ครั้งที่ 3

โครงสร้างทางเลือก :IF และ SWITCH

โครงสร้างการปฏิบัติการ

- sequence
- selection
- repetition

นิพจน์ตรรก (Logical Expressions)

variable	relational-operator	variable
variable	relational-operator	constant
variable	equality-operator	variable
variable	equality-operator	constant

relational and Equality Operators

Operator	Meaning	Type
<	less than	relational
>	greater than	relational
<=	less than or equal to	relational
>=	greater than or equal to	relational
==	equal to	Equality
!=	not equal to	Equality

ตัวอย่างเงื่อนไข

Operator	Condition	Value
<=	x <= 0	true
<	power < maxPower	false
>=	x >= y	false
>	item > minItem	true
==	momOrDad == ‘m’	true
!=	num != sentinel	false

Logical operator

Operand1	Operand2	Operand1 || Operand2
true	true	true
true	false	true
false	true	true
false	false	false

Operand1	Operand2	Operand1 && Operand2
true	true	true
true	false	false
false	true	false
false	false	false

Logical operator

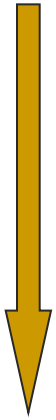
Operand	! Operand
true	false
false	true

```
if (weight > 100.00)
        ShipCost = 10.00 ;
    else
        ShipCost = 5.00 ;
```

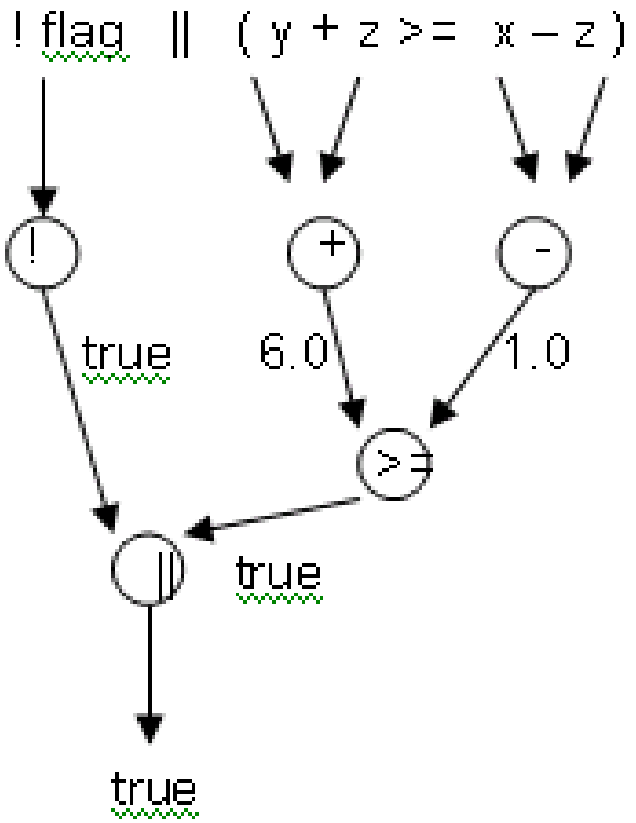
```
(salary < minSalary  ||  dependents > 5)
(temerature > 90.0  && humidity > 0.90)
```

Operator Precedence

Operator	Precedence	Description
! , + , -	Highest	Logical not,unary plus,unary minus
* , / , %		Multiplication, division , modulus
+ , -		Addition , subtraction
< , <= , >= , >		Relational inequality
== , !=		Equal ,not equal
&&		Logical and
||		Logical or
=	Lowest	Assignment



ตัวอย่าง ! flag || (y + z >= x - z)



flag	y	z	x
false	4.0	2.0	3.0

! flag || (y + z >= x – z)

