โครงสร้างของโปรแกรมและการควบคุม

Programming Structure and Control Flow

4

เอกสารประกอบการสอนวิชา CP111

มหาวิทยาลัยศรีนครินทรวิโรฒ

จัดทำโดย อ. เอก ตังคนานนท์
สไลด์ในบทนี้บางส่วนได้รับการปรับปรุงจากข้อเสนอแนะของอาจารย์สาโรช เมาลานนท์

เนื้อหา

Agenda

- โครงสร้างของโปรแกรม
- เงื่อนไข if-else
- การวนซ้ำด้วย for
- การวนซ้ำด้วย while, do-while
- การเลือกทำด้วย switch-case

โครงสร้างของโปรแกรม

Programming Structure

โครงสร้างของโปรแกรม

Programming Structure

ex04_01.c

```
#include<stdio.h>
#define PI 3.1415
int gSerial = 2518325;
float square(float f);

int main
{
   float r;
   float area;

   scanf("%f", &r);
   area = PI * r * r;
   printf("Area : %f", area);

   return 0;
}

float square(float f)
{
   return f * f;
}
```

การประกาศค่า (declaration)

บล๊อกคำสั่ง (statement block)

บล๊อกคำสั่ง (statement block)

โครงสร้างของโปรแกรม

Programming Structure

ex04_01.c

```
...

int main
{
   float r;
   float area;

   scanf("%f", &r);
   area = PI * r * r;
   printf("Area : %f", area);

   return 0;
}

...
```

บล็อกคำสั่ง

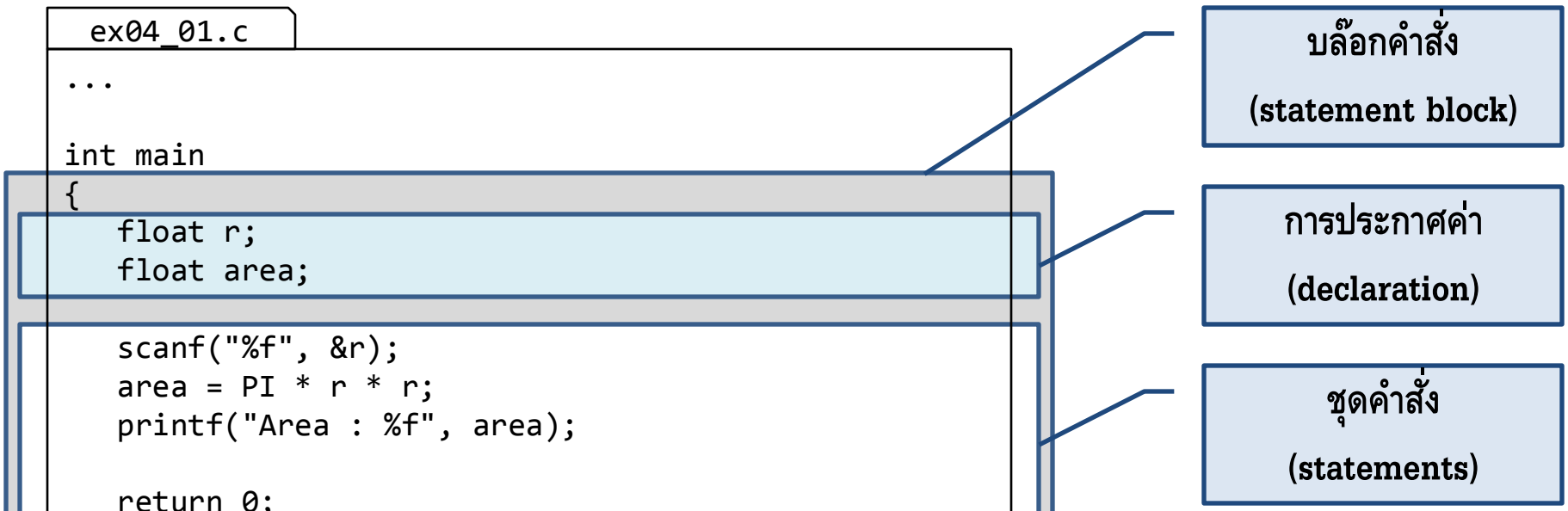
(statement block)

โครงสร้างของโปรแกรม

Programming Structure

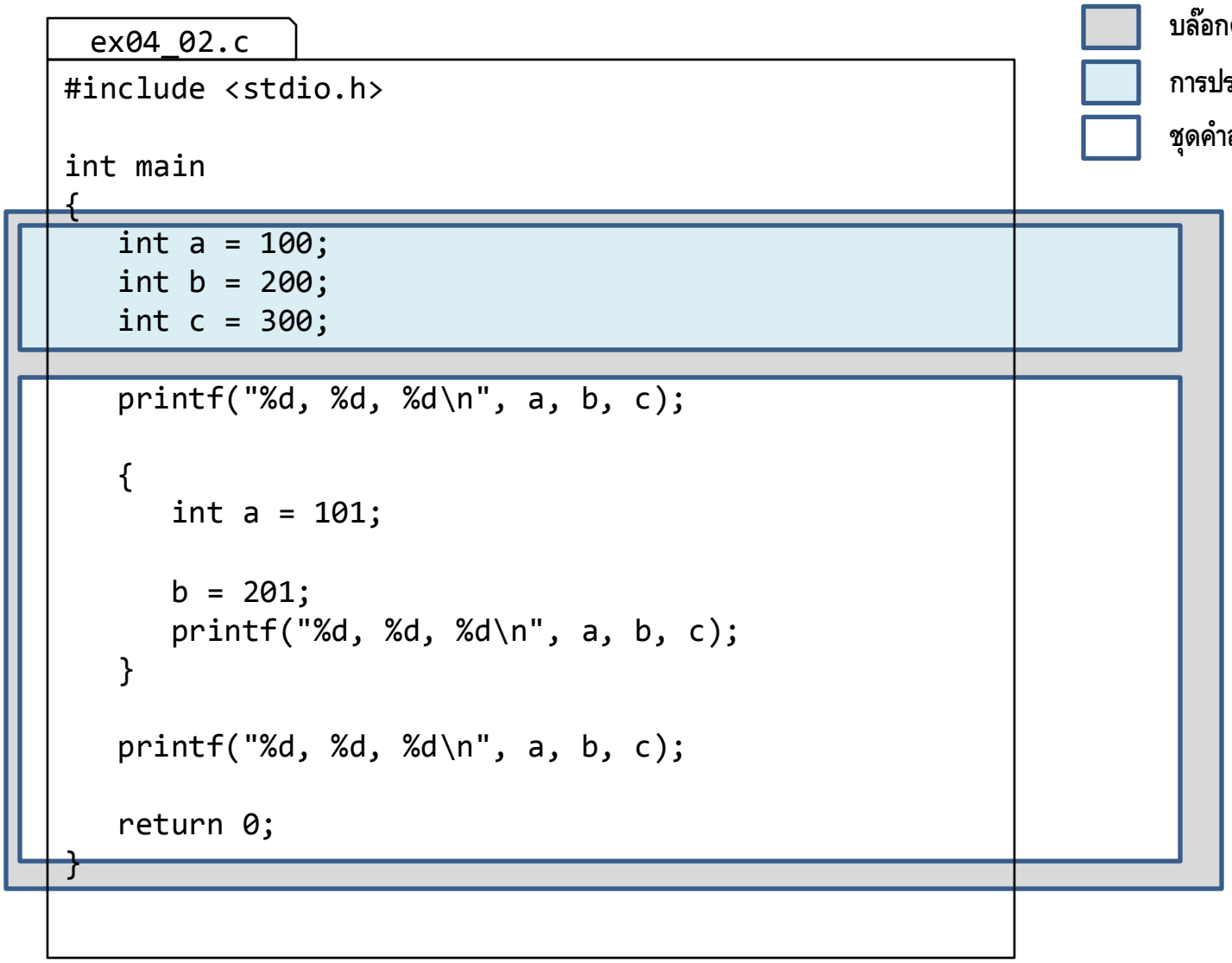
ex04_01.c

```
...

int main
{
   float r;
   float area;

   scanf("%f", &r);
   area = PI * r * r;
   printf("Area : %f", area);

   return 0;
}

...
```



