

Outline

- Basic Loop
- Loop Statement in C
 - for Statement
 - while Statement
 - do .. while Statement



Overview

- การพิจารณาเลขจำนวนเต็ม 1 ตัว ว่าเป็นเลขคี่ หรือไม่เขียนส่วนของโปรแกรมได้ดังนี้

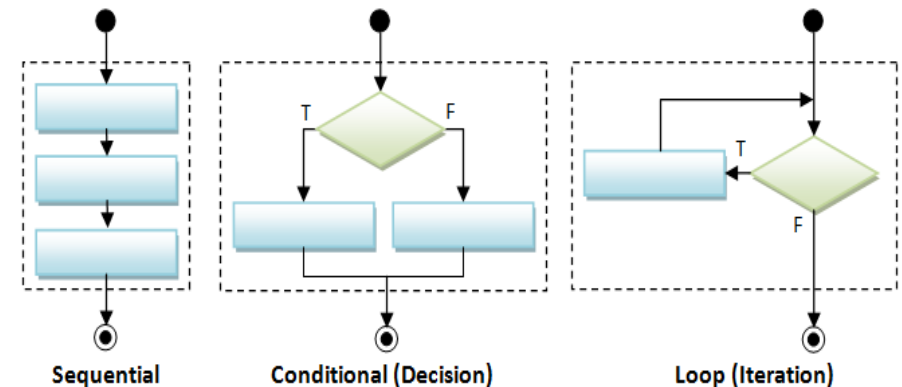
```
if (n%2 == 1)
    printf("%d",n);
```

- หากเราต้องการแสดงเลขคี่ที่มีค่า ตั้งแต่ 1 -100 จะต้องเขียนโปรแกรมอย่างไร

1,3,5,7,9, ... 99



Review: Flow control



โครงสร้างควบคุมหลักในการเขียนโปรแกรม

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



1. Basic Loop

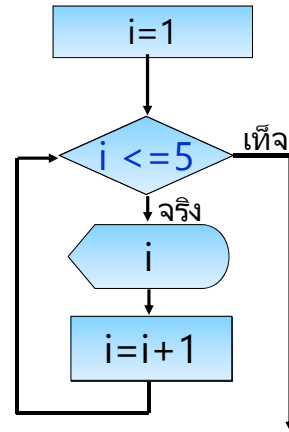
■ การทำซ้ำ (Looping or iteration)

การเขียนโปรแกรมที่ขั้นตอนการทำงานบางขั้นตอนได้รับ **การประมวลผลมากกว่า 1 ครั้ง** ทั้งนี้ขึ้นอยู่กับเงื่อนไขในการทำงานซ้ำ

1 2 3 4 5

ตัวอย่างอื่นๆ ที่ต้องใช้การทำซ้ำการทำงาน เช่น

- การคำนวณเฉลี่ยคะแนนสอบ ของนักเรียน 50 คน
- การสรุปยอดขายประจำวัน

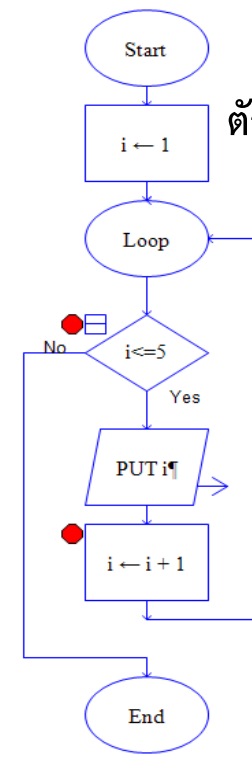


5

1. Basic loop

ตัวอย่าง 1: Flow chart ของการแสดงผลตัวเลขตั้งแต่ 1-5

ค่าของตัวแปร i	i <= 5 หรือไม่	คำสั่ง ทำซ้ำ	ผลทางจอภาพ
1	ใช่	แสดงค่า i i = 1+1	1
2	ใช่	แสดงค่า i i = 2+1	2
3	ใช่	แสดงค่า i i = 3+1	3
4	ใช่	แสดงค่า i i = 4+1	4
5	ใช่	แสดงค่า i i = 5+1	5
6	ไม่ใช่	จบการทำซ้ำ	



