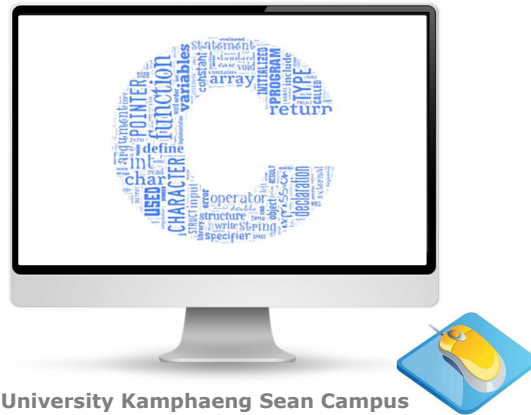


## Chapter 4 : Flow Chart & Selection Statement



**Computer Engineering, Kasetsart University Kamphaeng Sean Campus**

# Outline

- Flow chart
- Introduce: Boolean data type
- Selection Statement
- Short Circuit Evaluation

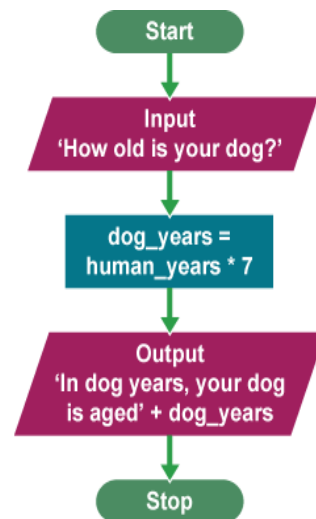
2



## 1. Flow Chart

ผังงาน (Flow Chart) เป็นการอธิบายขั้นตอนวิธีการแก้ปัญหา โดยใช้รูปสัญลักษณ์มาเรียงต่อกัน

- สัญลักษณ์แต่ละแบบจะมีความหมายถึงกระบวนการที่แตกต่างกัน
- สัญลักษณ์ของผังงานที่ใช้ในปัจจุบันกำหนดโดยสถาบันมาตรฐานแห่งชาติอเมริกา (The American National Standard Institute: ANSI)




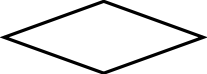
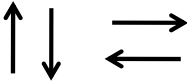




ตัวอย่าง: พังงานคำนวณอายุสุนัข



3

## 1. Flow chart

สัญลักษณ์	ความหมาย
	จุดเริ่มต้นหรือจุดจบของโปรแกรม (Terminal)
	การรับและแสดงผลของข้อมูล (Input /Output)
	การประมวลผล (Process)
	การตัดสินใจ (Decision /Selection)
	ทิศทางของขั้นตอนการดำเนินงาน (Flow Line)
	จุดเชื่อมการทำงานที่อยู่หน้าเดียวกัน(Connector)
	จุดเชื่อมการทำงานที่อยู่คนละหน้า (Off Page Connector)



4

# 1. Flow chart

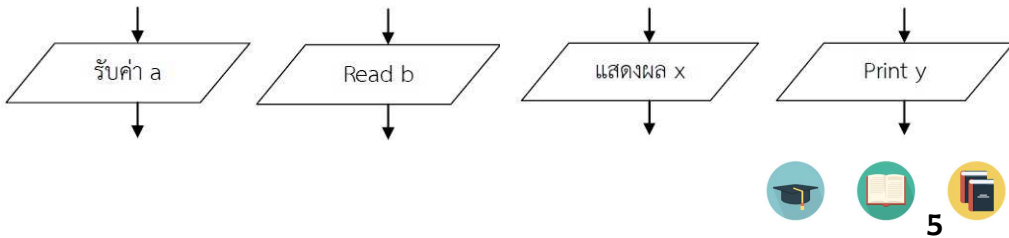
- **สัญลักษณ์ การประมวลผล:** ใช้สำหรับการกำหนดค่าเริ่มต้น รวมถึงการคำนวณในรูปของสูตรสมการคณิตศาสตร์ ซึ่งขั้นตอนเหล่านี้จะเขียนข้อความภายในสัญลักษณ์กรอบสี่เหลี่ยมผืนผ้า ดังภาพ

num = 5

Area = (a \* ha) / 2

num = 5;  
Area = (a \* ha) / 2;  
value++;

