

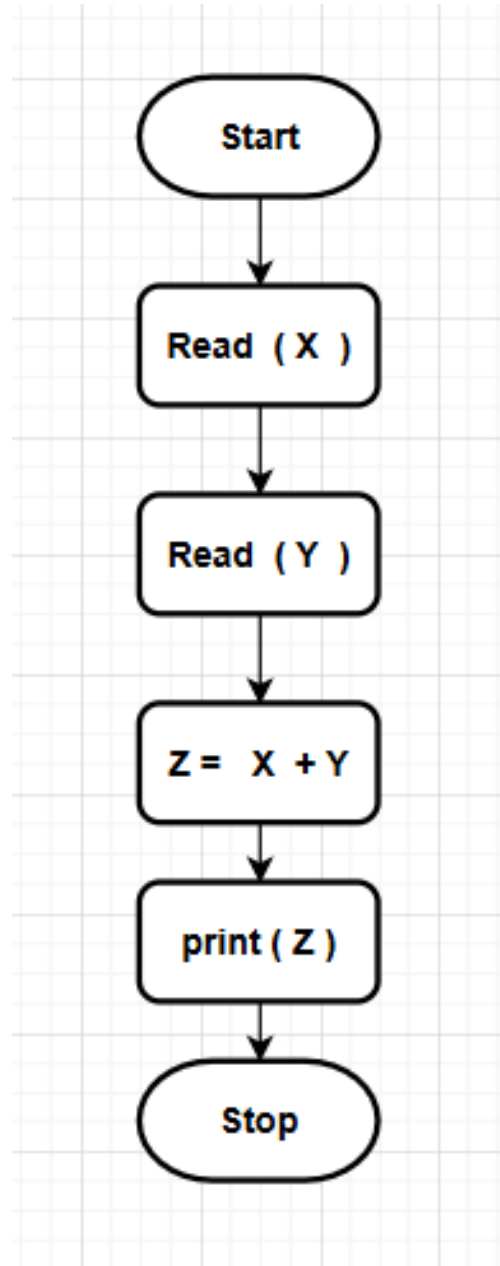
การเขียนโปรแกรมโดยใช้ภาษาสัญลักษณ์

flowchart basics

Sequence flow problem

Sequential Structure

ทำทีละบรรทัด จากบนลงล่าง



Assignment Statement

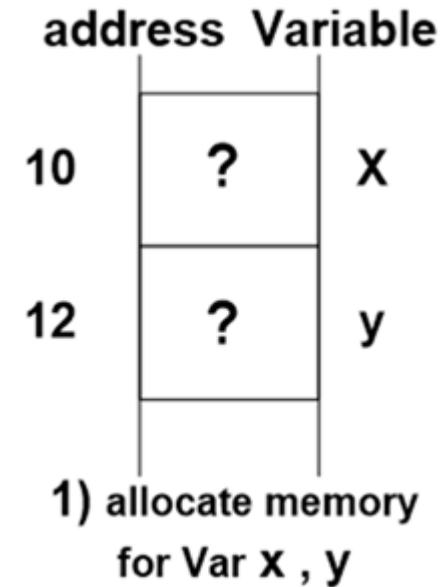
- **Assignment Operator** คือเครื่องหมายที่ใช้ในการกำหนดค่าให้กับตัวแปร(เริ่มต้น) หรืออัปเดต (update) ค่าที่จัดเก็บอยู่ในหน่วยความจำ ณ ตำแหน่งที่ตัวแปรเป็นตัวแทน

สัญลักษณ์	ใช้ในภาษา	ความหมาย
=	C, C++, Java, Python, Swift, JavaScript, Go ฯลฯ	กำหนดค่า (assignment) เช่น <code>x = 5</code>
:=	Pascal, Go, Swift (บางกรณี), Ada	กำหนดค่า (assignment แบบชัดเจนว่าไม่ใช่เท่ากับ)
<-	R, Haskell	กำหนดค่า เช่น <code>x <- 5</code>

