02204171 Structured Programming

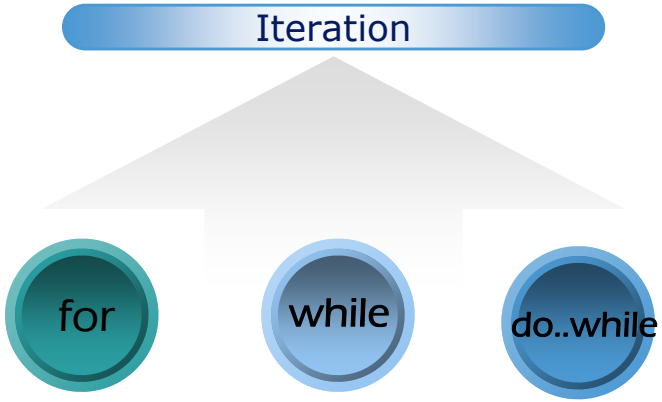
Chapter 7 : Advanced Loop control

Computer Engineering, Kasetsart University Kamphaeng Sean Campus

Outline

- Review: loop statements
- loop statement with if statement
- Interrupting Loop Flow:
  - break statements
  - continue statements
- Nested loop

1. Review: Loop Statement in C



1. Review: Loop Statement in C

- for และ while มีการประมวลผลแบบเดียวกัน คือ มีการตรวจสอบเงื่อนไขก่อนการทำซ้ำ
- do...while ต่างจากทั้งสองคำสั่งคือ มีการทำคำสั่งก่อนแล้วจึงตรวจสอบเงื่อนไขการทำซ้ำ
  - do...while จะทำคำสั่งแน่นอนอย่างน้อย 1 รอบ

1. Review: Loop Statement in C

- ตัวอย่าง 1: while/ for statements

```
int x = 4;
while (x > 0)
{
  printf("%d ",x);
  x /= 2;
}
```

```
int x;
for (x = 4;x > 0;x /= 2)
  printf("%d ",x);
```

```
bool x = 0, y = false;
int i = 0;
while (x = !y)
{
   printf("%d\n",i);
   i++;
   y = !(y||x);
}
```


1. Review: Loop Statement in C

- ตัวอย่าง 1: do .... while statements

```
int x = 4;
do
{
  printf("%d ",x);
  x /= 2;
}
while (x > 0);
```

```
bool x = 0, y = false;
int i = 0;
do{
   printf("%d\n",i);
   i++;
   y = !(y||x);
}
while (x = !y);
```


2. Loop statement with if statement

- ในการแก้ไขปัญหาหนึ่งอาจจำเป็นต้องใช้ โครงสร้างควบคุมหลักในการเขียนโปรแกรมทุกแบบ
  - **โปรแกรมคำนวณผลรวมของเลขคี่ตั้งแต่ 1 จนถึง n**
    - คำสั่ง if ภายใต้ loop
  - **โปรแกรมทายตัวเลข**
    - คำสั่ง if ภายใต้ loop
  - **โปรแกรมคำนวณค่าแฟคทอเรียลของจำนวนเต็มบวก**
    - คำสั่ง loop ภายใต้ if

2. Loop statement with if statement

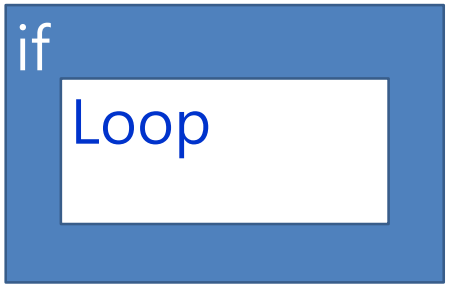
- ตัวอย่าง 2: โปรแกรมคำนวณผลรวมของเลขคี่ตั้งแต่ 1 จนถึง n

```
i = 1;
sum=0;
while (i <= n)
{

  sum=sum+i;
  i = i + 1;
}
```

2. Loop statement with if statement

- **ตัวอย่าง 3:** โปรแกรมเพื่อคำนวณหาผลบวกของเลขจำนวนเต็มตั้งแต่ 0 - n โดยที่ n>=0 โดยให้รับค่า n ทางคีย์บอร์ด
  - n >= 0 หาผลบวก
  - n < 0 แสดงข้อความ Invalid input!