6

1. Basic loop

■ ลักษณะของลูปแบ่งได้เป็น 2 ประเภทหลัก เมื่อพิจารณาจากเงื่อนไข

— คำสั่งวนซ้ำแบบใช้ตัวนับ (Counter-controlled loop)

- มีจำนวนครั้งของการทำซ้ำเป็น **จำนวนที่แน่นอน** (อาจอยู่ในรูปของค่าคงที่หรือตัวแปร)

— คำสั่งวนซ้ำแบบใช้ตัวหยุด (Sentinel-controlled loop)

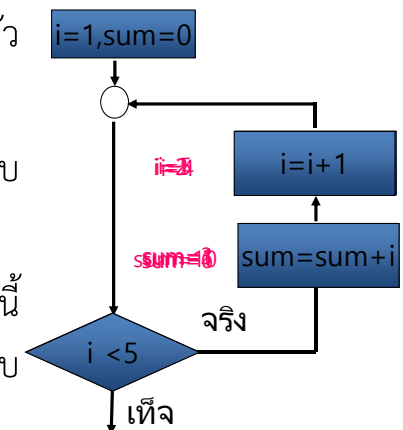
- มีจำนวนครั้งของการทำซ้ำ **ไม่แน่นอน**
- ทำซ้ำตราบใดที่เงื่อนไขบางอย่างยังคงเป็นจริง



7

1. Basic loop (Counter-controlled loop)

- การทำซ้ำที่การทำงานถูกควบคุมด้วยตัวแปรที่ทำหน้าที่เป็น **ตัวนับ (counter)**
- การทำซ้ำจะหยุดเมื่อโปรแกรมทำงานครบตามจำนวนรอบที่ต้องการ
- การเขียนโครงสร้างโปรแกรมในลักษณะนี้เป็นการแก้ปัญหาที่โปรแกรมเมอร์ทราบจำนวนการทำซ้ำที่แน่นอน



ผังงานนี้ มีการทำซ้ำกี่รอบ ?



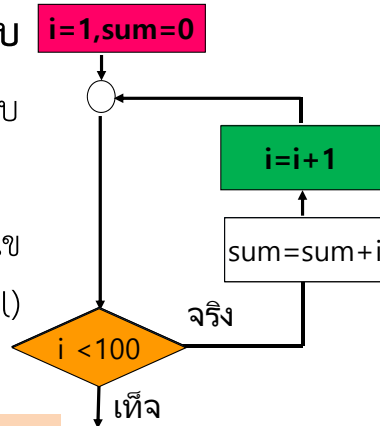
8

1. Basic loop (Counter-controlled loop)

■ รูปแบบทั่วไป: คำสั่งวนซ้ำแบบใช้ตัวนับ

- กำหนดค่าเริ่มต้น ของตัวแปรที่เป็นตัวนับ (initialize counter)
- ตรวจสอบค่าตัวนับ ว่าเป็นไปตามเงื่อนไขการทำซ้ำหรือไม่ (Testing loop control)
- ปรับค่าตัวนับ (Updating counter)

เช่น $i = i + 1$ หรือ $i++$
 $num = num - 5$ หรือ $num -= 5$

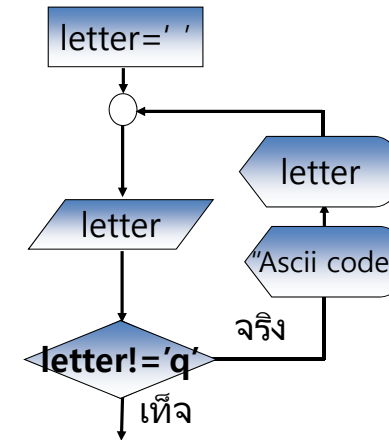


9

1. Basic loop (Sentinel-controlled loop)

- ใช้การกำหนดเงื่อนไขในการควบคุมการทำซ้ำ
- จะหยุดการทำงานเมื่อพบค่าบางอย่างตรงตามเงื่อนไขหยุดการทำซ้ำ

letter
a
b
m
q




Ascii code: 97
Ascii code: 98
Ascii code: 109

End Loop!!