Variable vs Memory State

VarDemo.pas

```
1 program VariableExample;  
2 var  
3   x, y: Integer;  
4 begin  
5   x := 50;  
6   y := 80;  
7   x := y;  
8   y := 90;  
9   x := x + y;  
10  WriteLn(y);  
11  WriteLn(x);  
12 end.
```

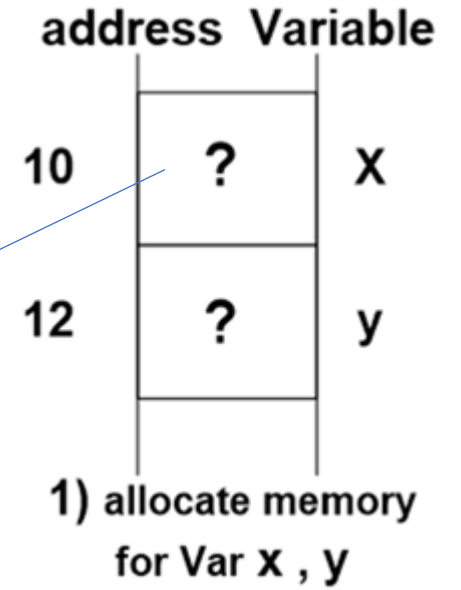
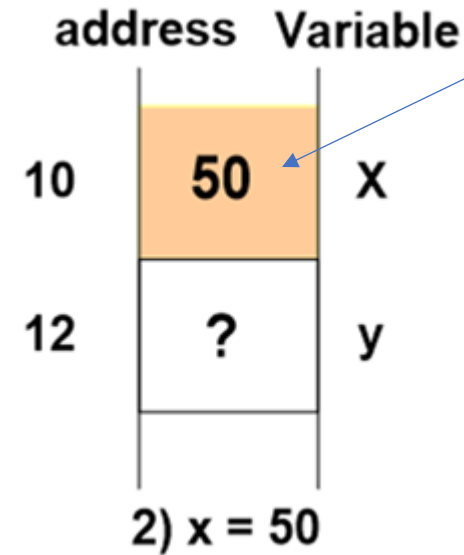


p x
p y

Variable vs Memory State

VarDemo.pas

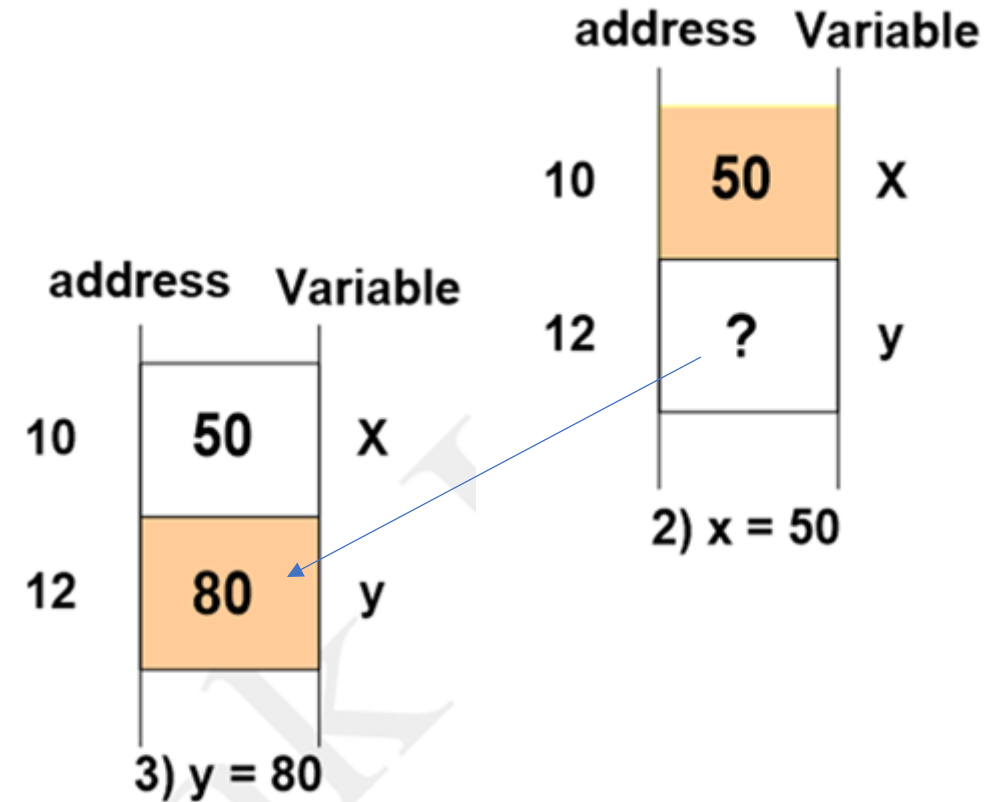
```
1 program VariableExample;  
2 var  
3   x, y: Integer;  
4 begin  
5   x := 50;  
6   y := 80;  
7   x := y;  
8   y := 90;  
9   x := x + y;  
10  WriteLn(y);  
11  WriteLn(x);  
12 end.
```