2. Loop statement with if statement

- ตัวอย่าง 3

```
int i, n;
scanf("%d",&n);
if (n >= 0)
{
  int sum = 0;
  for (i = 0; i<=n; i++)
  {
    sum = sum + i;
  }
  printf("%d",sum);
}
else
  printf("Invalid input!\n");
```


3. break and continue statements

- **คำสั่ง break** ใช้ในการออกจาก block ซึ่งมักจะใช้ในคำสั่ง switch...case และคำสั่ง loop

รูปแบบ

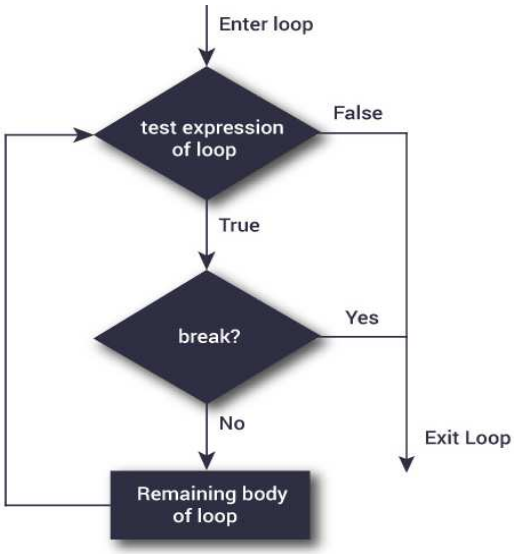
```
for (…)
{
 …
 if(expression)
    break;
 …
}
```

```
while(…)
{
 …
 if(expression)
   break;
 …
}
```

```
do
{
 …
 if(expression)
  break;
 …
}
while(…);
```


3. break and continue statements

- ผังงานของ break statements




```
for (…)
{
 …
 if(expression)
    break;
 …
}
```

```
while(…)
{
 …
 if(expression)
   break;
 …
}
```

3. break and continue statements

- ตัวอย่าง 4: break statement

```
int i;
for (i = 1;; i++)
{
   printf("%d\n",i);
   if (i == 4)
      break;
}
```


3. break and continue statements

- ตัวอย่าง 5: break statement

```
char letter = 'd';
while( letter <= 'n' )
 {
    printf("%c ",letter);
    if(letter == 'k' )
         break;
    letter++;
 }
```

โปรแกรมนี้ทำอะไร?

Quick check1

- จงเขียนโปรแกรมที่รับจำนวนเต็ม 2 ตัวเรื่อยๆ แล้วแสดงผลหารและเศษของเลข 2 ตัวนี้ และให้**หยุดการทำงานเมื่อผู้ใช้ป้อนตัวที่ 2 เป็น 0**

```
int a,b;
do {
  scanf("%d%d",&a,&b);
  if(_________)
  _____________
  else
    printf("%d %d\n",(a/b),(a%b));
} while(_____________);
```