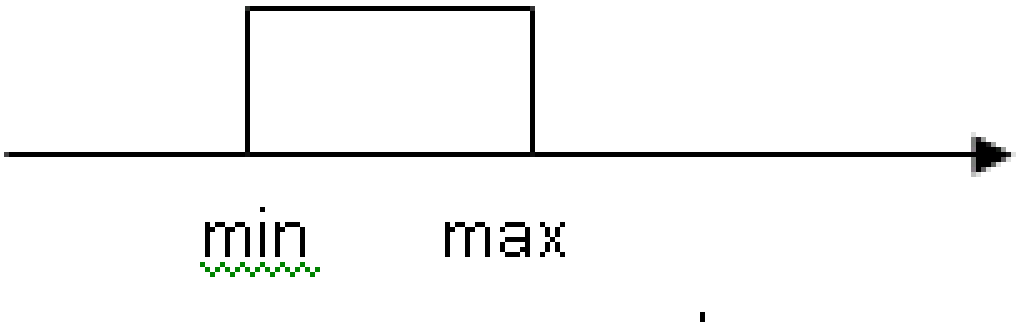
! false 4.0+2.0 3.0-2.0

true 6.0 >= 1.0

true || true

true

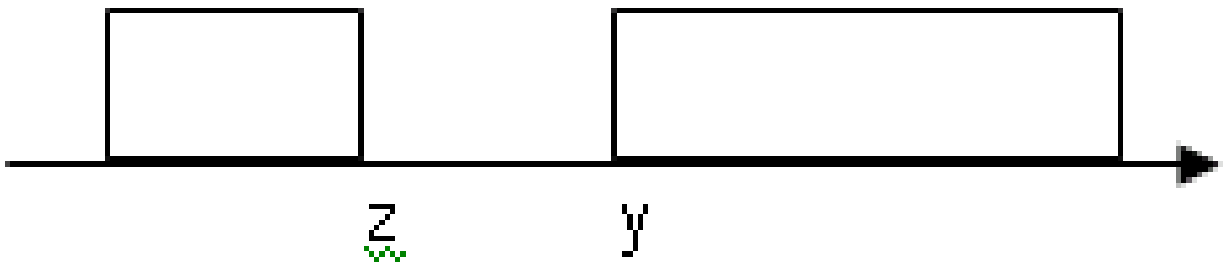
การตรวจสอบเงื่อนไขที่เป็นช่วงของข้อมูล



โดยเราสามารถสร้างนิพจน์ตรรกเพื่อใช้สำหรับตรวจสอบค่าของตัวแปร x ว่ามีค่าอยู่ในช่วงระหว่าง min และ max โดยใช้ตัวกระทำ && ดังนี้

(min <= x && x <= max)

การตรวจสอบเงื่อนไขที่เป็นช่วงของข้อมูล



ถ้าเราต้องการตรวจสอบค่าของตัวแปร X ว่าอยู่ในช่วงของข้อมูลดังรูปหรือไม่นั้น
จะใช้ตัวกระทำ && ไม่ได้เพราะข้อมูลมิได้อยู่ในช่วงเดียวกัน
แต่สามารถตรวจสอบโดยใช้ตัวกระทำ || แทน ดังนี้
(z > x || x > y) ซึ่งค่าของนิพจน์จะมีค่าเท่ากับ true
ถ้าข้อมูลเป็นจริงถ้าการเปรียบเทียบมีค่าเท่ากับ true เพียงค่าใดค่าหนึ่งเท่านั้น

นิพจน์ทางตรรกที่ผิด

```
x && y > z      // invalid logical expression
z <= x <= y
```

ตัวอย่างการเปรียบเทียบตัวอักษรและข้อความ

Expression	Value
'a' < 'c'	true
'X' <= 'A'	false
'3' > '4'	false
('A' <= ch) && (ch <= 'Z')	true if ch contains an uppercase letter;otherwise false
"XYZ" <= "ABC"	false (X <= A)
"acts" > "aces"	true (t > e)
"ace" != "aces"	true(string are different)
"ace" < "aces"	true

คำสั่ง **if**

- if statement with a dependent statement
- if statement with two alternatives
- Multiple-Alternative Decision

if Statement with Dependent Statement

if Statement with Dependent Statement

รูปแบบ:

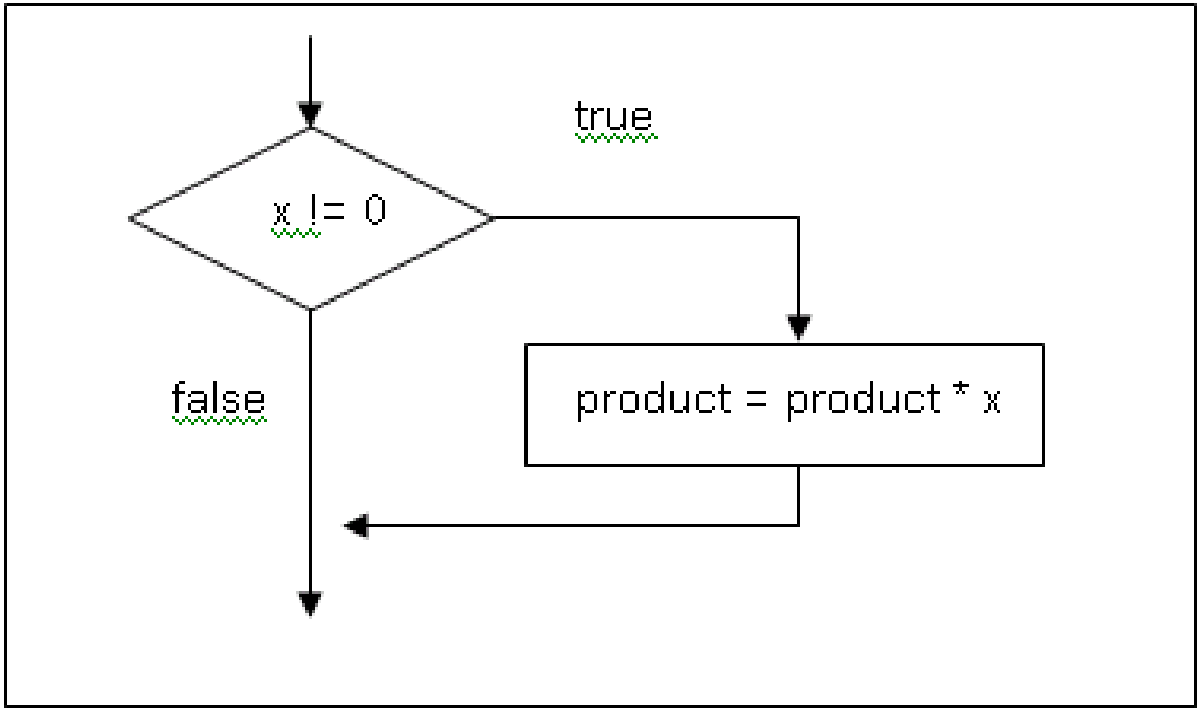
```
if (condition)
    statement ;
```

ตัวอย่าง:

```
if ( x != 0 )
    product = product * x ;
```

ความหมาย : จะกระทำ statement เมื่อ condition มีค่าเท่ากับ true ในกรณีอื่นๆจะข้ามไปทำคำสั่งต่อไป

if Statement with Dependent Statement



if Statement with Dependent Statement

ในกรณีที่ทางเลือกของการปฏิบัติงานนั้นมีการทำงานหลายๆคำสั่ง (Compound statements) ต้องเขียนคำสั่งเหล่านั้นภายใต้เครื่องหมาย { } ดังนี้

```
if   (popToday > popYesterday)
{
   growth = popToday – popYesterday ;
   growthPct = 100.0 * growth / popYesterday ;
   cout << “The growth percentage is ‘ , growthPct ;
}
```

if Statement with Two Alternatives

if Statement with Two Alternatives

รูปแบบ:

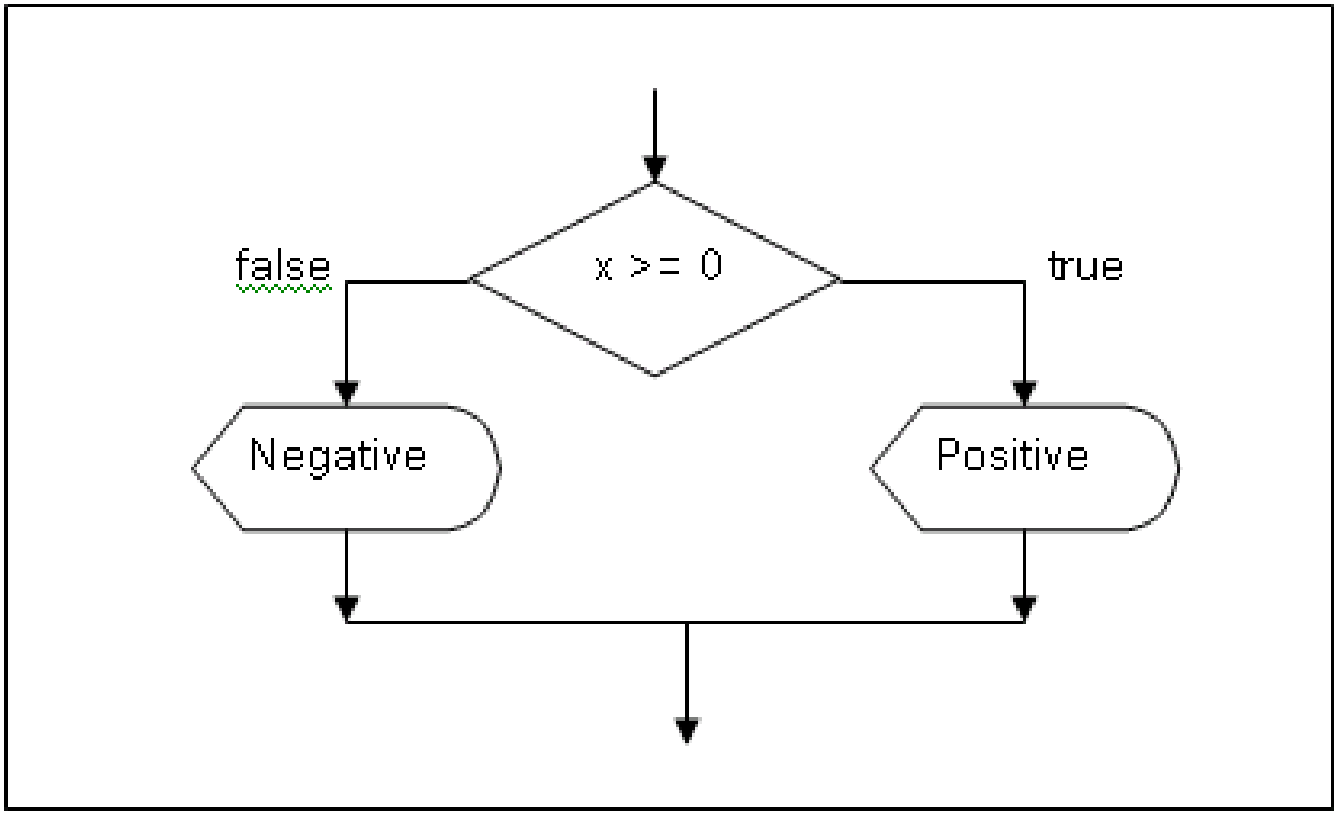
```
if (condition)
    statement_T ;
else
    statement_F ;
```

ตัวอย่าง:

```
if ( x >= 0.0 )
   cout << 'Positive" << endl ;
else
   cout <, "Negative" << endl ;
```

ความหมาย : จะกระทำ statement$_T$ เมื่อ condition มีค่าเท่ากับ true และกระทำ statement$_F$ เมื่อ condition เท่ากับ false

if Statement with Two Alternatives



if Statement with Two Alternatives

กรณีที่แต่ละทางเลือกมีการปฏิบัติงานหลายๆคำสั่ง
ต้องเขียนกลุ่มคำสั่งที่ปฏิบัติการภายใต้เครื่องหมาย { } ดังนี้

```
if  (transactionType == 'c')
{
    cout << "Check for $" << transactionAmount << endl ;
    balance = balance – transactionAmount ;
}
else
{
    cout << "Deposit of $" << transactionAmount << endl ;
    balance = balance + transactionAmount ;
}
```