Variable vs Memory State

VarDemo.pas

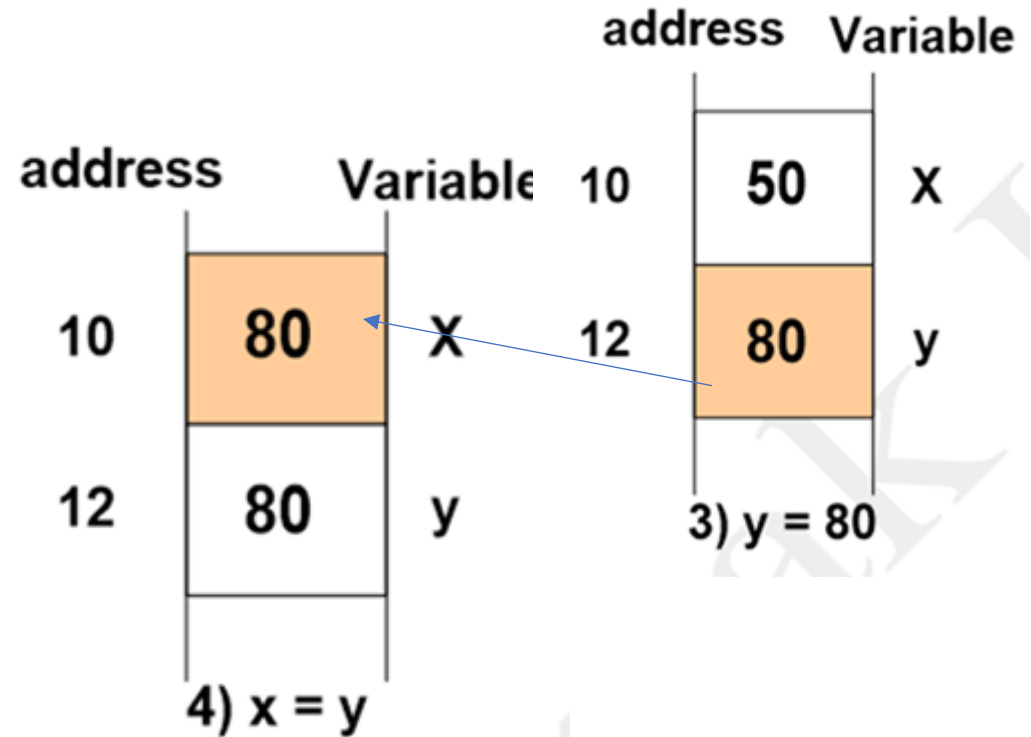
```
1 program VariableExample;
2 var
3   x, y: Integer;
4 begin
5   x := 50;
6   y := 80;
7   x := y;
8   y := 90;
9   x := x + y;
10  WriteLn(y);
11  WriteLn(x);
12 end.
```



