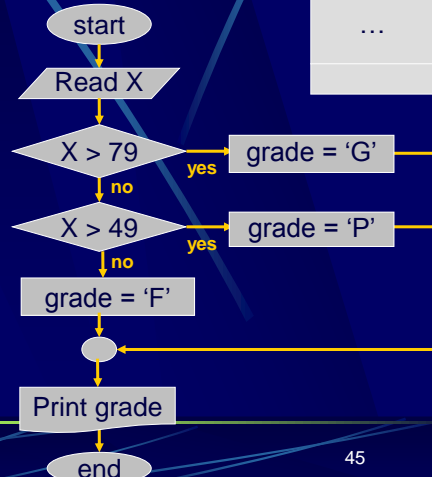
- แสดงการออกแบบ Flowchart เพื่อให้โปรแกรมรับคะแนนนักศึกษา (X) ตรวจสอบและจัดกลุ่มตามเงื่อนไข พร้อมแสดงผลลัพธ์

### เงื่อนไข

คะแนน 80-100 กลุ่ม G

คะแนน 50-79 กลุ่ม P

คะแนนต่ำกว่า 50 กลุ่ม F



45

## ตัวอย่าง 1.7

- แสดงการออกแบบ Flowchart เพื่อให้โปรแกรมรับคะแนนนักศึกษา (X) ตรวจสอบและตัดเกรดตามเงื่อนไข พร้อมแสดงผลลัพธ์

### เงื่อนไข

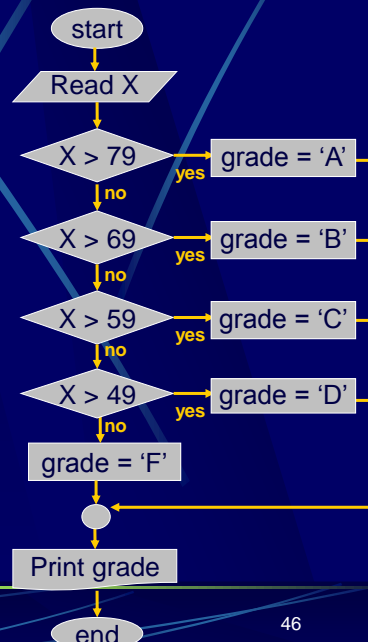
คะแนน 80-100 เกรด A

คะแนน 70-79 เกรด B

คะแนน 60-69 เกรด C

คะแนน 50-59 เกรด D

คะแนนต่ำกว่า 50 เกรด F



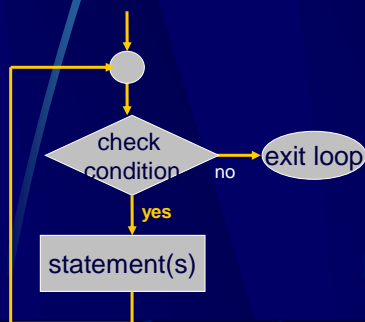
46



### 1.5.3.3 ผังงานแบบทำซ้ำ

- การทำซ้ำ (Looping) แบ่งเป็น 3 กรณี

- 1. การทำซ้ำเมื่อเงื่อนไขเป็นจริง



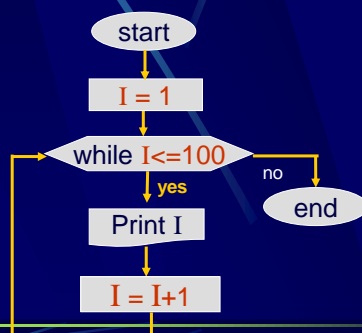
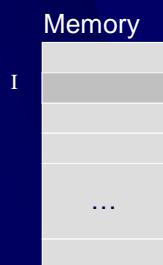
ตรวจสอบเงื่อนไขก่อน  
จะทำงาน (Statement) ซ้ำ  
เมื่อเงื่อนไขเป็น**จริง**  
(ออกจากทำงานซ้ำเมื่อเงื่อนไขเป็น**เท็จ**)