โครงสร้างของโปรแกรม

Programming Structure



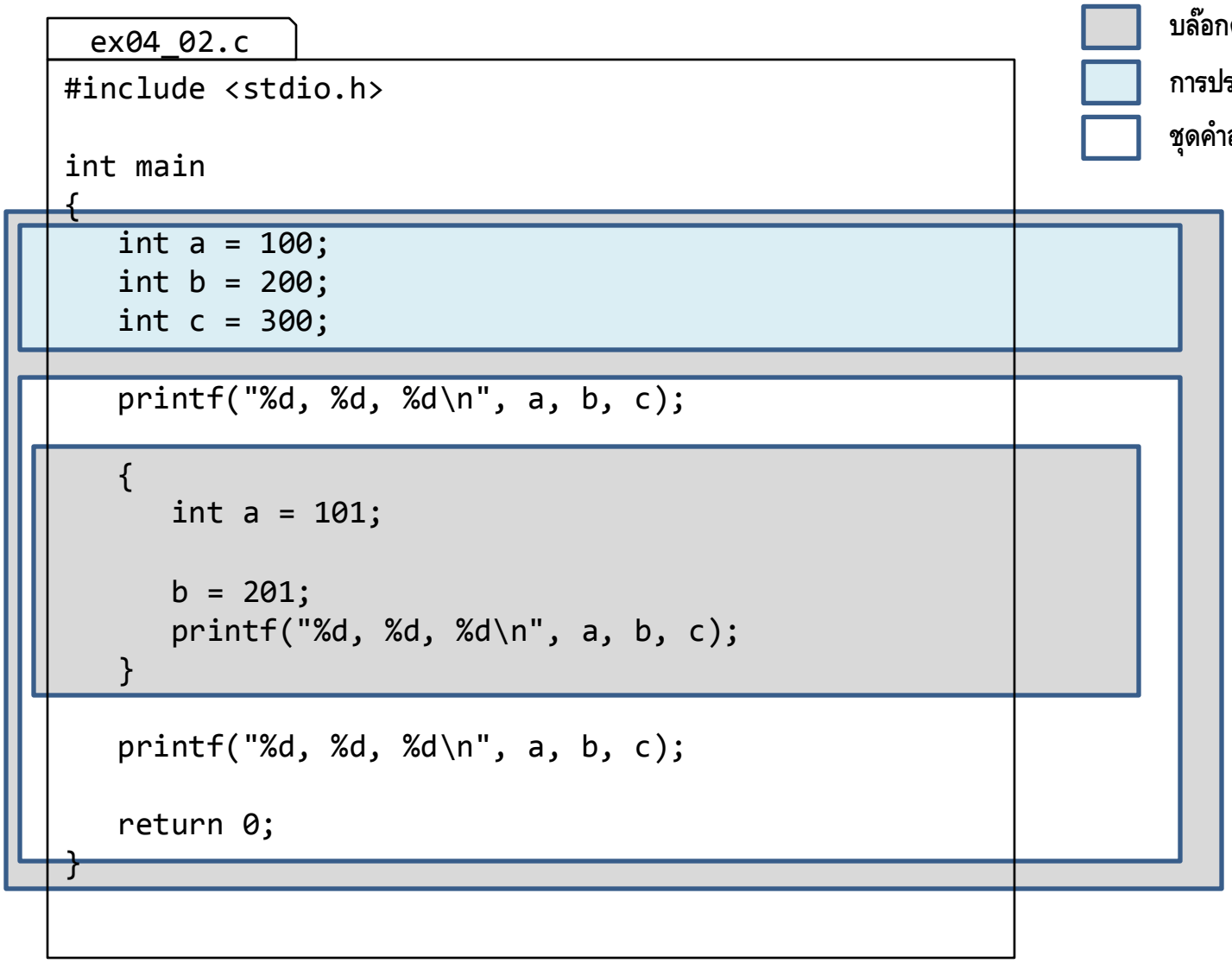
บล๊อกคำสั่ง (statement block)

การประกาศค่า (declaration)

ชุดคำสั่ง (statements)

โครงสร้างของโปรแกรม

Programming Structure



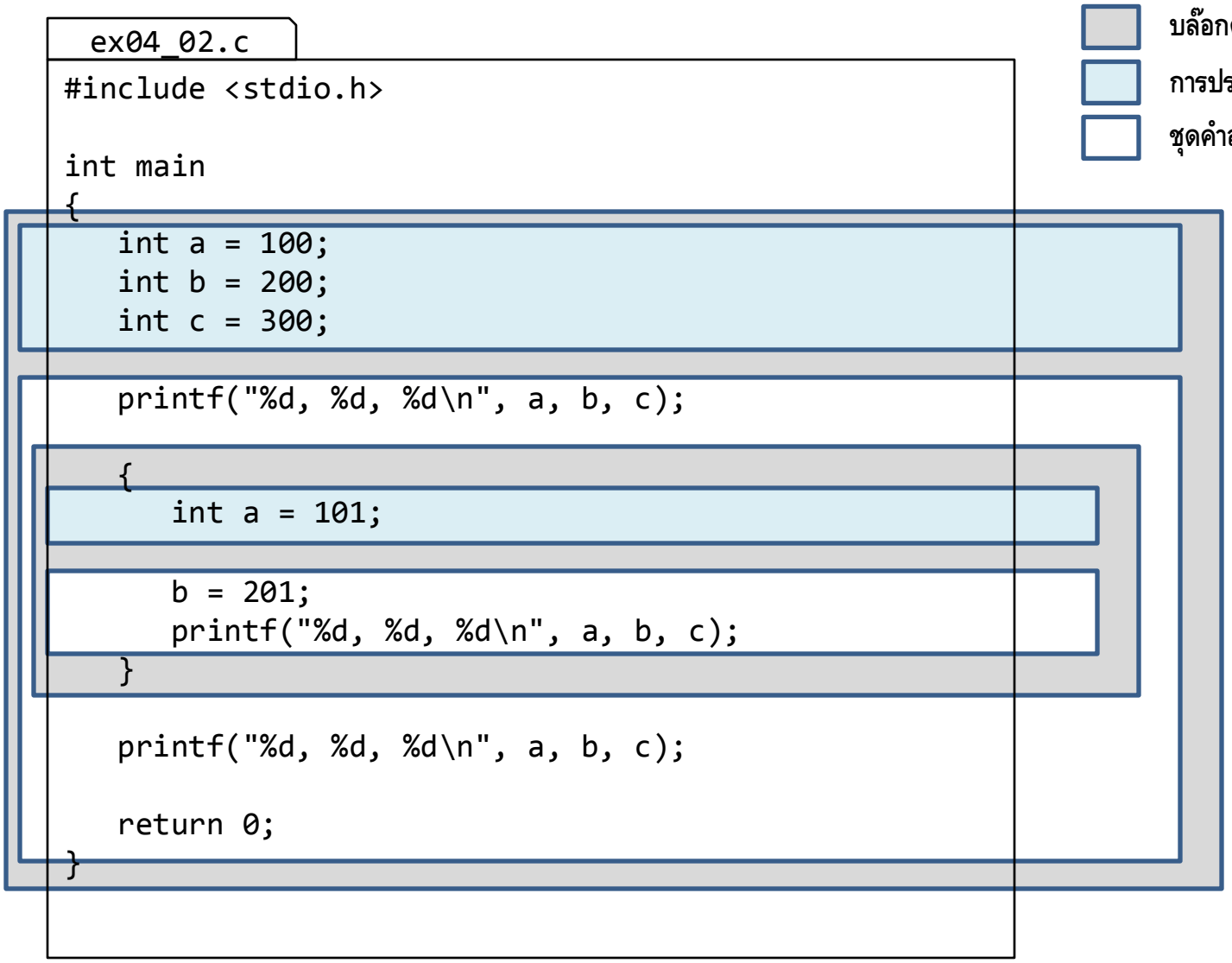
บล๊อกคำสั่ง (statement block)