- **สัญลักษณ์ การรับและแสดงผลของข้อมูล:** ใช้สำหรับการรับข้อมูลเข้าสู่โปรแกรม เพื่อนำไปประมวลผล และนำข้อมูลที่ได้จากการกำหนดค่า หรือการประมวลผลใดใด มาแสดงผลออกทางอุปกรณ์ที่กำหนด



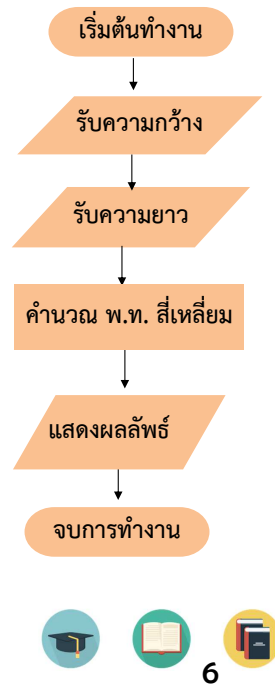
5

# 1. Flow chart

ตัวอย่างคำนวณหาพื้นที่ของรูปสี่เหลี่ยมผืนผ้า

ขั้นตอนการทำงานของโปรแกรม :

- เริ่มต้นทำงาน
- รับความกว้างของรูปสี่เหลี่ยมผืนผ้า
- รับความยาวของรูปสี่เหลี่ยมผืนผ้า
- คำนวณหาพื้นที่ของรูปสี่เหลี่ยมผืนผ้า
- แสดงผลลัพธ์ที่ได้ออกทางจอภาพ
- จบการทำงาน



6

# 1. Flow chart

- **วิธีการเขียนผังงานที่ดี :**

การเขียนผังงานควรคำนึงถึงสิ่งต่าง ๆ ดังนี้

1. ใช้สัญลักษณ์ที่มีรูปแบบมาตรฐาน
2. ผังงานจะต้องมีจุดเริ่มต้น (Start) และสิ้นสุด (Stop/End/Finish) เพียงหนึ่งจุด
3. ใช้หัวลูกศรแสดงทิศทางการไหลของข้อมูลจากบนลงล่าง หรือ ซ้ายไปขวา
4. เขียนคำอธิบายการทำงานในแต่ละขั้นตอนโดยใช้ข้อความที่สั้น กระชับ ชัดเจน และเข้าใจได้ง่าย
5. ควรหลีกเลี่ยงโยงเส้นไปมาทำให้เกิดจุดตัดมากเพราะจะทำให้เกิดข้อผิดพลาด (ควรใช้สัญลักษณ์เชื่อมจุดต่อเนื่องแทน)
6. ควรเขียนผังงานให้จบภายในหน้าเดียว
7. ควรมีความเป็นระเบียบเรียบร้อย สะอาดและชัดเจน สามารถเข้าใจและติดตามขั้นตอนได้ง่าย