Variable vs Memory State

VarDemo.pas

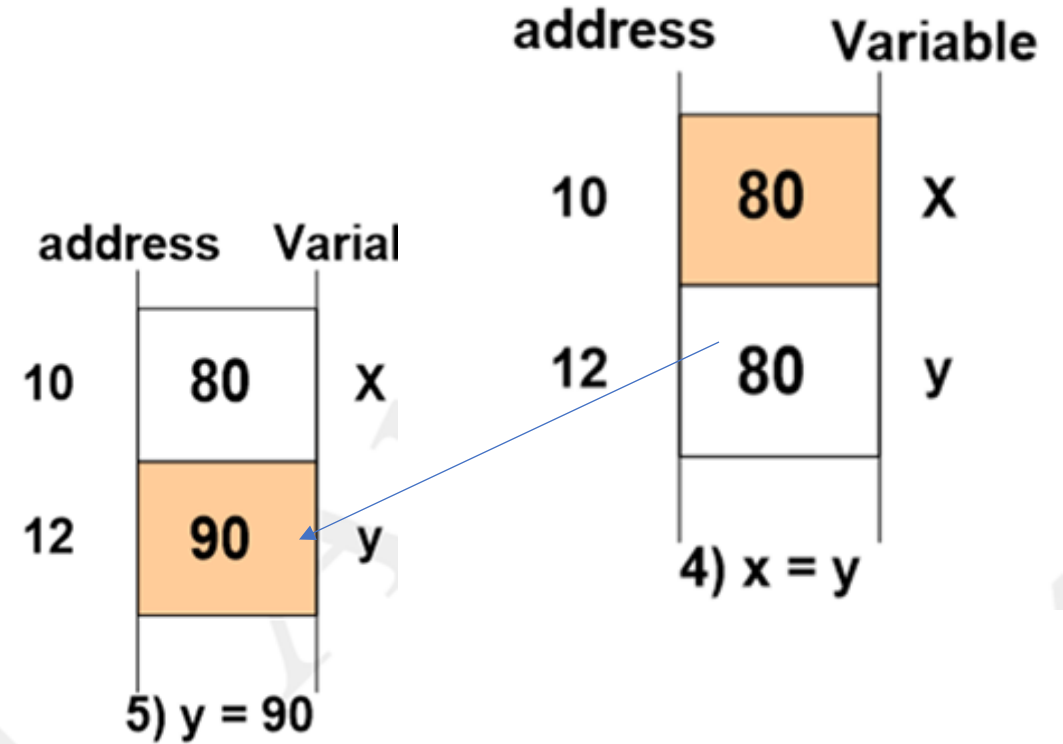
```
1 program VariableExample;
2 var
3   x, y: Integer;
4 begin
5   x := 50;
6   y := 80;
7   x := y;
8   y := 90;
9   x := x + y;
10  WriteLn(y);
11  WriteLn(x);
12 end.
```