10

Quick check1

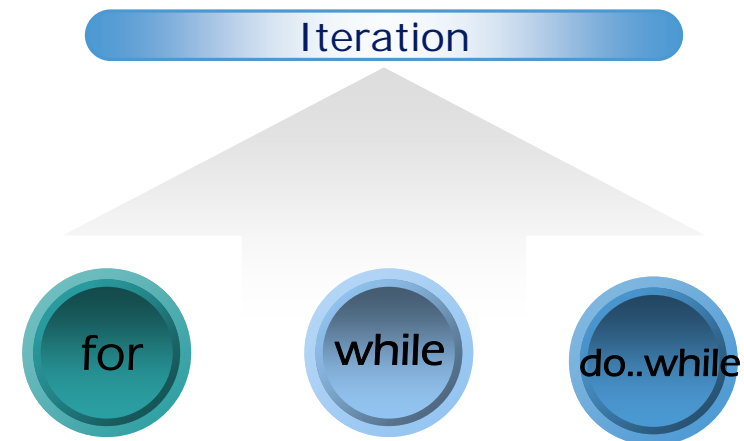
■ จงเขียนเงื่อนไขของลูปซึ่งระบุในสัญลักษณ์  ของผังงาน เป็นภาษา C (สร้างตัวแปรได้เองเลย)

- ให้ผู้ใช้ป้อนค่าไปเรื่อยๆ จนกว่าจะเจอตัวอักษร 'q' หรือ 'Q'
- ให้ผู้ใช้ป้อนเลขไปเรื่อยๆ จนกว่าจะเป็นเลขที่น้อยกว่า 0
- ให้รับค่าคะแนนวิชาคอมพิวเตอร์ของนิสิต 100 คน
- ให้ผู้ใช้ป้อนเลขไปเรื่อยๆจนกว่าจะเป็นเลขที่หาร 17 ลงตัว



11

2. Loop Statement in C



12

2.1 for statement

- รูปแบบ for statement ประกอบด้วย 3 ส่วน

```
for (คำสั่งเริ่มต้น; เงื่อนไข; คำสั่งปรับค่า)
{
    คำสั่ง;
    คำสั่ง;
}
```

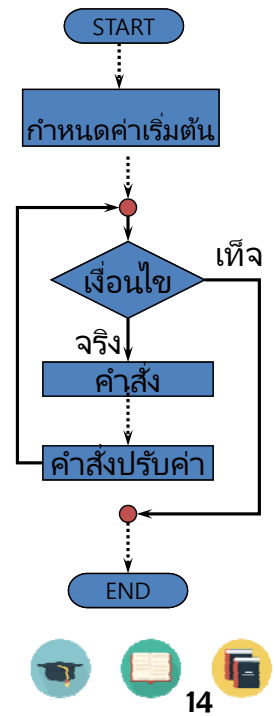
- คำสั่งเริ่มต้น** กำหนดค่าเริ่มต้นให้กับตัวนับรอบ
- เงื่อนไข** เป็นนิพจน์เงื่อนไขการนับรอบ การวนลูปจะทำงานกว่าเงื่อนไขเป็นเท็จ
- คำสั่งปรับค่า** เป็นนิพจน์การปรับค่าตัวนับรอบ
- คำสั่ง** เป็นคำสั่งที่ให้ทำในแต่ละรอบซึ่งเป็นคำสั่งเดี่ยว หรือคำสั่งประกอบ



2.1 for statement

- การทำงาน

- กำหนดค่าเริ่มต้นให้กับตัวแปรนับรอบ
- ทดสอบเงื่อนไขของตัวนับรอบ
 - ถ้าเงื่อนไขเท็จให้ทำข้อ 5
 - ถ้าเงื่อนไขจริงให้ทำข้อ 3
- ทำคำสั่งในลูป
- เพิ่มค่า (หรือลดค่า) ให้กับตัวนับ แล้วไปทำในข้อ 2
- จบการวนซ้ำ