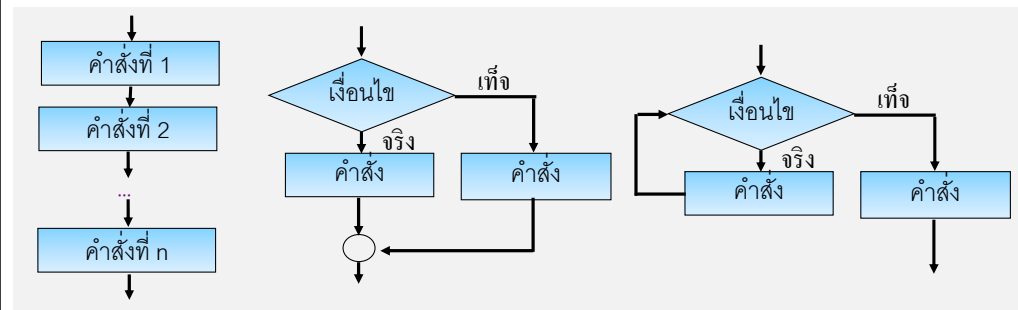


7

# 1. Flow chart

- **ผังงานโครงสร้างควบคุมหลักในการเขียนโปรแกรม**

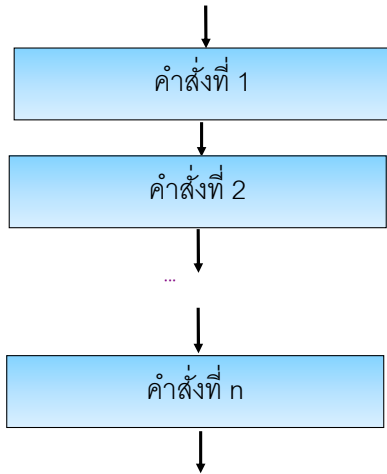
- โครงสร้างแบบลำดับ (Sequential structure)
- โครงสร้างแบบมีทางเลือก (Selection structure)
- โครงสร้างแบบทำซ้ำ (Repetition structure)



8

# 1. Flow chart

## ■ โครงสร้างแบบลำดับ (Sequential structure)



- โครงสร้างแสดงขั้นตอนการทำงานที่เป็นไปตามลำดับก่อนหลัง
- แต่ละขั้นตอนจะถูกประมวลผลเพียงครั้งเดียวเท่านั้น



9

# 1. Flow chart

## ■ โครงสร้างแบบมีทางเลือก (Selection structure)

โครงสร้างแบบมีทางเลือก คือ โครงสร้างที่มีเงื่อนไข ขั้นตอนการทำงานบางขั้นตอนที่ต้องมีการตัดสินใจ เพื่อเลือกวิธีการประมวลผลขั้นต่อไป และอาจจะมีบางขั้นตอนที่ไม่ได้รับการประมวลผล

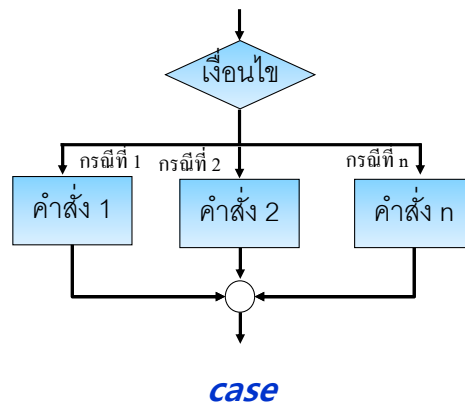
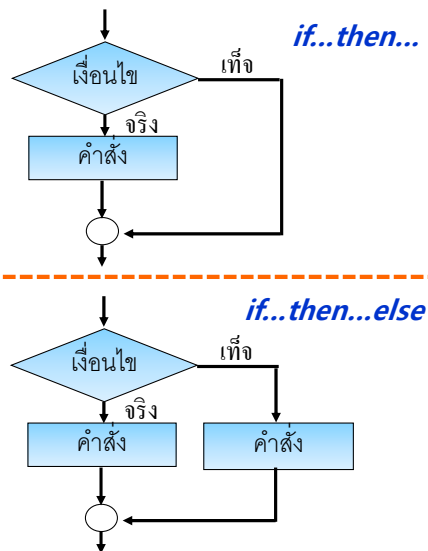
- โครงสร้างแบบเลือกทำทางเดียว ที่เรียกว่า *if...then...*
- โครงสร้างแบบเลือกทำทางใดทางหนึ่ง ที่เรียกว่า *if...then...else*
- โครงสร้างที่มีทางเลือกมากกว่า 2 ทาง ที่เรียกว่า *case*