การประกาศค่า (declaration)

ชุดคำสั่ง (statements)

โครงสร้างของโปรแกรม

Programming Structure



บล๊อกคำสั่ง (statement block)

การประกาศค่า (declaration)

ชุดคำสั่ง (statements)

ขอบเขตตัวแปร

Variable scope

ex04_02.c

```c
#include <stdio.h>

int main
{
   int a = 100;
   int b = 200;
   int c = 300;

   printf("%d, %d, %d\n", a, b, c);

   {
      int a;

      a = 101;
      b = 201;
      printf("%d, %d, %d\n", a, b, c);
   }

   printf("%d, %d, %d\n", a, b, c);

   return 0;
}
```

มองเห็นได้จากทั้งสองบล๊อก

มองเห็นได้เฉพาะจากบล๊อกใน บังตัวแปร a ที่อยู่บล๊อกนอก

การแก้ไขไม่กระทบบล๊อกนอก

ขอบเขตตัวแปร

Variable scope

ex04_02.c

```
#include <stdio.h>

int main
{
   int a = 100;
   int b = 200;
   int c = 300;

   printf("%d, %d, %d\n", a, b, c);

   {
      int a;

      a = 101;
      b = 201;
      printf("%d, %d, %d\n", a, b, c);
   }

   printf("%d, %d, %d\n", a, b, c);

   return 0;
}
```

ex04_02.c

```
100, 200, 300
101, 201, 300
100, 201, 300
```

ลำดับการทำงาน

Program Flow

- ทำงานทีละบรรทัด
- เมื่อทำเสร็จบรรทัดที่ **n** จะไปทำต่อบรรทัดที่ **n+1**
- เราสามารถควบคุมบรรทัดต่อไปที่จะให้โปรแกรมทำงานได้

ลำดับการทำงาน

Program Flow

ex04_01.c

```
...

int main
{
   float r;
   float area;

   scanf("%f", &r);
   area = PI * r * r;
   printf("Area : %f", area);

   return 0;
}

...
```

ลำดับการทำงาน

Program Flow

ex04_01.c

```
...

int main
{
   float r;
   float area;

   scanf("%f", &r);
   area = PI * r * r;
   printf("Area : %f", area);

   return 0;
}

...
```

ลำดับการทำงาน

Program Flow

ex04_01.c

```
...

int main
{
   float r;
   float area;

   scanf("%f", &r);
   area = PI * r * r;
   printf("Area : %f", area);

   return 0;
}

...
```

การควบคุมการทำงานของโปรแกรม

Program Flow

- การเลือกทำ
  - เงื่อนไขแบบ if – else
  - เงื่อนไขแบบ switch – case
- การทำซ้ำ หรือการวนซ้ำ
  - การวนซ้ำแบบ for
  - การวนซ้ำแบบ while
  - การวนซ้ำแบบ do-while

การควบคุมการทำงานของโปรแกรม

Program Flow

- การสั่งให้ไปทำต่อที่บรรทัดใดๆ ทันที
  - ทำให้โปรแกรมอ่านและเข้าใจได้ยาก
  - ผู้ใช้ต้องมีความรู้เรื่องสถาปัตยกรรมของคอมพิวเตอร์อย่างลึกซึ้ง
  - แทบไม่พบเห็นการใช้งานในภาษาระดับสูง
  - ภาษา C/C++ สนับสนุนวิธีนี้ด้วยคำสั่ง goto แต่ไม่รณรงค์
  - เพราะทุกอย่างที่ goto ทำได้ สามารถทำได้ด้วยวิธีสวยงามกว่า ที่จะอธิบายต่อไปนี้

เงื่อนไขแบบ if-else

Conditional branch by using "if-else"

นิพจน์ตรรกะ

Boolean Expression

- เป็นนิพจน์ที่ให้ค่าเป็นตรรกะ ซึ่งมีค่าคือจริง (true) หรือเท็จ (false)
- ในภาษาซีไม่มีตัวแปรประเภทตรรกะ (boolean)
- ใช้ตัวแปรประเภท int เก็บแทน โดยตีความหมายว่า
  - ศูนย์ (zero) เท็จ (false)
  - ไม่เท่ากับศูนย์ (non zero) จริง (true)
    โดยส่วนใหญ่มักจะพบเห็นการใช้ 1 แทนค่าตรรกะจริง

นิพจน์ตรรกะ

Boolean Expression

- **ตัวดำเนินการทางตรรกะ (boolean operator)**
  - == เท่ากัน
  - != ไม่เท่ากัน
  - || หรือ
  - && และ
  - ! นิเสธ

การเปรียบเทียบทางคณิตศาสตร์

Arithmetic Comparison

- เป็นการเปรียบเทียบจำนวนสองจำนวน
- ได้ผลลัพธ์เป็นตรรกกะ ซึ่งมีค่าคือจริง (true) หรือเท็จ (false)

การเปรียบเทียบทางคณิตศาสตร์

Arithmetic Comparison

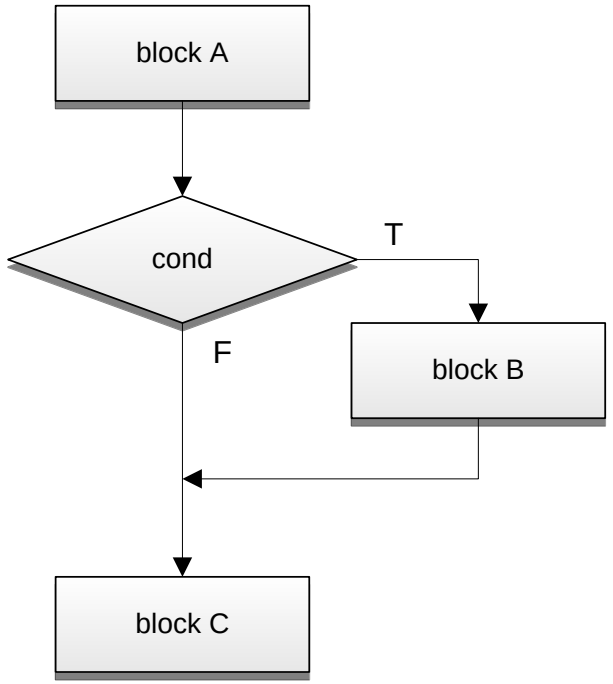
- **ตัวปฏิบัติการเปรียบเทียบทางคณิตศาสตร์ (arithmetic comparison operator)**
  - == เท่ากัน
  - != ไม่เท่ากัน
  - > มากกว่า
  - >= มากกว่าหรือเท่ากับ
  - < น้อยกว่า
  - <= น้อยกว่าหรือเท่ากับ