47

### ตัวอย่าง 1.8

- แสดงการออกแบบ **Flowchart** เพื่อให้โปรแกรมคำนวณ  
การกำหนดค่าที่เพิ่มทีละ 1 จาก 1 – 100

- กำหนดให้  $I = 1, 2, 3, \dots, 100$



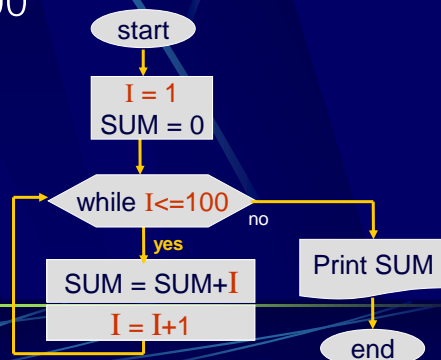
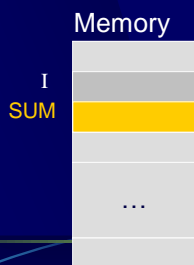
48

## ตัวอย่าง 1.9

● แสดงการออกแบบ **Flowchart** เพื่อให้โปรแกรมคำนวณการบวก  $1+2+\dots+100$

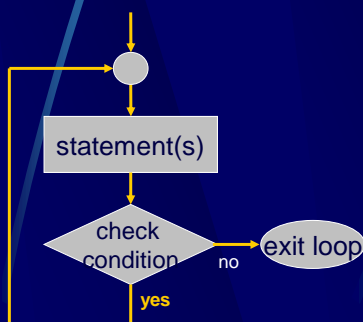
● กำหนดให้  $I = 1, 2, 3, \dots, 100$

และ  $SUM = 1+2+3+\dots+100$



## ผังงาน-ทำซ้ำ

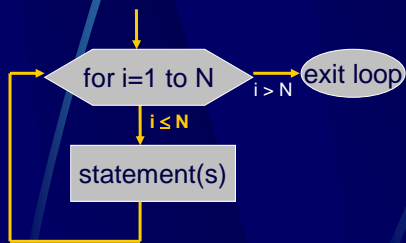
### 2. การทำซ้ำในขณะที่เงื่อนไขเป็นจริง



ทำงาน (Statement) ก่อน การ  
ตรวจเงื่อนไข  
และทำซ้ำถ้าเงื่อนไขเป็น**จริง**  
(จนเมื่อเป็น**เท็จ**จึงออกจากทำงานซ้ำ)

# ผังงาน-ทำซ้ำ

## 3. การทำซ้ำตามจำนวนที่ระบุ



ทำงานตามรอบที่กำหนด  
โดยเริ่มจากรอบเริ่มต้น ( $i=1$ )  
ไปยังรอบสุดท้าย ( $i=N$ )  
(ปกติการนับรอบจะเพิ่มทีละ  
1 ค่า ( $i = i+1$ ))

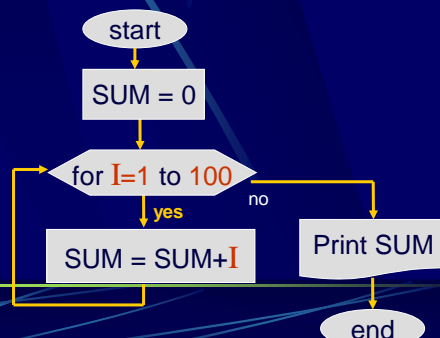
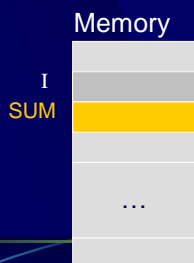
51