2.1 for statement

- ตัวอย่างที่ 2

คำสั่งเริ่มต้น เงื่อนไข คำสั่งปรับค่า

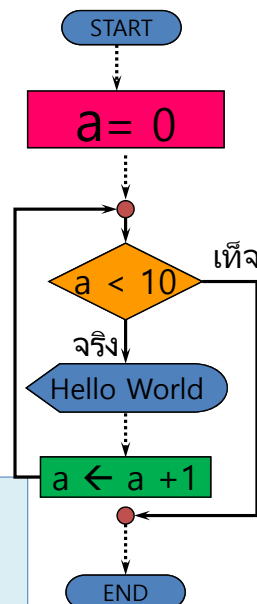
```
for(a=0; a < 10; a++)
    printf ("Hello World\n");
```

ประมวลผลเพียงครั้งเดียว

Hello World
Hello World
Hello World
Hello World
Hello World
Hello World
Hello World
Hello World
Hello World

a: 0
a: 1
a: 2
a: 3
a: 4
a: 5
a: 6
a: 9
a: 10

เริ่มต้นทำงาน กำหนดให้ a=0
ตรวจเท่าที่ a มีค่าน้อยกว่า 10
จะพิมพ์คำว่า Hello World และ เพิ่มค่า a ขึ้นทีละ 1



2.1 for statement

- ตัวอย่างที่ 3

```
for(a=5; a <=25; a+=5)
    printf ("%d", a);
```

เริ่มต้นทำงาน กำหนดให้ a=5
ตรวจเท่าที่ a มีค่าน้อยกว่าเท่ากับ 25
จะแสดงค่า a และ เพิ่มค่า a ขึ้นทีละ 5

ค่า a	a <= 25 หรือไม่	คำสั่ง ทำซ้ำ	ผลทางจอภาพ
5	ใช่	แสดงค่า a a = 5+5	5
10	ใช่	แสดงค่า a a = 10+5	5,10
15	ใช่	แสดงค่า a a = 15+5	5,10,15
20	ใช่	แสดงค่า a a = 20+5	5,10,15,20
25	ใช่	แสดงค่า a a = 25+5	5,10,15,20,25
30	ไม่ใช่	จบการทำซ้ำ	



Quick check2

- จงเติมส่วนของโปรแกรมสำหรับแสดงค่า i^2 นี้ ให้สมบูรณ์โดย i มีค่าตั้งแต่ 1 -10

```
for(
    printf("%d\n", i*i);
)
```

1	1
2	4
3	9
4	16
...	...
10	100
11	

End Loop!!

โปรแกรมนี้ มีจำนวนครั้งในทำซ้ำกี่รอบ ?

for statement เหมาะกับรูปแบบใด ?



17

2.1 for statement

for Statements	ผลทาง จลภาพ
<pre>for(x=1;x<0;x*=2) printf("%d\n",x);</pre>	
<pre>for(x=1;x<=10;x--) printf("%d\n",x);</pre>	
<pre>for(x=10;x>0;x/=2) printf("%d ",x); printf("\n%d",x*5);</pre>	
<pre>for(x=10;x>0;x/=2) { printf("%d ",x); printf("%d\n",x*5); }</pre>	

ข้อควรระวัง

for statement ตรวจสอบเงื่อนไขก่อนทำงานในลูป หากเงื่อนไขเป็นเท็จจะไม่ทำ

การปรับค่าตัวนับ ที่ทำให้เงื่อนไขเป็นจริงเสมอ จะทำให้เกิดการวนซ้ำไม่สิ้นสุด เรียกว่า ลูปอนันต์ (infinite loop)