การใช้เงื่อนไขแบบ if

The if statement

???

```
block A

if (cond)
{
   block B
}

block C
```

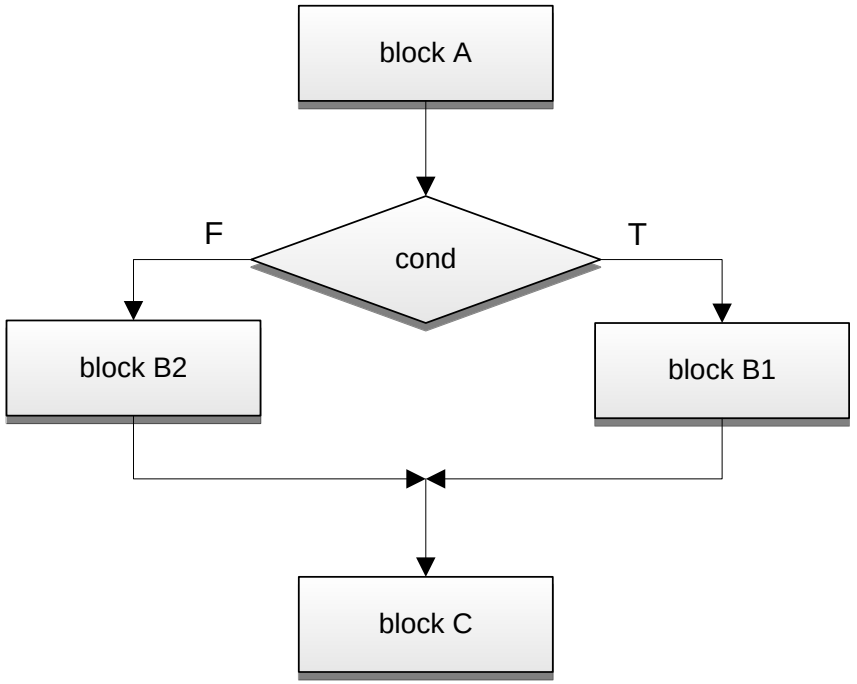


การใช้เงื่อนไขแบบ if – else

The if-else statement

???

```
block A

if (cond)
{
   block B1
}
else
{
   block B2
}

block C
```



ตัวอย่าง

An example of an if - else statements

ex04_03.c

```
#include <stdio.h>

void main()
{
   int n, r;

   printf("Enter a number : ");
   scanf("%d", &n);
   r = n % 3;

   if (r == 0)
   {
      printf("3 divides %d\n", n);
   }
   else
   {
      printf("3 does not divide %d\n", n);
      printf("The remainder is %d\n", r);
   }
}
```

ถ้าในบล๊อกมีคำสั่งเดียว ไม่ต้องมีก้ามปูก็ได้

ตัวอย่าง

An example of an if - else statements

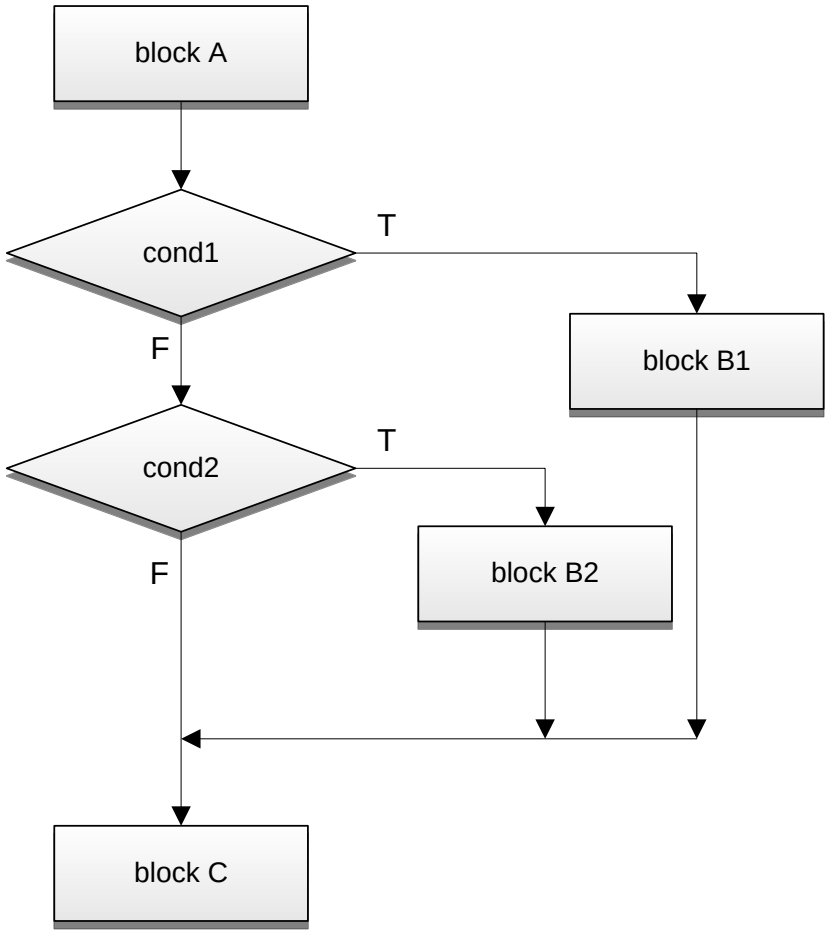
ex04_04.c

```
#include <stdio.h>

void main()
{
   int n, r;

   printf("Enter a number : ");
   scanf("%d", &n);
   r = n % 3;

   if (r == 0)
      printf("3 divides %d\n", n);
   else
   {
      printf("3 does not divide %d\n", n);
      printf("The remainder is %d\n", r);
   }
}
```

การใช้เงื่อนไขแบบ if – else if

The if - else if statement

```
???
block A

if (cond1)
{
   block B1
}
else if (cond2)
{
   block B2
}

block C
```

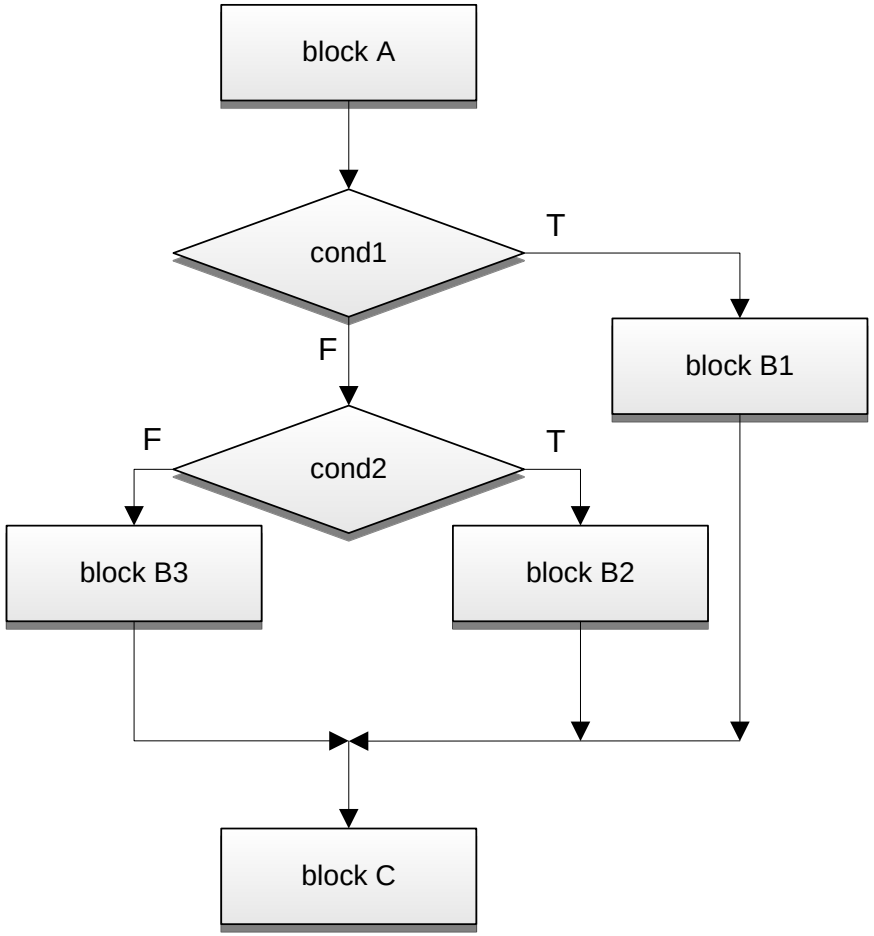


การใช้เงื่อนไขแบบ if – else if - else

The if - else if - else statement

???

```
block A

if (cond1)
{
   block B1
}
else if (cond2)
{
   block B2
}
else
{
   block B3
}

block C
```



ตัวอย่าง

An example of if - else if - else statements

ex04_05.c

```
#include <stdio.h>

void main()
{
   int score;
   char grade;

   printf("Enter the score : ");
   scanf("%d", &score);

   if (score >= 80)
      grade = 'A';
   else if (score >= 70)
      grade = 'B';
   else if (score >= 60)
      grade = 'C';
   else if (score >= 50)
      grade = 'D';
   else
      grade = 'F';

   printf("You got %c\n", grade);
}
```

การวนซ้ำด้วย for

Loop by using "for"

การวนซ้ำ

Loop

- ใช้ในการควบคุมบล๊อคให้ทำงานซ้ำหลายๆ ครั้ง
- ลักษณะคำสั่ง

```
for (init; cond; incr)
{
    statements
}
```