10

# 1. Flow chart

## ■ โครงสร้างแบบมีทางเลือก (Selection structure)

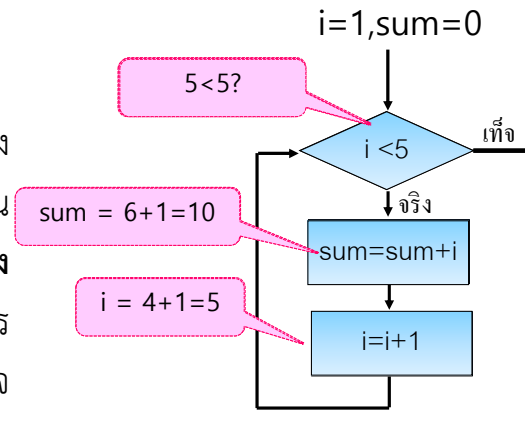


11

# 1. Flow chart

## ■ โครงสร้างแบบทำซ้ำ (Repetition structure)

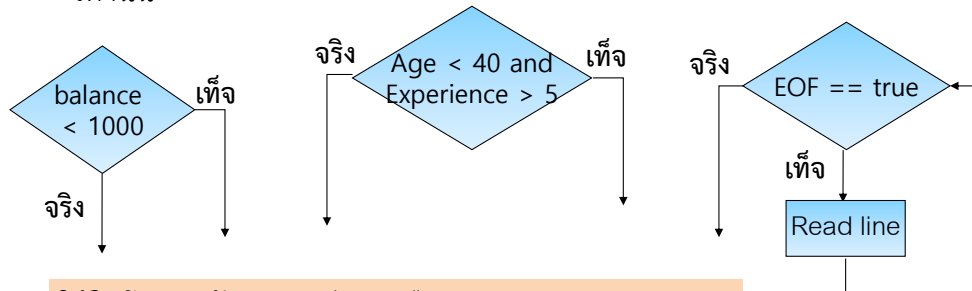
โครงสร้างแบบทำซ้ำ คือ โครงสร้างที่ขั้นตอนการทำงานบางขั้นตอนได้รับการประมวลผลมากกว่า 1 ครั้ง ทั้งนี้ขึ้นอยู่กับเงื่อนไขบางประการ โครงสร้างแบบซ้ำนี้ต้องมีการตัดสินใจในการทำงานซ้ำ



12

# 1. Flow chart

- **สัญลักษณ์ การตัดสินใจ:** ใช้สำหรับเปรียบเทียบเพื่อตรวจสอบเงื่อนไข ซึ่งจะได้ผลลัพธ์จากการตรวจสอบเป็นค่า **จริง** หรือ **เท็จ** อย่างใดอย่างหนึ่งเท่านั้น



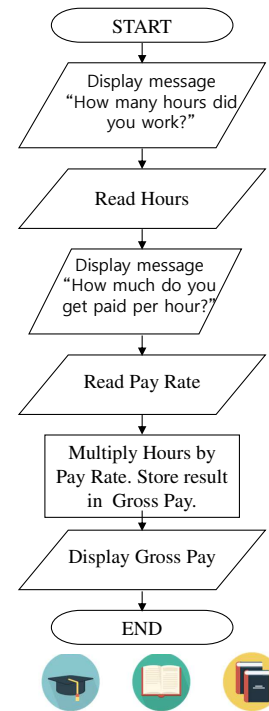
ใช้ในโครงสร้างแบบมีทางเลือก  
และ โครงสร้างแบบทำซ้ำ (เงื่อนไขหยุดการทำซ้ำ)



13

# Quick check1

1. ฟังก์ชันนี้ เป็นฟังก์ชันโครงสร้างแบบใด
2. จงเขียนโปรแกรมภาษา C จากฟังก์ชันนี้



14

## 2. Boolean data type

- ประเภทข้อมูลชนิดตรรกะ (boolean) ที่มีค่าได้สองค่าเท่านั้น คือ **จริง (true)** กับ **เท็จ (false)**
  - ค่าจริง แทนด้วย 1 และ ค่าเท็จแทนด้วย 0

- การประกาศตัวแปรชนิดตรรกะ

```
bool variable_name;
```

\*\* ตามข้อกำหนดมาตรฐานของภาษา C เวอร์ชัน C99 เพื่อใช้ข้อมูลประเภทตรรกะดังตัวอย่างจะต้อง include ไลบรารี “*stdbool.h*”