หากคำสั่งที่ต้องการทำซ้ำมีมากกว่า 1 คำสั่ง ควรใส่ปีกกา { }



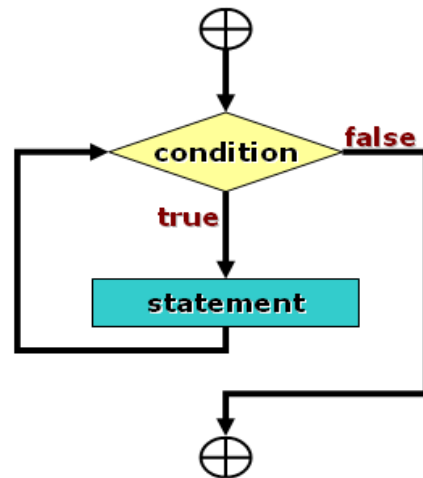
18

2.2 while statemen'

- รูปแบบไวยากรณ์ (syntax)

while (เงื่อนไข)
คำสั่ง;

```
while (เงื่อนไข)
{
    คำสั่ง 1
    คำสั่ง 2
    ...
}
```



ทำงาน คำสั่ง ซ้ำ ไปเรื่อยๆ
ตราบเท่าที่เงื่อนไขเป็น “จริง”



19

2.2 while statement

- ตัวอย่าง 4

```
int main() {
    int i;
    i = 0;

    while (i < 5) {
        printf ("CPE-%d\n", i);
        i++;
    }
}
```

0	CPE-0
1	CPE-1
2	CPE-2
3	CPE-3
4	CPE-4
5	

End Loop!!

การทำงานของ while statement: ตรวจสอบเงื่อนไขก่อนทำคำสั่งซ้ำ และทำซ้ำไปเรื่อยๆ ตราบเท่าที่เงื่อนไขเป็นจริง



20

2.2 while statement

ตัวอย่าง 4

```
int main() {
    int i, n;
    i = 0;
    scanf("%d", &n);

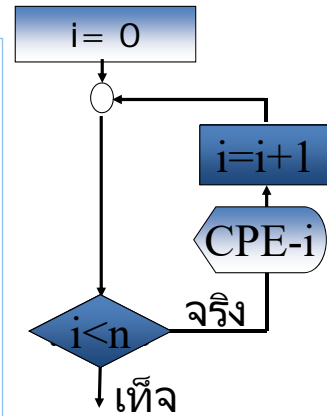
    while (i < n) {
        printf("CPE-%d\n", i);
        i++;
    }
}
```

$n=20$

0
1
2
...
19
20

CPE-0
CPE-1
CPE-2
...
CPE-19

End Loop!!



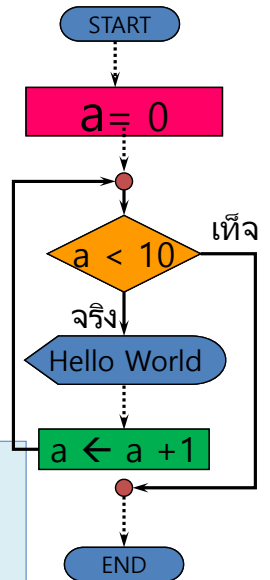
21

2.2 while statement

ตัวอย่าง 5

```
a = 0;
while (a < 10)
{
    printf("Hello World\n");
    a = a + 1;
}
```

เริ่มต้นทำงาน กำหนดให้ a=0
 ตรวจสอบค่าที่ a มีค่าน้อยกว่า 10
 จะพิมพ์คำว่า Hello World และ เพิ่มค่า a ขึ้นทีละ 1



22

2.2 while statement

ตัวอย่าง 5

```
a = 0;
while (a < 10)
{
    printf("Hello World\n");
    a = a + 1;
}
```

คำสั่งเริ่มต้น เงื่อนไข คำสั่งปรับค่า

```
for(a=0; a < 10; a++)
    printf("Hello World\n");
```