การวนซ้ำ

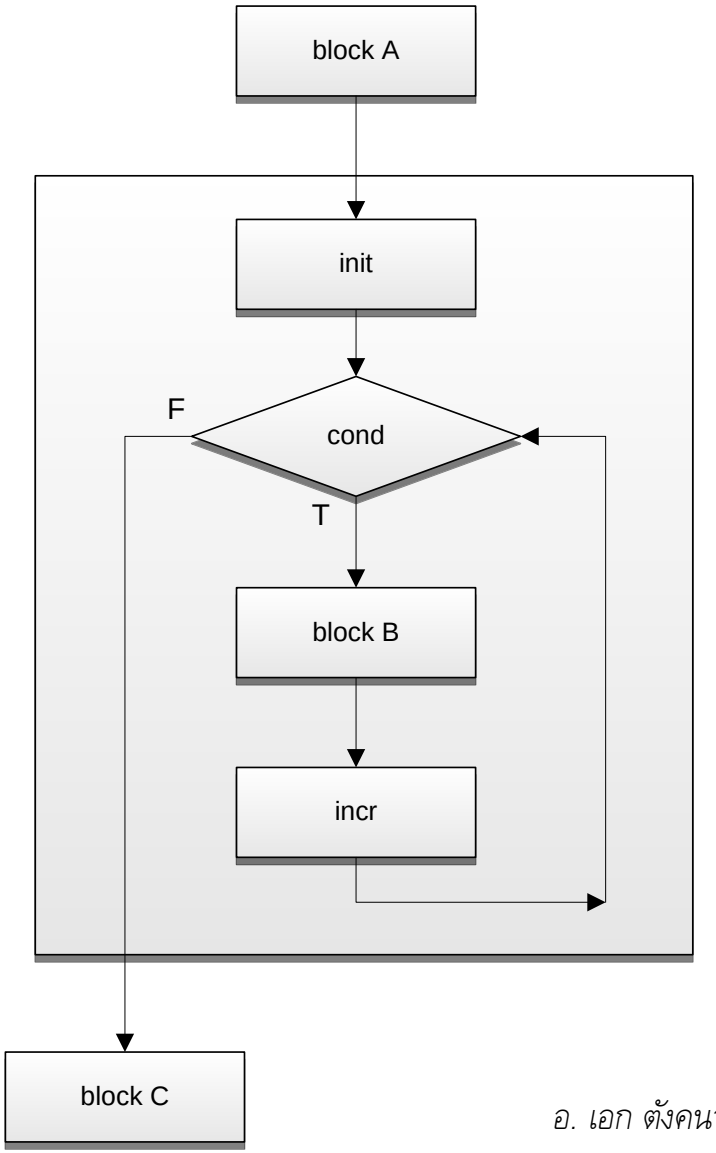
Loop

- for (*init*; *cond*; *incr*) { *statements* }
  - **init** คำสั่งที่ทำก่อนเริ่มวนซ้ำ
  - **cond** นิพจน์ตรรกะ (boolean expression) ซึ่งจะวนซ้ำในรอบถัดไปถ้าเป็นจริง (true)
  - **incr** คำสั่งที่ทำหลังการวนซ้ำในแต่ระรอบ
  - **statements** ชุดคำสั่งที่จะทำการวนซ้ำ

การวนซ้ำด้วย for

The for statement

```
???
block A

for (init; cond; incr)
{
   block B
}

block C
```



ตัวอย่าง

An example of for statements

ex04_06.c

```
#include <stdio.h>

void main()
{
   int i, n;

   printf("Enter a number : ");
   scanf("%d", &n);

   printf("Factors of %d are :\n", n);
   for (i = 1; i <= n; i++)
   {
      if (n % i == 0)
         printf("%d\n", i);
   }
}
```

ตัวควบคุมพิเศษ

Special control keywords

- เป็นคำสั่งที่ใช้ในการควบคุมลำดับการทำงานของโปรแกรม
- คำสั่งควบคุม
  - **break** ทำการออกจากการวนซ้ำรอบในสุด
  - **continue** ทำชุดคำสั่ง **incr** และวนซ้ำในรอบต่อไปทันที
  - **return** ออกจากฟังก์ชั่นทันที

ตัวอย่าง

An example of using keywords

ex04_07.c

```
#include <stdio.h>

void main()
{
   int i, n;

   printf("Enter a number : ");
   scanf("%d", &n);

   for (i = 2; i < n; i++)
   {
      if (n % i == 0)
         break;
   }

   if (n == 2 || i == n)
      printf("%d is prime\n", n);
   else
      printf("%d is not prime\n", n);
}
```

การวนซ้ำด้วย while, do-while

Loop by using "while", "do-while"

การวนซ้ำด้วย while

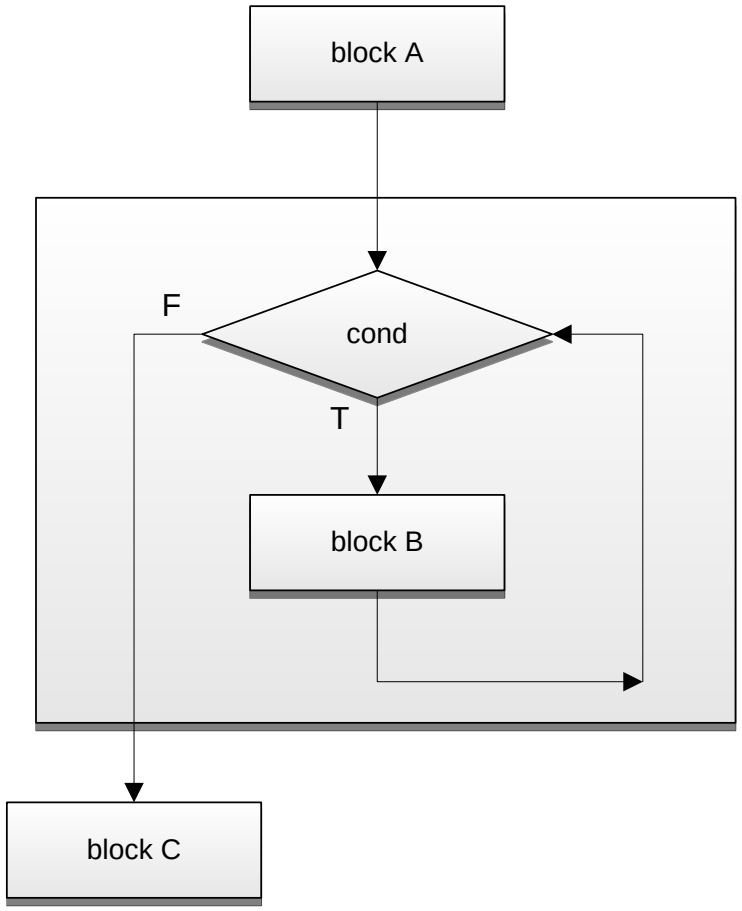
Loop by using "while"

- **while (*cond*) { *statements* }**
  - **cond** นิพจน์ตรรกะ (boolean expression) ซึ่งจะทำการวนซ้ำในรอบนี้ถ้าเป็นจริง (true)
  - **statements** ชุดคำสั่งที่จะทำการวนซ้ำ

การวนซ้ำด้วย while

The while statement

```
???
block A

while (cond)
{
   block B
}

block C
```



การวนซ้ำด้วย do-while

Loop by using "do-while"

- do { *statements* } while (cond);
  - **cond** นิพจน์ตรรกะ (boolean expression) ซึ่งจะทำการวนซ้ำในรอบถัดไปถ้าเป็นจริง (true)
  - **statements** ชุดคำสั่งที่จะทำการวนซ้ำ