## ตัวอย่าง 1.10

● แสดงการออกแบบ Flowchart เพื่อให้โปรแกรมคำนวณการบวก  $1+2+\dots+100$

● กำหนดให้  $I = 1, 2, 3, \dots, 100$

และ  $SUM = 1+2+3+\dots+100$

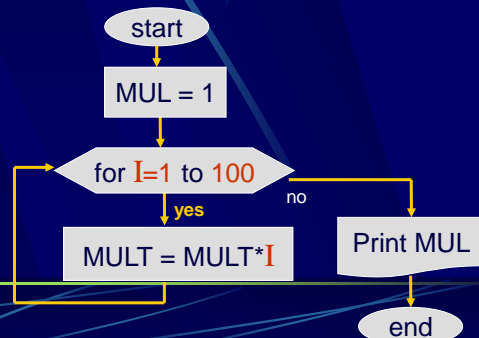
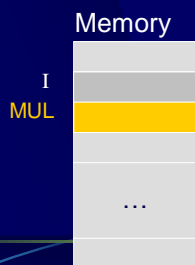


## ตัวอย่าง 1.11

- แสดงการออกแบบ Flowchart เพื่อให้โปรแกรมคำนวณการคูณ  $1 \times 2 \times \dots \times 100$

กำหนดให้  $I = 1, 2, 3, \dots, 100$

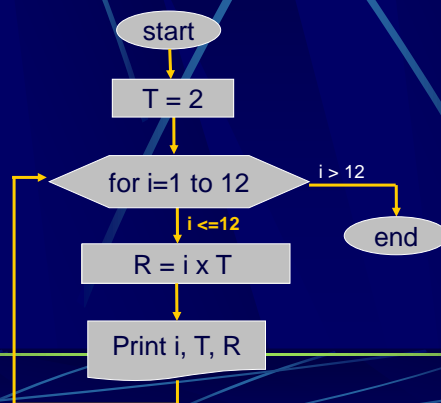
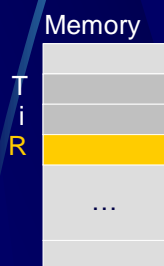
และ  $MUL = 1 \times 2 \times 3 \times \dots \times 100$



## ตัวอย่าง 1.12

- แสดงการออกแบบ Flowchart เพื่อให้โปรแกรมคำนวณตาราง สูตรคูณแม่ 2

กำหนดให้  $T = 2, i = 1, 2, \dots, 12$ , และ  $R = i \times T$

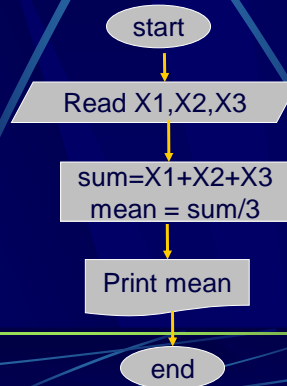
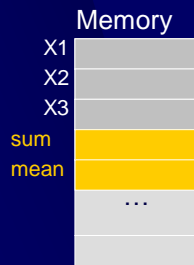


$i \times T = R$
$1 \times 2 = 2$
$2 \times 2 = 4$
$3 \times 2 = 6$
$4 \times 2 = 8$
$5 \times 2 = 10$
$6 \times 2 = 12$
$7 \times 2 = 14$
$8 \times 2 = 16$
$9 \times 2 = 18$
$10 \times 2 = 20$
$11 \times 2 = 22$
$12 \times 2 = 24$

54

## ตัวอย่าง 1.13

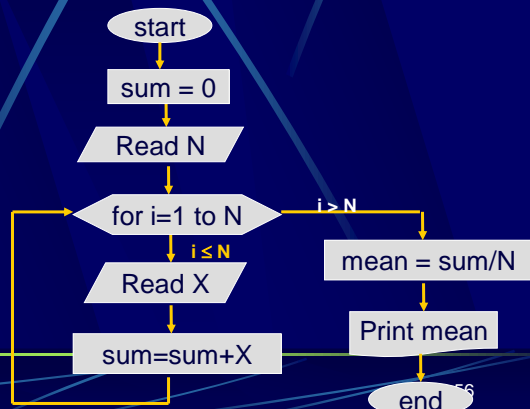
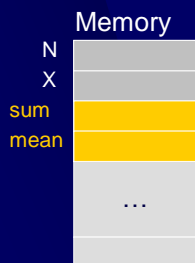
- แสดงการออกแบบ **Flowchart** เพื่อให้โปรแกรม รับข้อมูล 3 ค่า ( $X_1, X_2, X_3$ ) คำนวณค่าเฉลี่ย (mean) พร้อมแสดงผลค่าเฉลี่ย