3. break and continue statements

- **คำสั่ง continue** ใช้ในการข้ามการทำงานคำสั่งใน loop แล้วขึ้น loop รอบใหม่

รูปแบบ

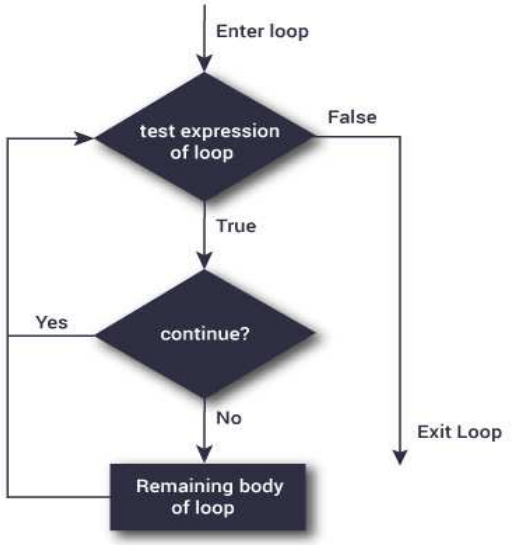
```
for (…)
{
 …
 if(expression)
    continue;
 …
}
```

```
while(…)
{
 …
 if(expression)
    continue;
 …
}
```

```
do
{
 …
 if(expression)
   continue;
 …
}
while(…);
```

3. break and continue statements

- ผังงานของ continue statements




```
for (…)
{
 …
 if(expression)
    continue;
 …
}
```

```
while(…)
{
 …
 if(expression)
    continue;
 …
}
```


3. break and continue statements

- ตัวอย่าง 6: continue statement

```
int i;
for (i = 1;; i++)
{  if (i == 4)
     continue;
   printf("%d ",i);
   if (i == 8)
      break;
}
```


3. break and continue statements

- ตัวอย่าง 7: continue statement

```
char letter;
for(letter = 'd'; letter <= 'k'; letter++)
 {
    if( letter == 'i' )
         continue;
    printf("%c ",letter);
 }
```


Quick check2

- จงเขียนโปรแกรมที่รับจำนวนเต็ม 2 ตัวเรื่อยๆ แล้วตรวจสอบว่าเลขที่ใส่มีเครื่องหมายเดียวกันหรือไม่ (เลขบวกทั้งคู่ หรือ ลบทั้งคู่) หากเหมือนกันให้แสดงผลคูณของเลขทั้งสอง แต่หากต่างกันไม่ต้องทำอะไร และให้หยุดการทำงานเมื่อผู้ใช้ป้อนตัวใดตัวหนึ่ง เป็น 0

```
int a,b;
do {
  scanf("%d%d",&a,&b);
  if(_____________)
    break;
  else if(_____________________________)
  _____________
  printf("%d\n",(a*b));
} while(_____________);
```

4. Nested loop

- **คำสั่งวนซ้อน (Nested Loop)** คือ การใช้คำสั่งวนซ้ำ **ซ้อนกัน** ใช้ช่วยในงานที่คำสั่งวนซ้ำรอบเดียวทำไม่ได้หรือทำได้ลำบาก เช่น การประมวลผลข้อมูลประเภทตาราง, อะเรย์ 2 มิติ

ตัวอย่าง: โปรแกรมแสดง * เป็นรูปสี่เหลี่ยมจัตุรัส 3x3

```
***
***
***
```

```
int j;
for(j=0;j<3;j++)
  printf("*");
printf("\n");
for(j=0;j<3;j++)
  printf("*");
printf("\n");
for(j=0;j<3;j++)
  printf("*");
```

```
* * *
* * *
* * *
```

```
int i,j;
for(i=0;i<3;i++)
{
  for(j=0;j<3;j++)
    printf("*");
  printf("\n");
}
```


4. Nested loop

- **ตัวอย่าง 8:** ตารางสูตรคูณตั้งแต่ 2 x 1 ถึง 8 x 8

```
int i, j;
i = 2;
while (i <= 8)
{
  j = 1;
  while (j <= 8)
  {
   printf(" %2d",i*j);
   j = j + 1;
  }
  printf("\n");
  i = i + 1;
}
```

i	j	Output
2		
3		

4. Nested loop

- **ตัวอย่าง 8:** ตารางสูตรคูณตั้งแต่ 2 x 1 ถึง 8 x 8

```
 2  4  6  8 10 12 14 16
 3  6  9 12 15 18 21 24
 4  8 12 16 20 24 28 32
 5 10 15 20 25 30 35 40
 6 12 18 24 30 36 42 48
 7 14 21 28 35 42 49 56
 8 16 24 32 40 48 56 64
```