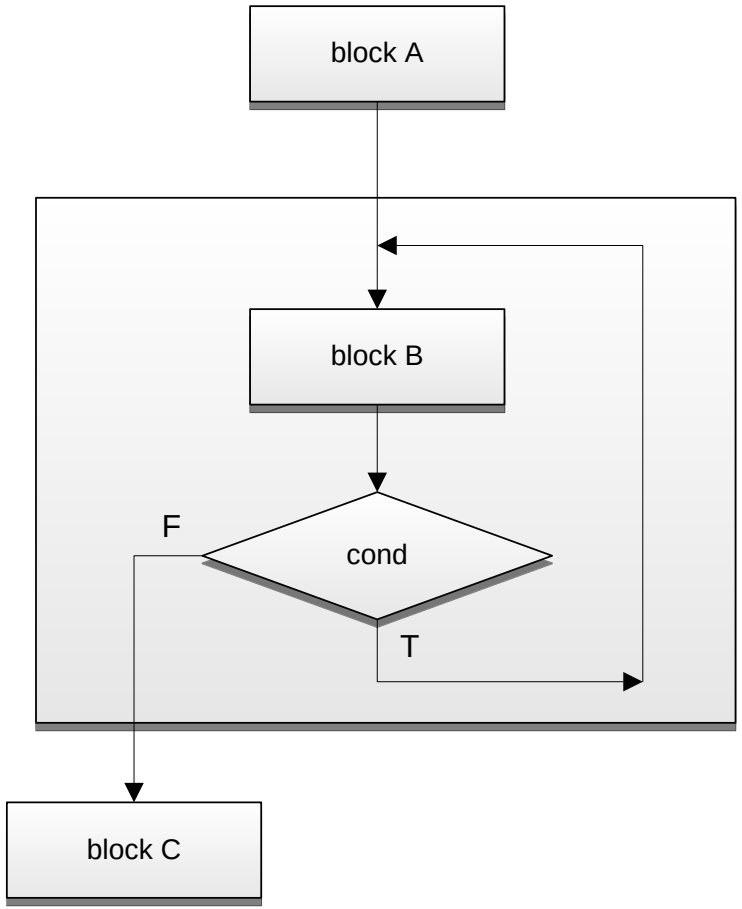
อ. เอก ตังคนานนท์

การวนซ้ำด้วย do-while

The do-while statement

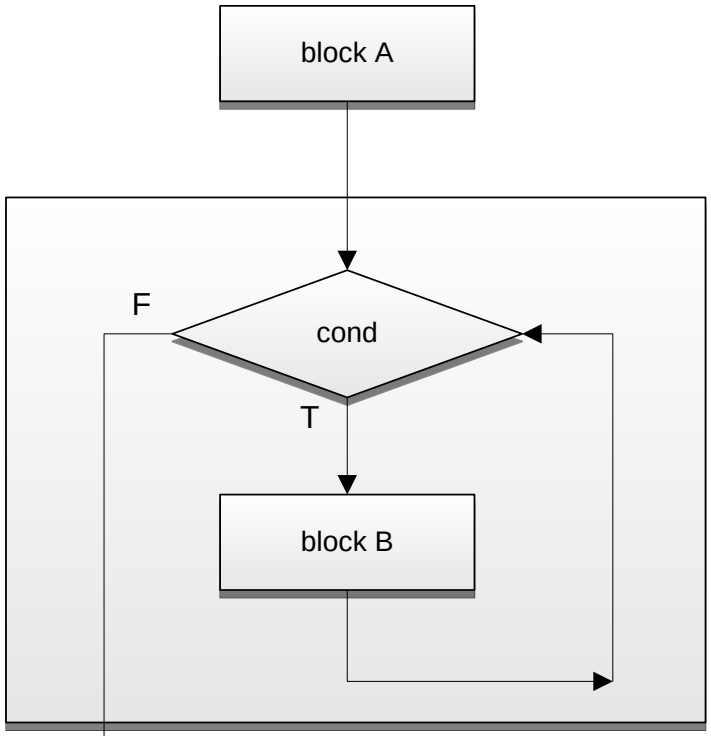
???

```
block A

do
{
   block B
}
while (cond);

block C
```

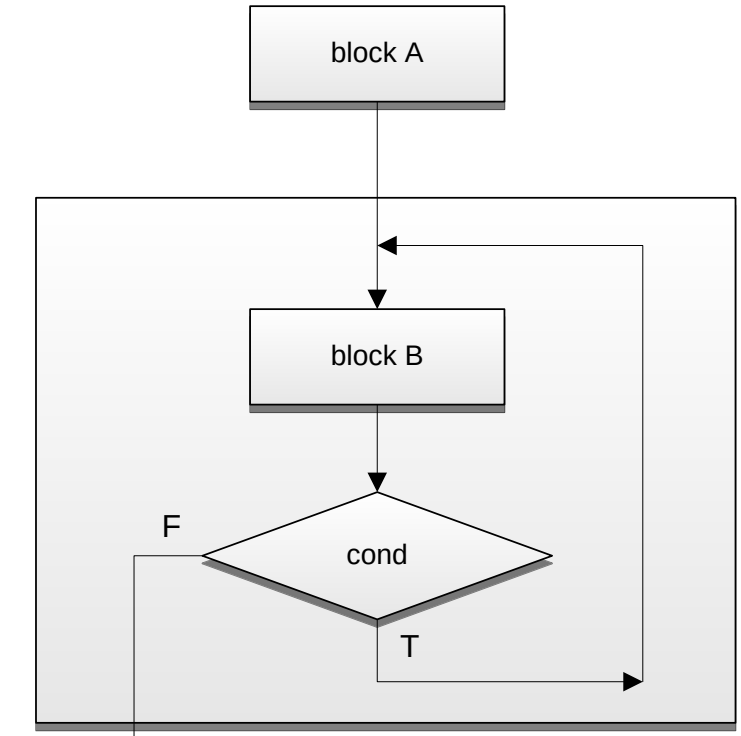


ข้อแตกต่าง

Differences




```
block A

while (cond)
{
   block B
}

block C
```




```
block A

do
{
   block B
}
while (cond);

block C
```

การเขียนโปรแกรมวนซ้ำ

Loop programming

- มันคือศิลปะแขนงหนึ่ง
- สามารถเขียนได้หลายวิธีที่ได้ผลลัพธ์ในการทำงานเหมือนกัน
- ควรเลือกวิธีที่เหมาะสมกับลักษณะงาน สวยงาม อ่านง่าย และไม่ขัดกับความรู้สึก
- ไม่มีวิธีไหนที่ดีที่สุดเสมอไป และการวนซ้ำก็อาจจะไม่ใช่วิธีที่ดีที่สุด

ตัวอย่าง

Example programs

- โปรแกรมหาผลรวมของจำนวนเต็มตั้งแต่ 1 ถึง n
  - รับข้อมูลมาหนึ่งตัว คือ n
  - กรณีที่ n < 1 ควรตอบ 0
- โปรแกรมหาค่าเฉลี่ยทั้งหมดของข้อมูลที่พิมพ์เข้ามา
  - โปรแกรมรับข้อมูลเข้ามาเรื่อยๆ
  - หยุดเมื่อข้อมูลที่รับมาเท่ากับ 0 โดยไม่นำไปรวมในกลุ่มที่จะหาค่าเฉลี่ย
  - ควรพิจารณากรณีที่จำนวนข้อมูลเป็น 0 ด้วย แต่ขอละไว้ก่อน

โปรแกรมหาผลรวมตั้งแต่ 1 ถึง n

The summation of the integers from 1 to n

ex04_08.c

```
#include <stdio.h>

void main()
{
   int i, n, sum;

   printf("Enter a number :");
   scanf("%d", &n);

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


   printf("Sum 1 to %d is %d\n",
      n, sum);
}
```

เปรียบเทียบ while กับ do-while

A comparison between while loop and do-while loop

ex04_09.c

สังเกตุกรณี n < 1

```
#include <stdio.h>

void main()
{
   int i, n, sum;

   printf("Enter a number :");
   scanf("%d", &n);

   i = 1;
   sum = 0;
   do
   {
      sum += i;
      i++;
   }
   while (i <= n);

   printf("Sum 1 to %d is %d\n",
      n, sum);
}
```

ex04_08.c

```
#include <stdio.h>

void main()
{
   int i, n, sum;

   printf("Enter a number :");
   scanf("%d", &n);

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


   printf("Sum 1 to %d is %d\n",
      n, sum);
}
```