Variable vs Memory State

VarDemo.pas

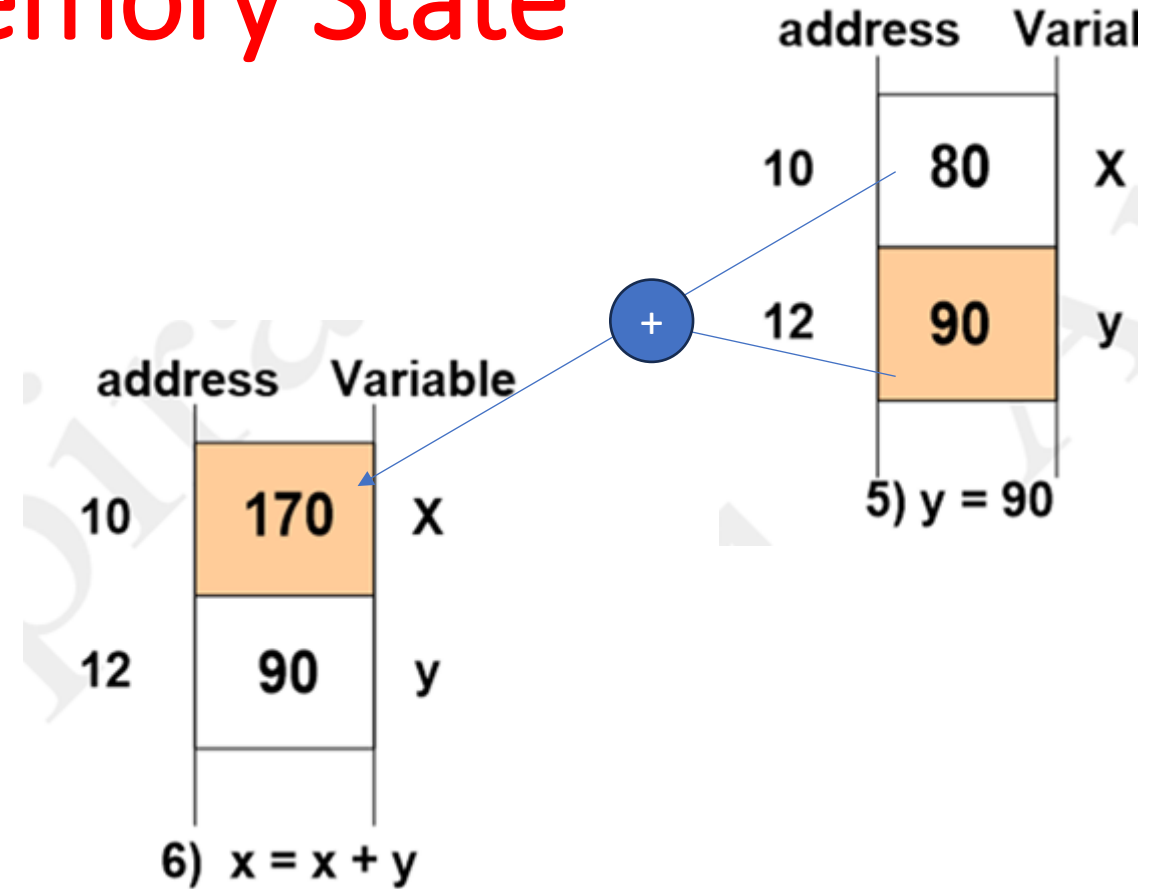
```
1 program VariableExample;  
2 var  
3   x, y: Integer;  
4 begin  
5   x := 50;  
6   y := 80;  
7   x := y;  
8   y := 90;  
9   x := x + y;  
10  WriteLn(y);  
11  WriteLn(x);  
12 end.
```