เขียนโปรแกรมนี้โดยใช้ for loop

4. Nested loop

- **ตัวอย่าง 9:** โปรแกรมสำหรับพิมพ์สามเหลี่ยมดังรูป

```
*
**
***
****
```

```
int i,j;
for(i=1;i<=4;i++)
{
  for(   ;    ;    )
     printf("*");
  printf("\n");
}
```

i	j	Output
1		
2		
3		
4		
5		

จำนวนครั้งที่ลูปในประมวลผลขึ้นกับเงื่อนไขของลูปนอก

Quick check 3

- **จงเติมคำสั่ง**โปรแกรมสำหรับพิมพ์สามเหลี่ยมด้วยตัวเลข 1 ถึง n

n = 5

```
1
12
123
1234
12345
```

```
int i,j,n;
for(i=1;_______;i++)
{
  for(_____;_____;_____)
    ____________________
 printf("\n");
}
```

i	j	Output
1		
2		
3		
4		
5		

4. Nested loop

- **ตัวอย่าง 10:** โปรแกรมสำหรับพิมพ์สามเหลี่ยมด้วยตัวเลข 1 ถึง n

n = 5

```
12345
1234
123
12
1
```

```
int i,j,n;
scanf("%d",&n);
for(i=1;i<=n;i++)
{
 for(     ;                ;       )
    printf("%d",j);
 printf("\n");
}
```

i	j
1	1 2 3 4 5
2	1 2 3 4
3	1 2 3
4	1 2
5	1

4. Nested loop

- **ข้อควรระวัง:** ตัวแปรที่เป็นเงื่อนไข ของลูปใน และลูปนอก ควรจะเป็นคนละตัวกัน

```
for(x=0;x<5;x++)
  for(x=5;x>0;x--)
    printf("%d",x);
```

```
for(x=0;x<5;x++)
{
  for(y=5;y>0;y--)
    printf("%d",x);
  printf("\n");
}
```


Quick check4

- (a) จงหาจำนวนครั้งที่ลูปในสุดประมวลผล

```
int x,y,z;
for (x=1;x<=10;x++)
  for (y=1;y<=10;y++)
    for (z=1;z<=10;z++)
      printf("%d\n", x+y+z);
```

- (b) จงหาผลลัพธ์ของส่วนของโปรแกรมต่อไปนี้

```
int outCount, inCount;
for(outCount = -1;outCount<2;outCount++)
  for (inCount=3;inCount>0;inCount--)
    printf("%d\n", outCount + inCount);
```

4. Nested loop

- break และ continue ใน nested loop

break;
ใช้เมื่อต้องการควบคุม
การจบออกจาก loop

continue;
ใช้เมื่อต้องการควบคุม
ให้ไปที่จุดเริ่มต้นของ loop

Break และ continue จะมีผลต่อ loop เพียงชั้นเดียวเท่านั้น

4. Nested loop

- break และ continue ใน nested loop

```
for(i=1;i<=5;i++)
{
  for(j=1;j<=i;j++)
     printf("*");
 printf("\n");
}
```

```
*
**
***
****
*****
```

```
for(i=1;i<=5;i++)
{
  for(j=1;j<=i;j++)
  {
    if (i==3)
      break;
    printf("*");
  }
  printf("\n");
}
```

ผลลัพธ์ของโปรแกรมนี้ คือ

4. Nested loop

- break และ continue ใน nested loop

```
for(i=1;i<=5;i++)
{
  for(j=1;j<=i;j++)
    printf("%d",j);
 printf("\n");
}
```

```
1
12
123
1234
12345
```

```
for(i=1;i<=5;i++)
{
  for(j=1;j<=i;j++)
  {
    if (j==3)
      continue;
    printf("%d",j);
  }
 printf("\n");
}
```