เปรียบเทียบ while กับ for

A comparison between while loop and for loop

ex04_09.c

```
#include <stdio.h>

void main()
{
   int i, n, sum;

   printf("Enter a number :");
   scanf("%d", &n);

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


   printf("Sum 1 to %d is %d\n",
      n, sum);
}
```

ex04_10.c

ดูอ่านง่ายกว่ามั้ย

```
#include <stdio.h>

void main()
{
   int i, n, sum;

   printf("Enter a number :");
   scanf("%d", &n);

   sum = 0;
   for (i = 1; 1 <= n; i++)
      sum += i;

   printf("Sum 1 to %d is %d\n",
      n, sum);
}
```

เปรียบเทียบ for กับใช้สูตร

A comparison between using for loop and using a formula

ex04_10.c

ลอง n เยอะๆ คง ทำนานน่าดู

```
#include <stdio.h>

void main()
{
   int i, n, sum;

   printf("Enter a number :");
   scanf("%d", &n);

   sum = 0;
   for (i = 1; 1 <= n; i++)
      sum += i;

   printf("Sum 1 to %d is %d\n",
      n, sum);
}
```

ex04_11.c

```
#include <stdio.h>

void main()
{
   int i, n, sum;

   printf("Enter a number :");
   scanf("%d", &n);

   if (n > 1)
      sum = n * (n + 1) / 2;
   else
      sum = 0;

   printf("Sum 1 to %d is %d\n",
      n, sum);
}
```

โปรแกรมหาค่าเฉลี่ย

The average of input values

ex04_12.c

```
#include <stdio.h>

void main()
{
   int a, sum = 0, n = 0;

   printf("Please enter data.\n");
   printf("Enter 0 to terminate.\n");
   do
   {
      printf("%3d: ", n + 1);
      scanf("%d", &a);

      sum += a;
      n++;
   }
   while (a != 0);


   printf("The average is %.2f\n",
      1.0f * sum / (n - 1));
}
```

เปรียบเทียบ do-while กับ while

A comparison between do-while loop and while loop

ex04_12.c

```
#include <stdio.h>

void main()
{
   int a, sum = 0, n = 0;

   printf("Please enter data.\n");
   printf("Enter 0 to terminate.\n");
   do
   {
      printf("%3d: ", n + 1);
      scanf("%d", &a);

      sum += a;
      n++;
   }
   while (a != 0);


   printf("The average is %.2f\n",
      1.0f * sum / (n - 1));
}
```

ex04_13.c

ยาวกว่าหน่อย แต่ดูเป็นธรรมชาติกว่า

```
#include <stdio.h>

void main()
{
   int a, sum = 0, n = 0;

   printf("Please enter data.\n");
   printf("Enter 0 to terminate.\n");
   while (1)
   {
      printf("%3d: ", n + 1);
      scanf("%d", &a);
      if (a == 0)
         break;

      sum += a;
      n++;
   }

   printf("The average is %.2f\n",
      1.0f * sum / n);
}
```

เปรียบเทียบ while กับ for

A comparison between while loop and for loop

ex04_13.c

```
#include <stdio.h>

void main()
{
   int a, sum = 0, n = 0;

   printf("Please enter data.\n");
   printf("Enter 0 to terminate.\n");
   while (1)
   {
      printf("%3d: ", n + 1);
      scanf("%d", &a);
      if (a == 0)
         break;

      sum += a;
      n++;
   }

   printf("The average is %.2f\n",
      1.0f * sum / n);
}
```

ex04_14.c

ทำงานถูก แต่อ่านเข้าใจยากกว่ามั้ย

```
#include <stdio.h>

void main()
{
   int a, sum = 0, n = 0;

   printf("Please enter data.\n");
   printf("Enter 0 to terminate.\n");

   for (a = 1; a != 0; n++)
   {
      printf("%3d: ", n + 1);
      scanf("%d", &a);
      sum += a;
   }

   printf("The average is %.2f\n",
      1.0f * sum / (n - 1));
}
```

การเลือกทำด้วย switch-case

Branch by using "switch-case"

การเลือกทำด้วย switch-case

Branch by using "switch-case"

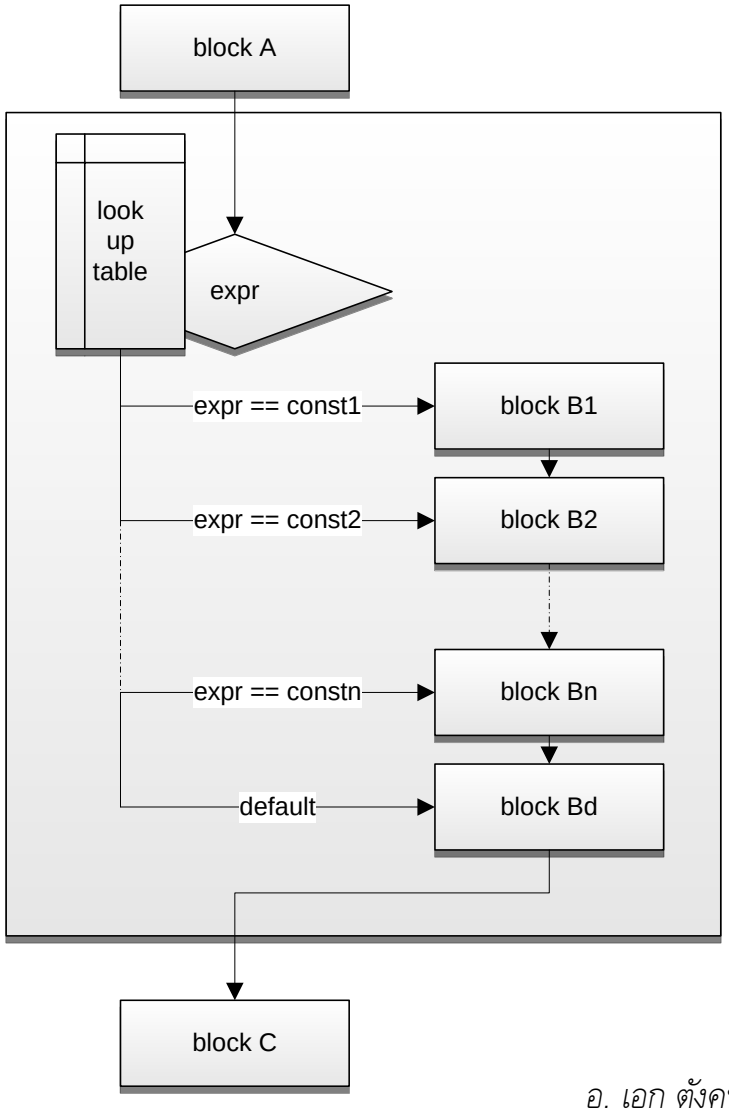
- ลักษณะคล้ายๆ กับการใช้ if และ else if ซ้อนกันหลายๆ อัน
- มีประสิทธิภาพมากกว่า
- ใช้ได้เฉพาะกับการเปรียบเทียบการเท่ากันกับค่าคงที่ประเภทจำนวนเต็ม (integer constant) เท่านั้น

การเลือกทำด้วย switch-case

Branch by using "switch-case"

???