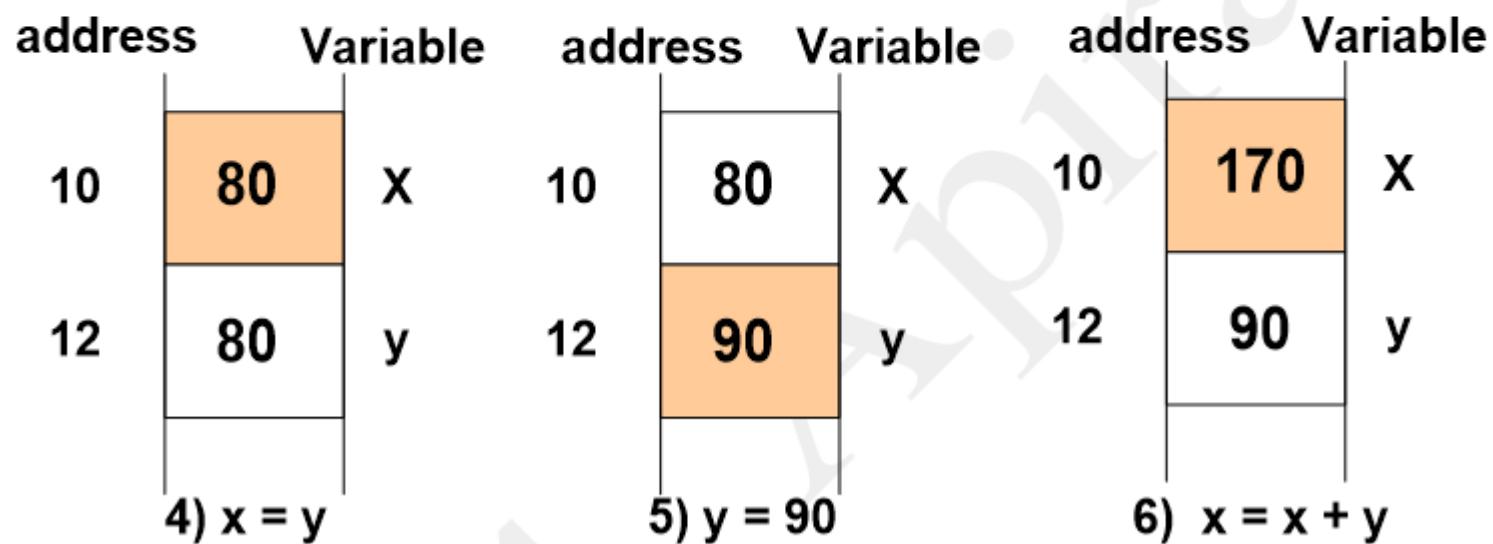
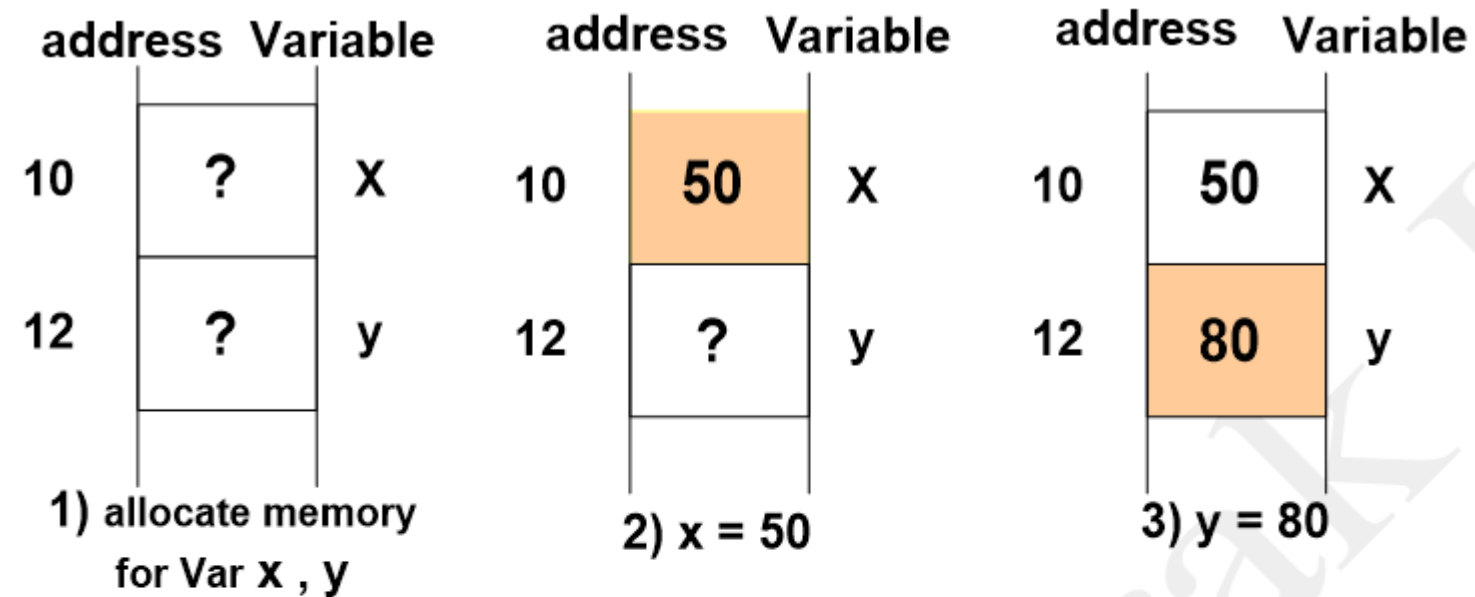