ส่วนคำสั่งปรับค่าอยู่ในลูป

คำสั่งวนซ้ำแบบใช้ตัวนับ สามารถเขียนโปรแกรมได้
 โดยใช้ while และ for statement



23

2.2 while statement

ตัวอย่าง 6

```
code
1 a = 1;
2 while(a < 10) {
3     printf("%d\n", a);
4     a += 2;
}
```

output

```
for(a=1; a < 10; a+=2)
    printf("%d\n", a);
```

variables



24

2.2 while statement

- ตัวอย่าง 7: เกมทายตัวอักษร ผู้ใช้จะต้องใส่ตัวอักษรไปเรื่อยๆ จนกว่าจะทายถูก

```
int main() {  
    char letter, ans = 'T';  
    scanf("%c", &letter);  
  
    while (letter != ans)  
    {  
        printf("Try again\n");  
        scanf("%c", &letter);  
    }  
  
    printf("That's correct, Loop End");  
}
```

letter

a
b
m
t
T

Try again
Try again
Try again
Try again

That's correct, Loop End

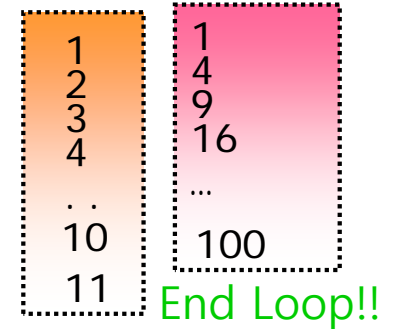


25

Quick check3

- จงเขียนส่วนของโปรแกรมแสดงค่า i^2 โดย i มีค่าตั้งแต่ 1 -10

```
while( )  
{  
    printf("%d\n", i*i);  
}
```



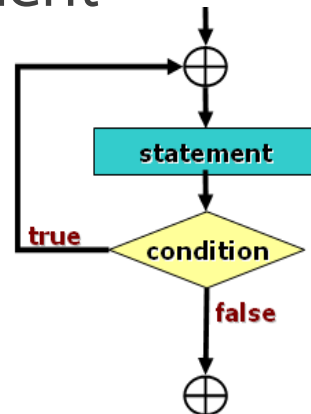
26

2.3 do... while statement

- รูปแบบไวยากรณ์ (syntax)

```
do  
    คำสั่ง;  
while (เงื่อนไข);
```

```
do  
{  
    คำสั่ง 1  
    คำสั่ง 2  
    ...  
} while (เงื่อนไข);
```



ทำงาน คำสั่ง (statement)
แล้วทดสอบ เงื่อนไข (condition)
และ ทำซ้ำถ้าเงื่อนไขยังคงเป็นจริง



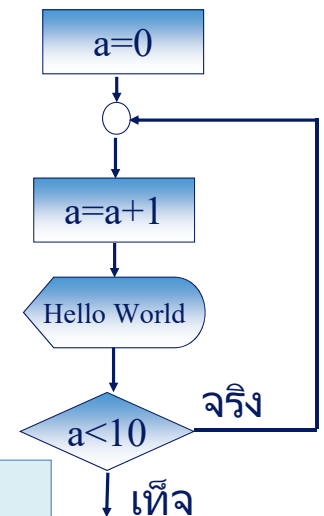
27

2.3 do... while statement

- ตัวอย่าง 8

```
a = 0;  
do  
{  
    printf("Hello World\n");  
    a = a + 1;  
}  
while (a < 10);
```

เริ่มต้นทำงาน กำหนดให้ a=0
พิมพ์คำว่า Hello World และ เพิ่มค่า a ขึ้นทีละ 1
ตราบเท่าที่ a มีค่าน้อยกว่า 10



28

2.3 do... while statement

ตัวอย่าง 9: while vs do...while

```
char letter;

do
{
    print("Enter q to quit: ");
    scanf("%c", &letter);
} while(letter != 'q');
```

Output 1:

Enter q to quit: *a*
 Enter q to quit: *B*
 Enter q to quit: *q*

```
char letter;
print("Enter q to quit: ");
scanf("%c", &letter);

while (letter != 'q')
{
    print("Enter q to quit: ");
    scanf("%c", &letter);
}
```