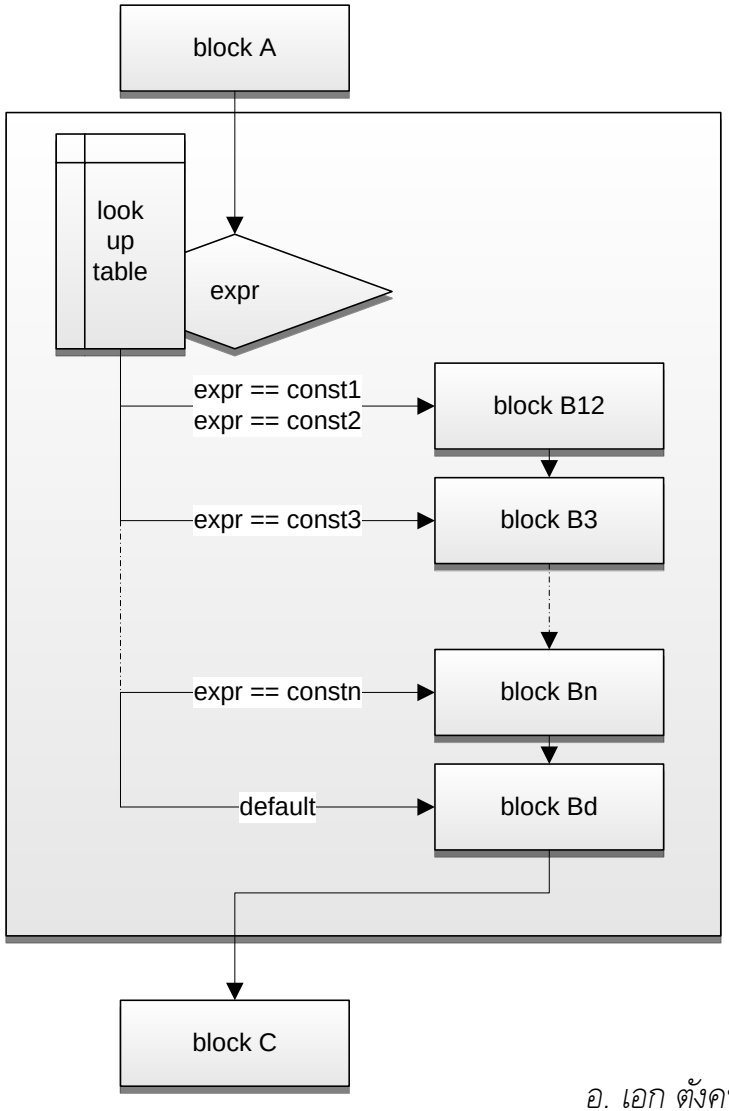
```
block A

switch (expr)
{
case const1:
   block B1;
case const2:
   block B2;
...
case constn:
   block Bn;
default:
   block Bd;
}

block C
```



การเลือกทำด้วย switch-case

Branch by using "switch-case"

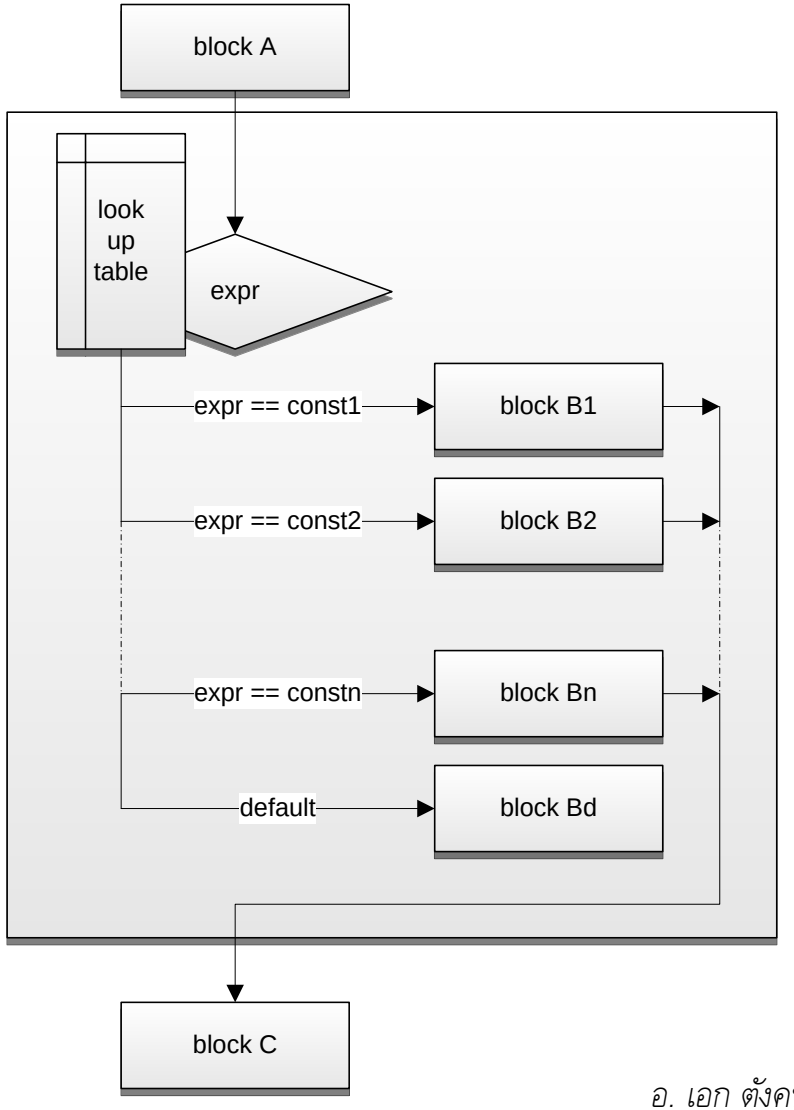
```
???
block A

switch (expr)
{
case const1:
case const2:
   block B12;
case const3:
   block B3;
...
case constn:
   block Bn;
default:
   block Bd;
}

block C
```



การเลือกทำด้วย switch-case

Branch by using "switch-case"

```
???
block A

switch (expr)
{
case const1:
   block B1;
   break;
case const2:
   block B2;
   break;
...
case constn:
   block Bn;
   break;
default:
   block Bd;
}

block C
```



ตัวควบคุมพิเศษ

Special control keywords

- คล้ายๆ กับในเรื่องของการวนซ้ำ (loop)
- ตัวควบคุม
  - **break** ทำการออกจากประโยค switch-case ทันที
  - **default** ใช้แทนเงื่อนไขอื่นๆ นอกเหนือจากตัวเลือก (case) ที่กำหนดไว้

เปรียบเทียบ switch-case กับ if-else

Branch by using "switch-case"

ex04_15.c

```
#include <stdio.h>

void main()
{
   int a, b, ans, sel;

   printf("a = "); scanf("%d", &a);
   printf("b = "); scanf("%d", &b);
   printf("1 a + b\n");
   printf("2 a - b\n");
   printf("3 a * b\n");
   printf("4 a / b\n");

   printf("Select : ");
   scanf("%d", &sel);

   switch (sel)
   {
   case 1:
      ans = a + b;
      break;
```

page 1/2

```
   case 2:
      ans = a - b;
      break;
   case 3:
      ans = a * b;
      break;
   case 4:
      ans = a / b;
      break;
   default:
      printf("Invalid selection\n");
      ans = -1;
   }

   printf("Ans : %d\n", ans);
}
```

page 2/2

เปรียบเทียบ switch-case กับ if-else

Branch by using "switch-case"

ex04_16.c

```
#include <stdio.h>

void main()
{
   int a, b, ans, sel;

   printf("a = "); scanf("%d", &a);
   printf("b = "); scanf("%d", &b);
   printf("1 a + b\n");
   printf("2 a - b\n");
   printf("3 a * b\n");
   printf("4 a / b\n");

   printf("Select : ");
   scanf("%d", &sel);

   if (sel == 1)
   {
      ans = a + b;
   }
   else if (sel == 2)
   {
      ans = a – b;
   }
   else if (sel == 3)
   {
      ans = a * b;
   }
   else if (sel == 4)
   {
      ans = a / b;
   }
   else
   {
      printf("Invalid selection\n");
      ans = -1;
   }

   printf("Ans : %d\n", ans);
}
```

page 1/2

page 2/2

จบบทที่ 4

The End of Chapter 4

บทที่ 5 ฟังก์ชั่น

ช่วยให้เขียนโปรแกรมได้สวยงามและอ่านง่าย นอกจากนี้ยังช่วยในการทำงานบางอย่างได้เป็นธรรมชาติมากยิ่งขึ้น ตัวอย่างเช่นการหาลำดับฟิโบนันซี่ที่ n ที่มีนิยามว่า f(n) = f(n-1) + f(n-2) โดยที่ f(0) = 0, f(1) = 1