Variable vs Memory State

VarDemo.pas

```
1 program VariableExample;  
2 var  
3   x, y: Integer;  
4 begin  
5   x := 50;  
6   y := 80;  
7   x := y;  
8   y := 90;  
9   x := x + y;  
10  WriteLn(y);  
11  WriteLn(x);  
12 end.
```





รูปที่ 3.7 การเปลี่ยนสถานะตัวแปรของโปรแกรมรูปที่ 3.6

Expression

- **Expression** (เอ็กสเพรสชัน) ในภาษาโปรแกรม คือ “คำสั่งที่มีการใช้เครื่องหมาย operator $+$, $-$, $*$, $/$, $<$, $>$, and $,$ or เพื่อให้ได้ผลลัพธ์ในการคำนวณ” โดยผลลัพธ์และเงื่อนไขของการทำ expression ต่างๆขึ้นอยู่กับภาษาโปรแกรม แต่โดยทั่วไปจะใกล้เคียงกัน

```
1 program VariableExample;  
2 var  
3   x, y: Integer;  
4 begin  
5   x := 50;  
6   y := 80;  
7   x := y;  
8   y := 90;  
9   x := x + y;  
10  WriteLn(y);  
11  WriteLn(x);  
12 end.
```

VarDemo.pas

$x := 50$ -> ไม่ใช่ expression คำสั่งเก็บข้อมูล

$x := x + y$

expression

Arithmetic Expression

- ตัวแปรหรือค่าคงที่ที่นำมาเป็น **input** ในการทำ **operation** (+,-,/,*,div,mod) บน **expression** ต้องมีชนิดข้อมูล(**type**) เดียวกัน โดยชนิดข้อมูลของผลลัพธ์ยังคงมีชนิดเดียวกับ **input** เช่น

A	+	B	->	result
real	+	real	ผลลัพธ์	real/float
real	+	integer	ผลลัพธ์	real/float
integer	+	integer	ผลลัพธ์	integer

Result type

Float	op	Float	➡	Float
2.0	+	3.0		5.0
4.0	/	3.0		1.3
Integer	op	integer	➡	integer
2	+	3		5
4	div	3		1
Integer	/	integer		real
4	/	3		1.3

BASED ON PASCAL

main.pas

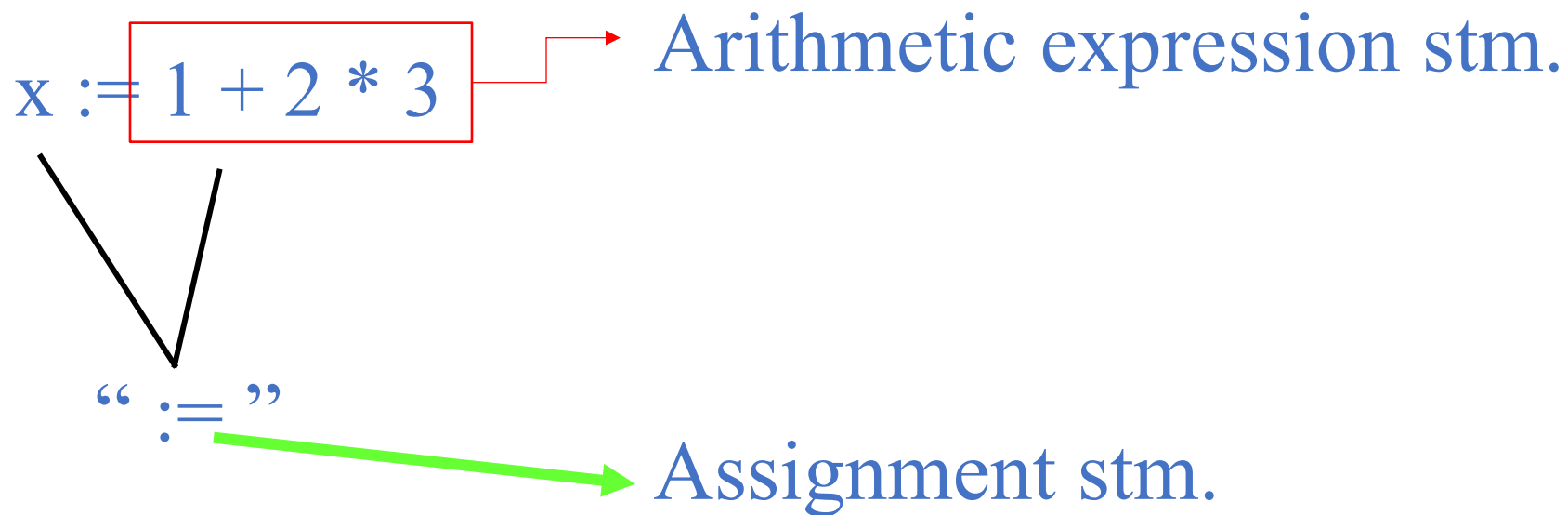
```
1 program VariableExample;
2 var
3     x, y: Integer;
4 begin
5     WriteLn('2+3 = ', 2+3);
6     WriteLn('4 div 3 = ', 4 div 3);           // integer division
7     WriteLn('4 mod 3 = ', 4 mod 3);           // remainder
8     WriteLn('4 / 3 = ', 4 / 3);               // real division
9     WriteLn('2.0+3 = ', 2.0+3);
10    WriteLn('2.0+3.0 = ', 2.0+3.0);
11    WriteLn('4.0/3.0 = ', 4.0/3.0);           // real division
12 end.
13
```