15

## 2. Boolean data type

- ตัวอย่าง

```
— bool x;  
  x = true;  
  
— bool y = false;  
  
— bool test = (z > 0);  
  // true ถ้า z มีค่ามากกว่าศูนย์
```

```
0  
4 is not an odd number  
1  
15 is an odd number
```

```
#include <stdio.h>  
#include <stdbool.h> //for boolean datatype  
  
int main()  
{  
  n = 15;  
  int n=4;  
  bool odd = n%2;  
  printf("%d\n", odd);  
  if(odd)  
    printf("%d is an odd number",n);  
  else  
    printf("%d is not an odd number",n);  
  return 0;  
}
```



16

## Quick check 2

- จงแสดงผลลัพธ์ทางจอภาพของโปรแกรมนี้

```
#include <stdio.h>
#include <stdbool.h> //library for boolean data type
int main()
{
    bool p = true;
    bool q = 20;
    bool r = false;
    bool s = 0;

    printf("p && q is %d\n", p && q);
    printf("q || s is %d\n", q || s);
    printf("!( ( p || q ) && ( r || s ) ) is %d\n", !( ( p || q ) && ( r || s ) ));
    printf("!( p || q && r || s ) is %d\n", !( p || q && r || s ));

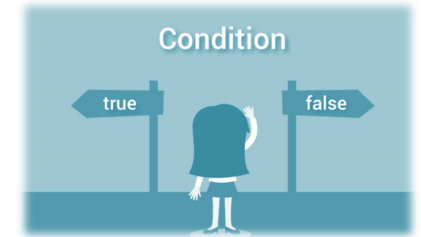
    return 0;
}
```



17

## 3. Selection Statement

- if Statement
- if-else Statement
- Nested if Statement
- Switch statement



18

## 3. Selection Statement (if statement)

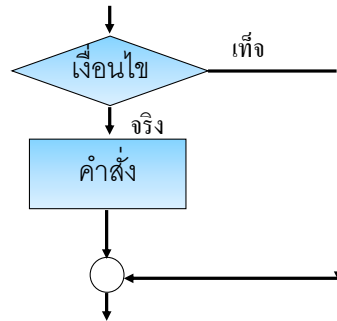
- คำสั่ง if ใช้สำหรับกรณีที่ต้องมีการตัดสินใจหรือมีเงื่อนไข ประกอบด้วย (1) ส่วนของการตรวจสอบเงื่อนไข และ (2) ส่วนของคำสั่งที่ต้องการให้ทำในกรณีที่เงื่อนไขนั้นเป็นจริง

- รูปแบบ

if (นิพจน์หรือเงื่อนไข) คำสั่ง;

if (นิพจน์หรือเงื่อนไข)  
คำสั่ง;

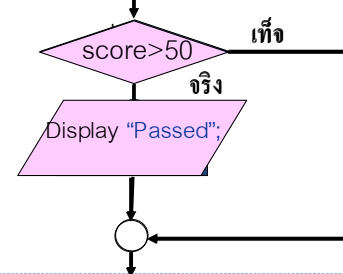
```
if (นิพจน์หรือเงื่อนไข)
{
    คำสั่ง;
    คำสั่ง;
}
```



19

## 3. Selection Statement (if statement)

- ตัวอย่าง: If Statement



```
if (score > 50) printf( "Passed");
```

```
if (score > 50)
    printf( "Passed");
```

```
if (score > 50)
{
    printf( "Passed");
}
```

```
#include <stdio.h>
int main() {

    int score;
    scanf("%d",&score);

    if (score > 50)
        printf("Passed\n");

    return 0;
}
```



20

### 3. Selection Statement (if statement)

- สำหรับกรณีที่เงื่อนไขเป็นจริงแล้วต้องการจะทำหลายคำสั่ง ให้ใช้ปีกกาครอบคำสั่งทั้งหมดนั้น ซึ่งเรียกว่า **"Block"**