กรณีศึกษา :
โปรแกรมการคิดเงินเดือนของคนงาน

ปัญหา

บริษัทแห่งหนึ่งต้องการคิดเงินเดือนให้กับคนงานในแต่ละสัปดาห์ โดยคนงานแต่ละคนมีอัตราค่าจ้างต่อชั่วโมง และจำนวนชั่วโมงทำงานแตกต่างกัน ถ้าคนงานคนใดที่ทำงานเกินสัปดาห์ละ 40 ชั่วโมงจำนวนชั่วโมงที่เกินบริษัทจะให้อัตราค่าจ้างเพิ่มจากเดิมเป็น 1.5 เท่าของอัตราปกติที่ได้รับ นอกจากนี้ถ้าคนงานคนใดที่มีรายได้ต่อสัปดาห์ตั้งแต่ 2500 บาทขึ้นไปจะต้องหักค่าภาษี ณ ที่จ่ายคนละ 50 บาท

โปรแกรมการคิดเงินเดือนของคนงาน

วิเคราะห์

ในที่นี้ข้อมูลเข้าคือ

- อัตราค่าจ้างต่อชั่วโมง(rate)
- จำนวนชั่วโมงที่คนงานทำงานต่อสัปดาห์(hours)

ผลลัพธ์ที่ต้องการคือ

- จำนวนเงินที่ได้รับในแต่ละสัปดาห์(gross)
- จำนวนเงินที่ได้รับจริง(net)หลังจากหักภาษีแล้ว

โปรแกรมการคิดเงินเดือนของคนงาน

DATA REQUIREMENTS

Problem Constant

MAX_NO_DUES = 2500.00 // maximum earning without paying union dues
DUES = 50.00 // union dues to be paid
MAX_NO_OVERTIME = 40.0 // maximum hours without overtime pay
OVERTIME_RATE = 1.5 // time and a half for overtime