55

## ตัวอย่าง 1.14

- แสดงการออกแบบ **Flowchart** ให้โปรแกรมรับค่า  $N$  (จำนวนข้อมูล) และข้อมูลแต่ละค่า  $X$  ( $X_1, X_2, \dots, X_N$ ) คำนวณค่าเฉลี่ย ( $\sum_{i=1}^N X_i / N$ ) และแสดงค่าเฉลี่ย

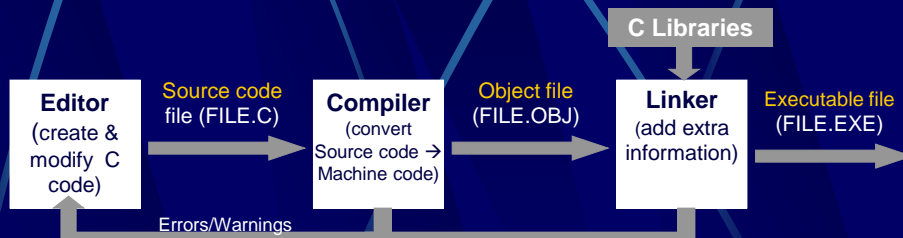


56



## 1.6 การ Compile โปรแกรม

- ขั้นตอนการแปล Source code (file) ของโปรแกรมภาษา C ให้เป็น Machine code (Object file)



- ขั้นตอนการประมวลผลโปรแกรมภาษา C



57

## Compile & Run

- โปรแกรมภาษา C จะอยู่ในรูปแบบของฟังก์ชันซึ่งมีอย่างน้อยหนึ่งฟังก์ชัน (คือ main)
- ตัวอย่างการ Edit โปรแกรมภาษา C โดย Turbo C

The screenshot shows the Turbo C IDE with the **Edit** menu selected. The file being edited is **C:FILE.C**. The code in the editor is as follows:

```
#include <stdio.h>
void main()
{
    printf ("Programming is fun.\n");
}
```

At the bottom of the window, there is a **Message** pane.

# เริ่มต้นเขียนโปรแกรม

```
#include "stdio.h"
#include "conio.h"
main()
{
    getch();
}
```

เขียนคำสั่งต่าง ๆ แทรกลง

# ข้อผิดพลาดในการเขียนโปรแกรม

- **Syntax Error หรือ Coding Error**
- **Logic Error**

## 1.6.1 โครงสร้างโปรแกรม(พื้นฐาน)

- **Preprocessor Directive** (ข้อความสั่งตัวประมวลผลก่อน)

- **Main Function** (ฟังก์ชันหลัก)

```
void main()           /* main function */
{                     /* Begin (เริ่มต้น) */
    variable declaration; /* ประกาศตัวแปรที่เก็บข้อมูล */
    statements;         /* ข้อความสั่งประมวลผล */
}                     /* End (จบ) */
```

- **ตัวอย่างเช่น** ถ้าต้องการพิมพ์ Programming is fun.

- **C Program** คือ

```
#include <stdio.h>      /* Preprocessor directive */
void main()
{
    printf("Programming is fun.\n");
}
```

- \n หมายถึง การย้าย cursor ไปบรรทัดใหม่ (newline)

61

## Program-1

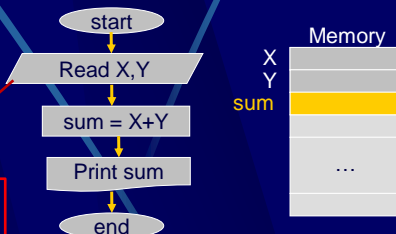
- โปรแกรมแสดงการรับข้อมูล 2 ค่า (X, Y) และแสดงผลบวก (X+Y) ทางจอภาพ