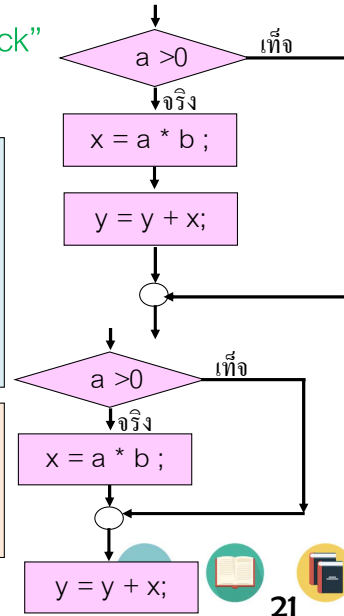
#### รูปแบบ

```
if (นิพจน์หรือเงื่อนไข)
{
    คำสั่ง;
    คำสั่ง;
}
```

#### ตัวอย่าง

```
if (a > 0)
{
    x = a * b;
    y = y + x;
}
```

```
if (a > 0)
    x = a * b;
    y = y + x;
```



21

### 3. Selection Statement (if statement)

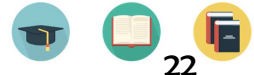
```
int x = 5, y = 5;
```

คำสั่ง (Statements)	ค่า	
	x	y
<code>if (x == 10)</code> <code>    x = x + 1;</code> <code>    y = y + 1;</code>		
<code>if ( x == 10 )</code> <code>    x = x + 1;</code> <code>y = y + 1;</code>		
<code>if ( x == 10 )</code> <code>{</code> <code>    x = x + 1;</code> <code>    y = y + 1;</code> <code>}</code>		
<code>if ( x == 10 );</code> <code>    x = x + 1;</code> <code>    y = y + 1;</code>		

#### ข้อควรระวัง

อย่าให้การเว้นระยะ ย่อหน้า (indentation) ทำให้เกิดความสับสนได้

ภาษา C จะไม่สนใจ ช่องว่าง (space), แท็บ (tab), การขึ้นบรรทัดใหม่ (newline)



22

### 3. Selection Statement (if-else statement)

- คำสั่ง if-else ใช้สำหรับกรณีที่ต้องการตัดสินใจ 2 ทางเลือก
  - ถ้าเงื่อนไขเป็นจริงจะทำคำสั่งภายในส่วนของ if
  - ถ้าเงื่อนไขเป็นเท็จจะทำคำสั่งภายในส่วนของ else

```
if (นิพจน์หรือเงื่อนไข) คำสั่งที่ 1; else คำสั่งที่ 2;
```

```
if (นิพจน์หรือเงื่อนไข) คำสั่งที่ 1;
else คำสั่งที่ 2;
```

```
if (นิพจน์หรือเงื่อนไข)
    คำสั่งที่ 1;
else
    คำสั่งที่ 2;
```

```
if (นิพจน์หรือเงื่อนไข)
{
    คำสั่งที่ 1;
}
else
{
    คำสั่งที่ 2;
}
```

23

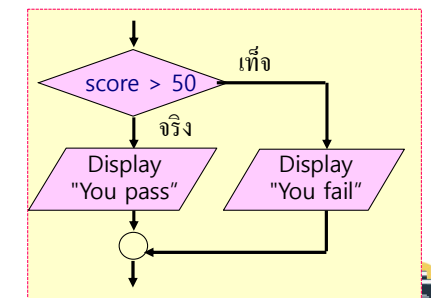
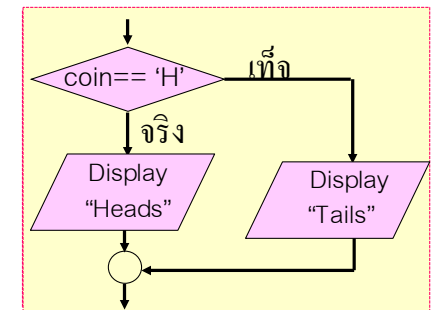
### 3. Selection Statement (if-else statement)

- ตัวอย่าง: if-else Statement

```
if (coin == 'H')
    printf("Heads");
else
    printf("Tails");
```



```
if (score > 50)
    printf("You pass");
else
    printf("You fail");
```