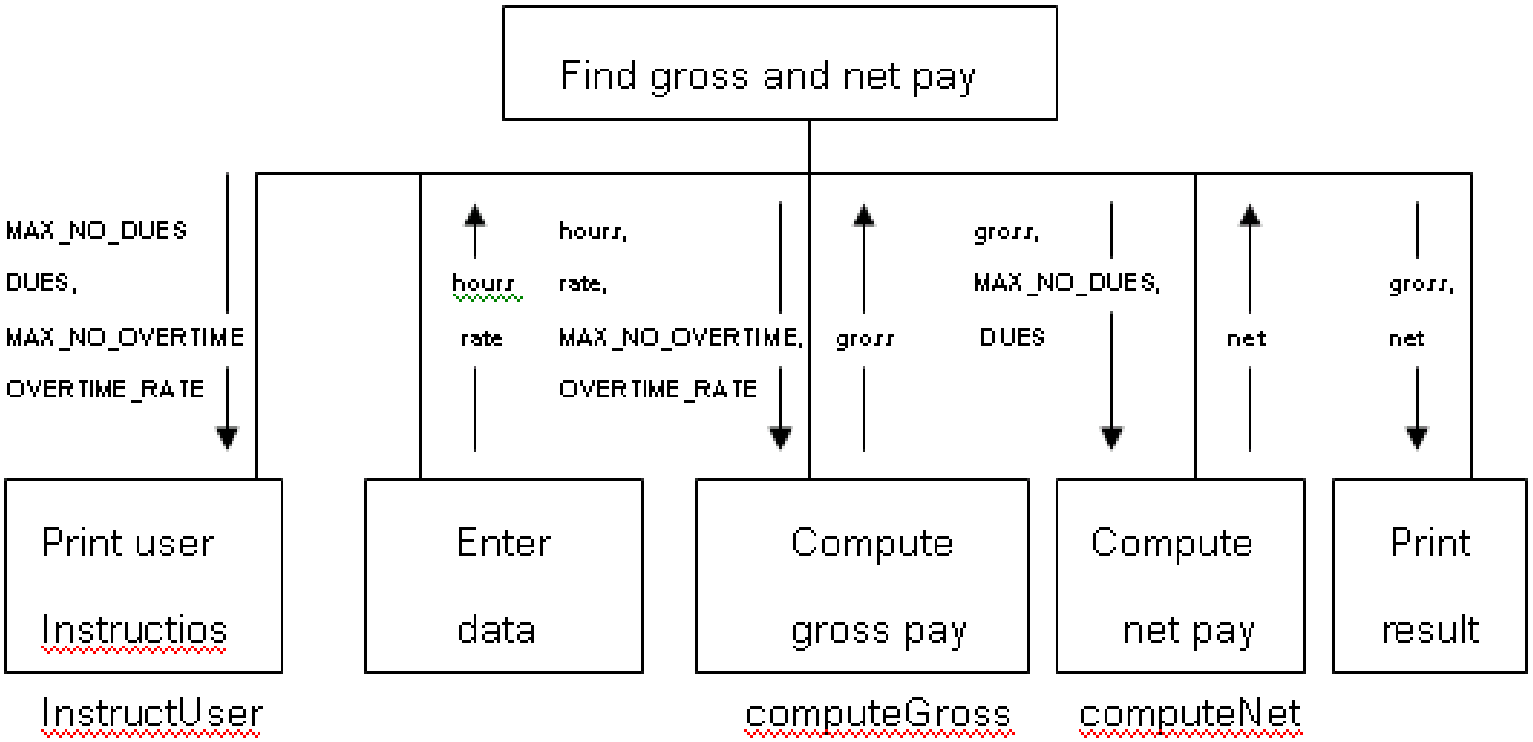
Problem Input

float hours // hours work
float rate // hourly rate

Problem Output

float gross // gross pay
float net // net pay

โปรแกรมการคิดเงินเดือนของคนงาน



โปรแกรมการคิดเงินเดือนของคนงาน

INITIAL ALGORITHM

1. แสดงข้อความเพื่อแนะนำผู้ใช้งาน (ฟังก์ชัน instructUser)
2. ป้อนชั่วโมงทำงานและอัตราค่าจ้างต่อชั่วโมง
3. คำนวณหารายได้ (ฟังก์ชัน computeGross)
4. คำนวณหารายได้สุทธิ (ฟังก์ชัน computeNet)
5. แสดงรายได้ และรายได้สุทธิออกทางจอภาพ

โปรแกรมการคิดเงินเดือนของคนงาน

```
#include <iostream>
using namespace std;
// Function used
void instructUser () ;
float computeGross (float , float) ;
float computeNet (float) ;
const float  MAX_NO_DUES = 2500.00 ;       // max earning before dues
const float DUES = 50.00 ;                 // dues amount
const float MAX_NO_OVERTIME = 40.00 ;      // max hours before overtime
const float OVERTIME_RATE = 1.5 ;                  // overtime rate
```

```
int main ( )
{
      float  hours ;   // input : hours worked
      float rate ;                  // input : hourly pay rate
      float gross ;    // output : gross pay
      float net ;                   // output : net pay
      // Display user instructions
      instructUser ( ) ;
      // Enter hours and rate
     cout << “Hours worked : “ ;
     cin >> hours ;
     cout << “ Hourly rate : “ ;
     cin >> rate ;
     // Compute gross salary
     gross = computeGross (hours , rate);
     // Compute net salary
     net = computeNet (gross) ;
     // Print gross and net
     cout << “Gross salary is “ << gross << endl ;
     cout << “ Net salary is “ <, net << endl ;
     return 0;
}
```

โปรแกรมการคิดเงินเดือนของคนงาน

```
// Displays user instructions
void instructUser ( )
{
    cout << " This program computes gross and net salary. " << endl ;
    cout << " A dues amount of " << DUES << " is deducted for " <, endl;
    cout << "an emplayee who raens more than " << MAX_NO_DUES << endl <<endl;
    cout << " Overtime is paid at the rate of " << OVERTIME_RATE << endl;
    cout << "times the regular rate for hours worked over "
         << MAX_NO_OVERTIME << endl << endl ;
    cout << "Enter hours worked and hourly rate " << endl ;
    cout << "pn separate lines after the prompts. " << endl ;
    cout << "Press <return> after typing each number. " << endl << endl;
}   // end instructUser
```

โปรแกรมการคิดเงินเดือนของคนงาน

```
// Find the gross pay
float computeGross ( float hours ,     // IN : number of hours worked
                     float rate)               // IN : hourly pay rate
{
    // Local data
    float gross ;              // RESULT : gross pay
    float regularPay ;         // pay for first 40 hours
    float overtimePay ;        // pay for hours in exvess of 40
    // Compute gross pay.
    if (hours > MAX_NO_OVERTIME)
    {
            regularPay = MAX_NO_OVERTIME * rate ;
        overtimePay = (hours – MAX_NO_OVERTIME) * OVERTIME_RATE * rate ;
            gross = regularPay + overtimePay ;
    }
    else
            gross = hours * rate ;
    return gross ;
} // end computeGross
```