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int X, Y, sum;
    printf("Enter X: "); scanf("%d", &X);
    printf("Enter Y: "); scanf("%d", &Y);
    sum = X + Y;
    printf("sum = %d\n", sum);
    getch(); /* get 1 character from keyboard */
}
```

- **printf** หมายถึง ฟังก์ชันแสดงผลลัพธ์ (Output) ในภาษา C

- **scanf** หมายถึง ฟังก์ชันรับข้อมูล (Input) ในภาษา C

- **%d** หมายถึง ชนิดของข้อมูลแบบ Integer (หรือ Decimal)



62

## 1.6.2 การแสดงผล (printf)

● **printf** (ฟังก์ชันมาตรฐานใน stdio.h)

● รูปแบบ **printf("control string", variable,...);**

● **Variable** เป็นตัวแปรใช้เก็บค่า (ที่เปลี่ยนแปลงได้) ใน memory ในขณะประมวลผล

● **control string** ประกอบด้วย

● ข้อความอธิบาย เช่น printf("C Programming");

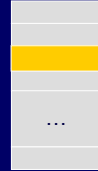
● %format เช่น printf("SUM = %d\n", sum);

● ลำดับหลัก (Escape sequence) เช่น \n (new line)

%d สำหรับ integer หรือ decimal %f สำหรับ real หรือ floating point

Memory

variable



63

## การจัด %format

● **%format** ระบุชนิดของตัวแปร

ชนิดข้อมูล	ชนิดตัวแปร	รูปแบบ
จำนวนเต็ม (integer)	int	%d
จำนวนจริง (real)	float	%f

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int X, Y, sum;
    printf("Enter X: "); scanf("%d", &X);
    printf("Enter Y: "); scanf("%d", &Y);
    sum = X + Y;
    printf("sum = %d\n", sum);
    getch();
}
```

64

## 1.6.3 การรับค่า (scanf)

- **scanf** (ฟังก์ชันมาตรฐานใน stdio.h)

- รูปแบบ `scanf("%format", &variable,...);`

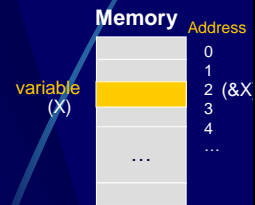
- **&variable** หมายถึง ตำแหน่งที่อยู่ (Address) ของตัวแปรที่เก็บในหน่วยความจำ

- **%format** ขึ้นต้นด้วย % เช่น %d

%d สำหรับ integer หรือ decimal    %f  
สำหรับ real หรือ floating point

- ตัวอย่างเช่น `scanf("%d", &X);`

รับข้อมูลชนิด integer จากแป้นพิมพ์  
แล้วนำไปเก็บไว้ในตัวแปร X ที่ &X



65

## คำสั่งพื้นฐานการรับและแสดงผลข้อมูล

- ฟังก์ชันแสดงผล

`putchar(ch);`

`puts()` →

- ฟังก์ชันรับข้อมูล

`getchar()`

`getch()`

`gets()`

```
char m = 'T';  
putchar(m);
```

```
char name[10] = 'Computer';  
puts(name);  
puts("THAI");
```

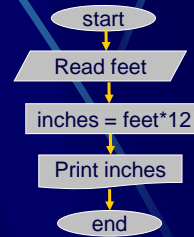
```
ch = getch();
```

66

## Program-2

- โปรแกรมแสดงการรับข้อมูลค่าความยาว (มีหน่วยเป็น ฟุต) คำนวณการเปลี่ยนหน่วยเป็นนิ้ว

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int feet, inches;
    printf("Enter number (in feet) ");
    scanf("%d", &feet);
    inches = feet*12;
    printf("= %d inches\n", inches);
    getch();
}
```



### ผลลัพธ์

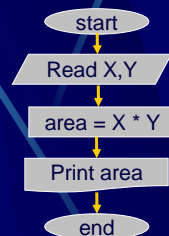
Enter number (in feet) 2  
= 24 inches

67

## Program-3

- โปรแกรมแสดงการรับข้อมูลค่าความกว้าง ความยาว ของสี่เหลี่ยม คำนวณพื้นที่และแสดงผล