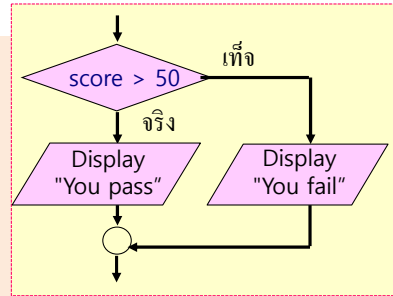
24

### 3. Selection Statement (if-else statement)

```
#include <stdio.h>
int main() {
    int score;
    scanf("%d",&score);

    if (score > 50)
        printf("You Pass\n");
    else
        printf("You Fail\n");

    return 0;
}
```

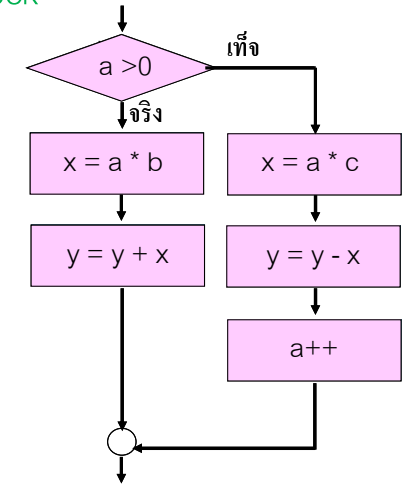


25

### 3. Selection Statement (if-else statement)

- กรณีที่แต่ละทางเลือกจะต้องทำงานมากกว่า 1 คำสั่ง ให้ใช้ ปีกกาครอบคำสั่งทั้งหมดนั้น ซึ่งเรียกว่า **"Block"**

```
if (a > 0)
{
    x = a * b ;
    y = y + x;
}
else
{
    x = a * c ;
    y = y - x;
    a++;
}
```



26

### 3. Selection Statement (if-else statement)

คำสั่ง (Statements)	ค่า	
	x	y
if (x == 5) x = x + 1; else y = y + 1;		
if (x = 0) x = x + 1; else y = y + 1;		
if ( x = 10 ) x = x + 1;		
if ( x == 10 ) x = x + 1; y = y + 1; else y = y + 2;		

**int x = 5, y = 5;**

#### ข้อควรระวัง

if/else จะทำแค่ทางใดทางหนึ่ง

ระวังเครื่องหมาย = (assign) และ == (เปรียบเทียบ)

การ assign จะเป็นเท็จก็ต่อเมื่อ assign ค่า 0 มิฉะนั้นจะเป็นจริง



27

### Quick check3

- กำหนดให้โปรแกรมมีขั้นตอนการทำงานดังนี้  
เริ่มต้น  
รับค่า x และ y  
ถ้า x > y และ y > 0 ให้นำ 0 ใส่ลงไปใน y  
แสดงค่า y  
จบ



จงเขียน flow chart ของโปรแกรมนี้

จงหาค่า y เมื่อคอมพิวเตอร์ทำโปรแกรมนี้นจบ และผู้ใช้ใส่ค่า 5 และ 3 ตามลำดับ



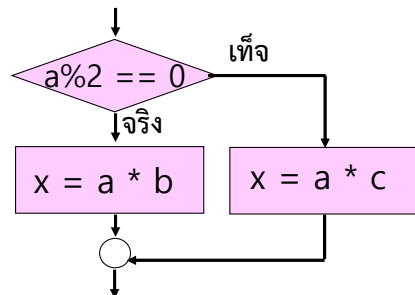
28



### 3. Selection Statement

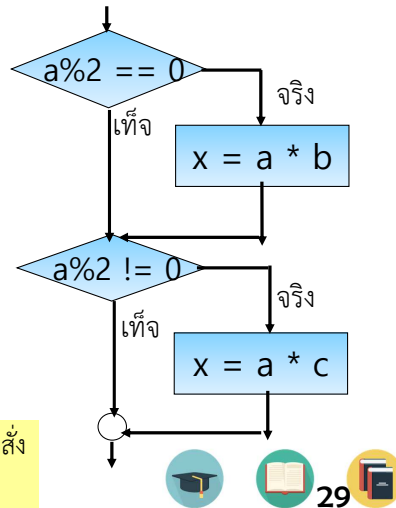
#### ■ if vs if/else statement

```
if (a%2 == 0)
    x = a * b;
else
    x = a * c;
```