โปรแกรมการคิดเงินเดือนของคนงาน

```
// Find the net pay
float computeNet (float gross)                    // IN: gross salary
{
    // Local data
    float net ;                                   // RESULT : net pay
    // Compute net pay
    if (gross > MAX_NO_DUES)
            net = gross – DUES ;                  // deduct dues amount
    else
            net = gross ;                         // no deductions
    return net
}
// end computeNet
```

โปรแกรมการคิดเงินเดือนของคนงาน

การทดสอบ โดยป้อนข้อมูลที่ทำให้โปรแกรมถูกทดสอบทุกๆทางเลือก

ในที่นี้ทดลองป้อนจำนวนชั่วโมงทำงานเท่ากับ 50

และอัตราค่าจ้างต่อชั่วโมงเท่ากับ 150 บาท

```
This program computes gross and net salary.
A dues amount of 50.00 is deducted for
an employee who earns more than 2500.00
Overtime is paid at the rate of 1.5
times the regular rate on hours worked over 40
Enter hours worked and hourly rate
on separate lines after the prompts.
Press <return> after typing each number.
Hours worked :   50
Hourly rate :    150
Gross salary is  9000.00
Net salary is   8950.00
```

Multiple-Alternative Decisions

Multiple-Alternative Decision Form

รูปแบบ:

```
if (condition 1)
    statement 1 ;
else if (condition 2)
    statement 2 ;
.....
else if (condition n)
    statement n ;
```

ตัวอย่างฟังก์ชันในการแสดงเกรด

คะแนนสอบ	เกรดที่ได้รับ
90 และมากกว่า	A
80 ถึง 89	B
70 ถึง 79	C
60 ถึง 69	D
ต่ำกว่า 60	F

```
// Displays  the letter grade corresponding to
// an exam
// score assuming a normal scale.
void displayGrade (int score)
{
        if (score >= 90 )
            cout  << “Grade  is  A “ << endl ;
        else if (score >= 80 )
             cout  << “Grade  is  B “ << endl ;
        else if (score >= 70 )
             cout  << “Grade  is  C “ << endl ;
        else if (score >= 60 )
             cout  << “Grade  is  D “ << endl ;
        else
             cout  << “Grade is F “ << endl ;
}
```

ตัวอย่าง
โปรแกรมหาค่าที่มากที่สุดของเลข 3 จำนวนใดๆ

```
// Program : Largest of three numbers
#include <iostream>
using namespace std;
double larger(double x , double y);
double comparethree(double x ,double y , double z);
int main()
{
  double one ,two ;
  cout << “The larger of 5 and 10 is “ << larger(5 , 10 ) << endl;
  cout << “Enter two numbers :” ;
  cin >> one >> two ;
  cout << endl ;
  cout << “the larger of “ << one <<”and “ << two << “is: << larger(one
       ,two) << endl;
  cout << “The largers of 23 , 34 , and 12 is “ << compareThree(23 , 34 ,
       12 ) << endl ;
  return 0 ;
}
```

```
double larger
          (double x , double y )
{
     if ( x>= y )
           return x ;
    else
          return y ;
}
double compareThree
           (double x ,
            double y ,
           double z)
{
     return larger( x, larger (y ,
      z)) ;
}
```

ทดสอบ

```
The larger of 5 and 10 is 10
Enter two numbers : 25  73
The larger of 25 and 73 is 73
The largest of 23 , 34 , and 12
      is 34
```

คำสั่ง switch

switch Statement

รูปแบบ:

```
switch (selector)
{
    case label1  : statements1 ;
                   break ;
    case label2  : statements2 ;
                   break ;
    . . .
    case labeln : statementsn ;
                   break ;
    default :  statementsd ; // optional
}
```