```
#include <stdio.h>
void main()
{
    int X, Y;
    float area;
    printf("Enter width (X): ");
    scanf("%d", &X);
    printf("Enter length (Y): ");
    scanf("%d", &Y);
    area = X * Y;
    printf("Area = %f\n", area);
    getch();
}
```



### ผลลัพธ์

Enter width (X): 5  
Enter length (Y): 10  
Area = 50.0

68

## Program-4

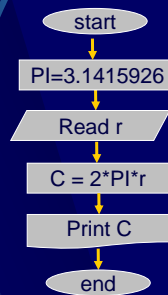
โปรแกรมแสดงการรับข้อมูลค่ารัศมีของวงกลม

คำนวณ เส้นรอบวง ( $2\pi r$ ) และแสดงผล

```
#include <stdio.h>
void main()
{
    float r, C, PI = 3.1415926;
    printf("Enter radius: ");
    scanf("%f", &r);
    C = 2*PI*r;
    printf("Circumference = %f\n", C);
    getch();
}
```

**ผลลัพธ์**

Enter radius 2  
Circumference = 12.566370

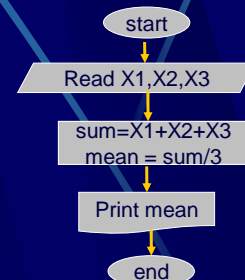


69

## Program-5

โปรแกรมแสดงการรับข้อมูล 3 ค่า คำนวณค่าเฉลี่ย และแสดงค่าเฉลี่ย ทางจอภาพ

```
#include <stdio.h>
void main()
{
    int X1, X2, X3;
    float sum, mean;
    printf("Enter X1,X2,X3: ");
    scanf("%d,%d,%d", &X1,&X2,&X3);
    sum = X1+X2+X3;
    mean = sum/3;
    printf("Mean = %f\n", mean);
    getch();
}
```



**ผลลัพธ์**

Enter X1,X2,X3: 5,2,10  
Mean = 5.666667

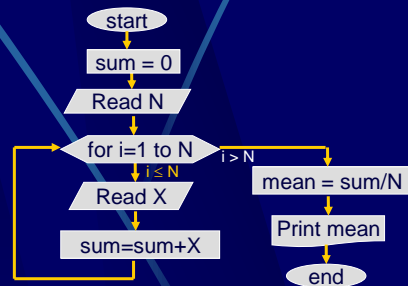
70



## Program-6

- โปรแกรมแสดงการรับข้อมูล  $N$  ค่า คำนวณค่าเฉลี่ย และแสดงค่าเฉลี่ยทางจอภาพ

```
#include <stdio.h>
void main()
{
    int i, N, X;
    float sum=0, mean;
    printf("Enter N: "); scanf("%d", &N);
    for (i=1; i<=N; i++) {
        printf("Enter X%d= ", i); scanf("%d", &X);
        sum = sum + X;
    }
    mean = sum/N;
    printf("Mean = %f\n", mean);
    getch();
}
```



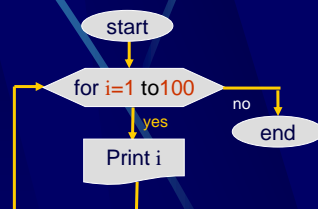
### ผลลัพธ์

Enter N: 3  
 Enter X1= 5  
 Enter X2= 2  
 Enter X3= 10  
 Mean = 5.666667 71

## Program-7

- โปรแกรมแสดงการกำหนดค่า 1, 2, 3, ..., 100

```
#include <stdio.h>
void main()
{
    int i;
    for (i=1; i<=100; i++) {
        printf("%d, ", i);
    }
    getch();
}
```



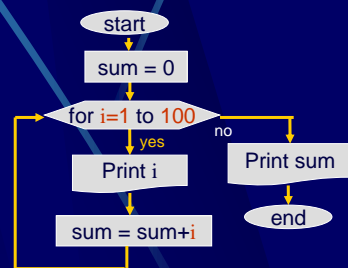
### ผลลัพธ์

1, 2, 3, 4, 5, ..., 100

# Program-8

โปรแกรมแสดงการบวก  $1+2+\dots+100$