ถ้าเงื่อนไขมี 2 ทางเลือกใดๆ โดยทั่วไปแล้วมักจะนิยมใช้คำสั่ง if-else หนึ่งคำสั่งแทนการใช้คำสั่ง if สองคำสั่งติดกัน

```
if (a%2 == 0)
    x = a * b;
if (a%2 != 0)
    x = a * c;
```



29

### 3. Selection Statement

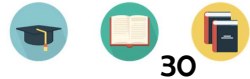
#### ■ if vs if-else statement

**a=8, b=2, c=3**

```
if (a < 10)
    x = a * b;

if (a < 20)
    x = a * c;
```

```
if (a < 10)
    x = a * b;
else
    x = a * c;
```



30

### Quick check4

- จงเขียน Flow chart และ โปรแกรมภาษาซี ที่รับเลขจำนวนเต็มสองตัวจากคีย์บอร์ด โปรแกรมนี้จะพิมพ์คำว่า positive หนึ่งครั้ง เมื่อมีตัวเลขอย่างน้อยหนึ่งตัวเป็นบวก และจะไม่พิมพ์อะไรเลยหากไม่มีตัวเลขที่เป็นบวกอยู่ด้วย

ใช้ if หรือ if-else ดี ??



31

### 4. Short Circuit Evaluation

- ภาษา C เป็นภาษาที่มีประสิทธิภาพสูงในการคำนวณหาผลลัพธ์ตรรก ที่มีเครื่องหมาย && หรือ || ของภาษา C

➔ โดยภาษา C จะทำการตรวจเงื่อนไข จากซ้ายไปขวา แค่เพียงพอที่จะสรุปค่าความจริงของเงื่อนไขรวมได้ เช่น

- ถ้ามี **p && q** โปรแกรมจะตรวจ p ก่อน ซึ่งหาก p เป็นเท็จ สามารถสรุปได้เลยว่าเงื่อนไขเป็นเท็จ โดยที่ไม่ต้องพิจารณา q ในทางตรงข้าม หาก p เป็นจริง โปรแกรมก็จะต้องตรวจสอบค่าความจริงของ q ด้วย
- ถ้ามี **p || q** โปรแกรมจะตรวจ p ก่อน ซึ่งหาก p เป็นจริง สามารถสรุปได้เลยว่าเงื่อนไขเป็นจริงได้เลย โดยที่ไม่ต้องพิจารณา q ในทางตรงข้าม หาก p เป็นเท็จ โปรแกรมก็จะต้องตรวจสอบค่าความจริงของ q ด้วย



32



## 4. Short Circuit Evaluation

### ■ ตัวอย่าง: Short Circuit1

```
if ((count != 0) && (sum_scores/count < 35))  
    printf("What a low class average. Fire the professor!\n");
```

ถ้า count เป็น 0 โปรแกรมจะไม่ทดสอบเงื่อนไข  
ซึ่งก็จะไม่เกิด การหารด้วย 0 (Runtime error)

ถ้า count ไม่ใช่ 0 โปรแกรมจะทดสอบเงื่อนไข เช่น  
count มีค่า 10, sum\_scores มีค่า 200



33

## 4. Short Circuit Evaluation

### ■ ตัวอย่าง: Short Circuit2

```
if ((count <= 0) || (sum_scores/count > 100))  
    printf("Error: Invalid Result!\n");
```

ถ้า count น้อยกว่าหรือเท่ากับ 0 โปรแกรมจะ  
ไม่ทดสอบเงื่อนไขถัดไป และทำคำสั่ง ใน if เลย  
ทำให้ตรวจสอบเงื่อนไขน้อยลง

ถ้า count มากกว่า 0 โปรแกรมจะทดสอบเงื่อนไข  
ถัดไป ทำให้ต้องตรวจสอบทั้ง 2 เงื่อนไข



34