Output 2:

Enter q to quit: *q*



29

2.3 do... while statement

ตัวอย่าง 10: while vs do...while

x=25

x=9

```
while (x >= 10)
    x = sqrt(x);
```

```
do
    x = sqrt(x);
while (x >= 10);
```

ตรวจสอบค่าที่ *x* มีค่ามากกว่าหรือเท่ากับ 10
 ทำการกำหนดค่า *x* เท่ากับรากที่สองของ *x*

กำหนดค่า *x* เท่ากับรากที่สองของ *x*
 ตรวจสอบค่าที่ *x* มีค่ามากกว่าหรือเท่ากับ 10
 ทำคำสั่งเดิมซ้ำ



30

2.3 do... while statement

do...while = 1 + while

— while จะมีการเช็คเงื่อนไขก่อนเข้าสู่รอบแรก แต่ do..while ไม่ต้องเช็ค

ผลลัพธ์ของ while จะเหมือน do..while เมื่อเรามั่นใจได้ว่า while จะต้องเข้าเงื่อนไขตั้งแต่รอบแรก เช่น

— กำหนด *a* = 0 และมีเงื่อนไข *a* < 10 → ไม่ว่าอย่างไรก็เข้าสู่ while
 — ฉะนั้นผลลัพธ์จะเหมือนกับ do..while คือ เปรียบเสมือนว่าไม่ต้องเช็คเงื่อนไขตอนเข้าสู่รอบแรก



31

2.3 while vs do... while statement

while Statements/ do ..while Statements	ผลทาง จลภาพ
<i>x</i> =1; while (<i>x</i> <0){ printf("%d\n", <i>x</i>); <i>x</i> *=2; }	
<i>x</i> =-1; while (<i>x</i> <0) printf("%d\n", <i>x</i>); <i>x</i> *=2;	
<i>x</i> =-1; do printf("%d\n", <i>x</i>); <i>x</i> *=2; while(<i>x</i> <0);	

ข้อควรระวัง

while statement ตรวจสอบเงื่อนไขก่อนทำงานในลูป หากเงื่อนไขเป็นเท็จจะไม่ทำ

หากคำสั่งที่ต้องการทำซ้ำมีมากกว่า 1 คำสั่ง ควรใส่ปีกกา { }



32

Summary:

- ลักษณะของลูปแบ่งได้เป็น 2 ประเภทหลัก เมื่อพิจารณาจากเงื่อนไข
 - คำสั่งวนซ้ำแบบใช้ตัวนับ (Counter-controlled repetition)
 - มีจำนวนครั้งของการทำซ้ำเป็นจำนวนที่แน่นอน
(อาจอยู่ในรูปของค่าคงที่หรือตัวแปร)
 - นิยมใช้: for loop
 - คำสั่งวนซ้ำแบบใช้ตัวหยุด (Sentinel-controlled repetition)
 - มีจำนวนครั้งของการทำซ้ำไม่แน่นอน
 - ทำซ้ำตราบใดที่เงื่อนไขบางอย่างยังคงเป็นจริง
 - นิยมใช้: while loop และ do...while loop



33

Quick check4

- จงบอกผลลัพธ์ของชุดคำสั่งต่อไปนี้

a.

```
int j = 1;
while ( j <= 9){
    printf("%d\n", j);
    j  += 2;
}
```

c.

```
int j = 0;
do{
    printf("%d\n",j);
    j  += 2;
}while ( j <= 8);
```

b.

```
int j = 1;
while ( j < = 9){
    j += 2;
    printf("%d\n",j);
}
```

d.

```
int j = 0;
do
{
    j += 2;
    printf("%d\n",j);
}while( j <= 8);
```



34

Quick check4

- จงบอกผลลัพธ์ของชุดคำสั่งต่อไปนี้

e.

```
int j;
for(j = 1; j <= 9; j  += 2)
    printf("%d\n",j);
```

f.

```
int j;
for(j=9;j > = 1; j-=2)
    printf("%d\n",j);
```



35