```
#include <stdio.h>
void main()
{
    int i, sum=0;
    for (i=1; i<=100; i++) {
        printf("%d+ ", i);
        sum = sum + i;
    }
    printf(" = %d\n", sum);
    getch();
}
```



**ผลลัพธ์**

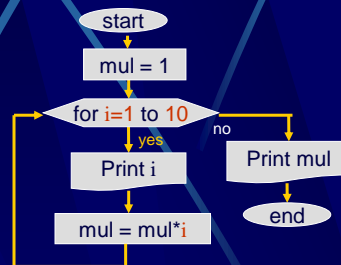
$1+2+3+\dots+100 = 5050$

73

# Program-9

โปรแกรมแสดงการคูณ  $1 \times 2 \times \dots \times 10$

```
#include <stdio.h>
void main()
{
    int i;
    float mul=1;
    for (i=1; i<=10; i++) {
        printf("%d x ", i);
        mul = mul * i;
    }
    printf(" = %f\n", mul);
    getch();
}
```



**ผลลัพธ์**

$1 \times 2 \times 3 \times \dots \times 10 = 3628800.0$

74

# Program-10

โปรแกรมแสดงการคำนวณ ตารางสูตรคูณแม่ 2

กำหนดให้  $T = 2$ ,  $i = 1, 2, \dots, 12$ , และ  $R = i \times T$

```
#include <stdio.h>
```

```
void main()
```

```
{ int i, T=2, R;
```

```
for (i=1; i<=12; i++) {
```

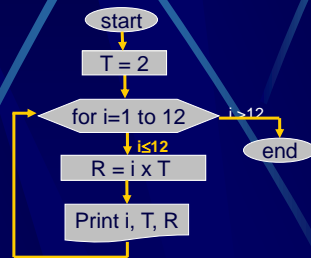
```
    R = i * T;
```

```
    printf("%d x %d = %d\n", i, T, R);
```

```
}
```

```
getch();
```

```
}
```



$1 \times 2 = 2$   
 $2 \times 2 = 4$   
 $3 \times 2 = 6$   
 $4 \times 2 = 8$   
 $5 \times 2 = 10$   
 $6 \times 2 = 12$   
 $7 \times 2 = 14$   
 $8 \times 2 = 16$   
 $9 \times 2 = 18$   
 $10 \times 2 = 20$   
 $11 \times 2 = 22$   
 $12 \times 2 = 24$

75

# Program-11

โปรแกรมแสดงการคำนวณ ตาราง Latin Square

ขนาด 5x5

```
#include <stdio.h>
```

```
void main()
```

```
{ int i, j, T=5, L;
```

```
for (i=1; i<=T; i++) { /* row i */
```

```
    for (j=0; j<T; j++) { /* col j */
```

```
        L = (i+j);
```

```
        if (L>T) L = L%(T+1)+1;
```

```
        printf("%d ", L);
```

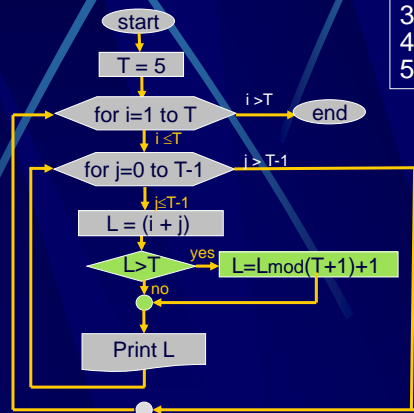
```
    }
```

```
    printf("\n"); /* new row */
```

```
}
```

```
getch();
```

```
}
```



1	2	3	4	5
2	3	4	5	1
3	4	5	1	2
4	5	1	2	3
5	1	2	3	4

Latin S

1	2	3	4	5
2	3	4	5	1
3	4	5	1	2
4	5	1	2	3
5	1	2	3	4

76