ตัวอย่างการเลือกโน๊ตเพลง

```
// Display a musical note
switch (musicalNote)
{
        case ‘c’ : cout << “do” ;
                   break ;
        case ‘d’ : cout << “re” ;
                   break ;
        case ‘e’ : cout << “mi” ;
                   break ;
        case ‘f’ : cout << “fa” ;
                   break ;
        case ‘g’ : cout << “sol” ;
                   break ;
        case ‘a’ ; cout << “la” ;
                   break ;
        case ‘b’ : cout << “ti” ;
                   break ;
        default :  cout << “An invalid note was read. “ << endl ;
}
```

ตัวอย่าง
เปรียบเทียบคำสั่ง if และ คำสั่ง switch

```
if  ((momOrDad == 'M') || (momOrDad == 'm'))
        cout << "Hello Mom – Happy Mother's Day" << endl ;
    else
        cout << "Hello Dad – Happy Father's Day" << endl ;
```

```
switch (momOrDad)
    {
        case 'M' : case 'm' :
            cout << "Hello Mom – Happy Mother's Day " << endl ;
            break ;
        case 'D' : case 'd' :
            cout << "Hello Dad – Happy Father's Day " << endl ;
          break ;
    }
```

ตัวอย่าง ฟังก์ชันการแสดงเกรด

```
void displayGrade (int score)
{
        switch (score/10 )
        {
         case 9 : case 10 :
                cout  << “Grade  is  A “   << endl ;    break ;
        case 8 :
                cout  << “Grade  is  B “   << endl ;    break ;
        case 7 :
                cout  << “Grade  is  C “   << endl ;    break ;
        case 6 :
                cout  << “Grade  is  D “    << endl ;  break ;
        default :
                cout  << “Grade is F “     << endl ;
        }
}
```

แบบฝึกหัด 1

- จงเขียนฟังก์ชันตรวจสอบตัวเลข

  กรณีเป็นเลขจำนวนบวก จะส่งค่า 1 กลับไปยังจุดเรียกใช้

  กรณีเป็นเลขจำนวนลบ จะส่งค่า -1 กลับไปยังจุดเรียกใช้

  กรณีเป็นเลขจำนวนศูนย์ จะส่งค่า 0 กลับไปยังจุดเรียกใช้

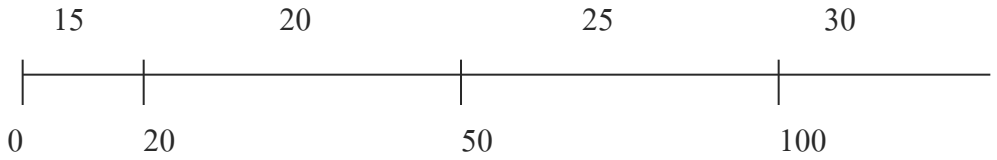
แบบฝึกหัด 2

- รับเลขจำนวนเต็ม 5 จำนวนทางแป้นพิมพ์ จงเขียนโปรแกรมหาเลขที่มีค่าน้อยที่สุด โดยเขียนเป็นฟังก์ชัน

แบบฝึกหัด 3

- จงเขียนโปรแกรมคิดค่าน้ำ โดยป้อนเลขที่มิเตอร์ใหม่ และมิเตอร์เดือนที่แล้วทางแป้นพิมพ์ หาค่าน้ำที่ต้องจ่ายจากอัตราดังนี้

จำนวนมิเตอร์ที่ใช้(ยูนิค)	อัตราจ่าย(บาทต่อยูนิค)
0-20	15
21-50	ส่วนที่เกิน 20 คิดยูนิคละ20 บาท
51-100	ส่วนที่เกิน 50 คิดยูนิคละ 25 บาท
>100	ส่วนที่เกิน 100 คิดยูนิคละ 30 บาท



แบบฝึกหัด 4

- จงเขียนโปรแกรมรับวัน เดือน ปี ปัจจุบัน และวัน เดือน ปี วันเกิดของบุคคลใดๆ จงหาอายุของบุคคลคนนี้ว่ามีอายุเท่ากับกี่ปี กี่เดือน กี่วัน

  กำหนดให้ 1 ปี เท่ากับ 12 เดือน

  1 เดือน เท่ากับ 30 วัน