OprDemo.pas

```
12 lines compiled, 0.0 sec
2 note(s) issued
2+3 = 5
4 div 3 = 1
4 mod 3 = 1
4 / 3 = 1.3333333333333333E+000
2.0+3 = 5.000000000E+00
2.0+3.0 = 5.000000000E+00
4.0/3.0 = 1.333333373E+00
```

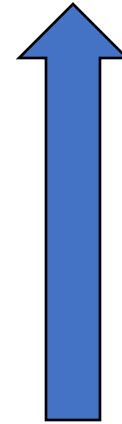
Precedence Operator

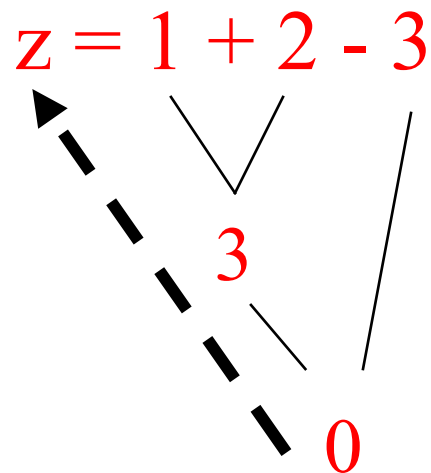
กฎที่กำหนดว่า ในนิพจน์ (expression) ที่มีหลายตัวดำเนินการ (operators) จะต้องคำนวณตัวไหนก่อนตัวไหน เช่น การบวก ลบ คูณ หาร หรือเปรียบเทียบต่างๆ



ขึ้นอยู่กับกฎเกณฑ์ของภาษาโปรแกรม ใน **course** นี้จะใช้กฎเกณฑ์เดียวกับภาษา **Pascal**

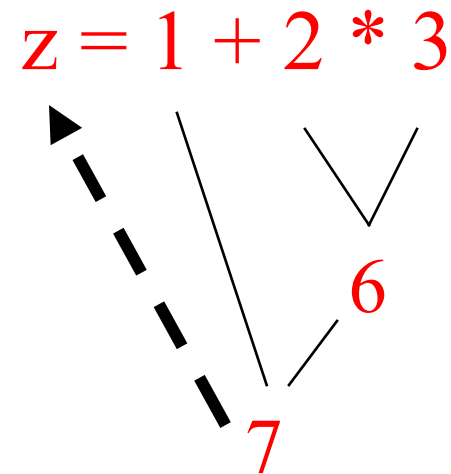
1. Minus sign used to change sign (-)
2. Multiplication and division (* , /, div)
3. Addition and subtraction (+ , -)
4. assign (=)



$$z = 1 + 2 - 3$$


3

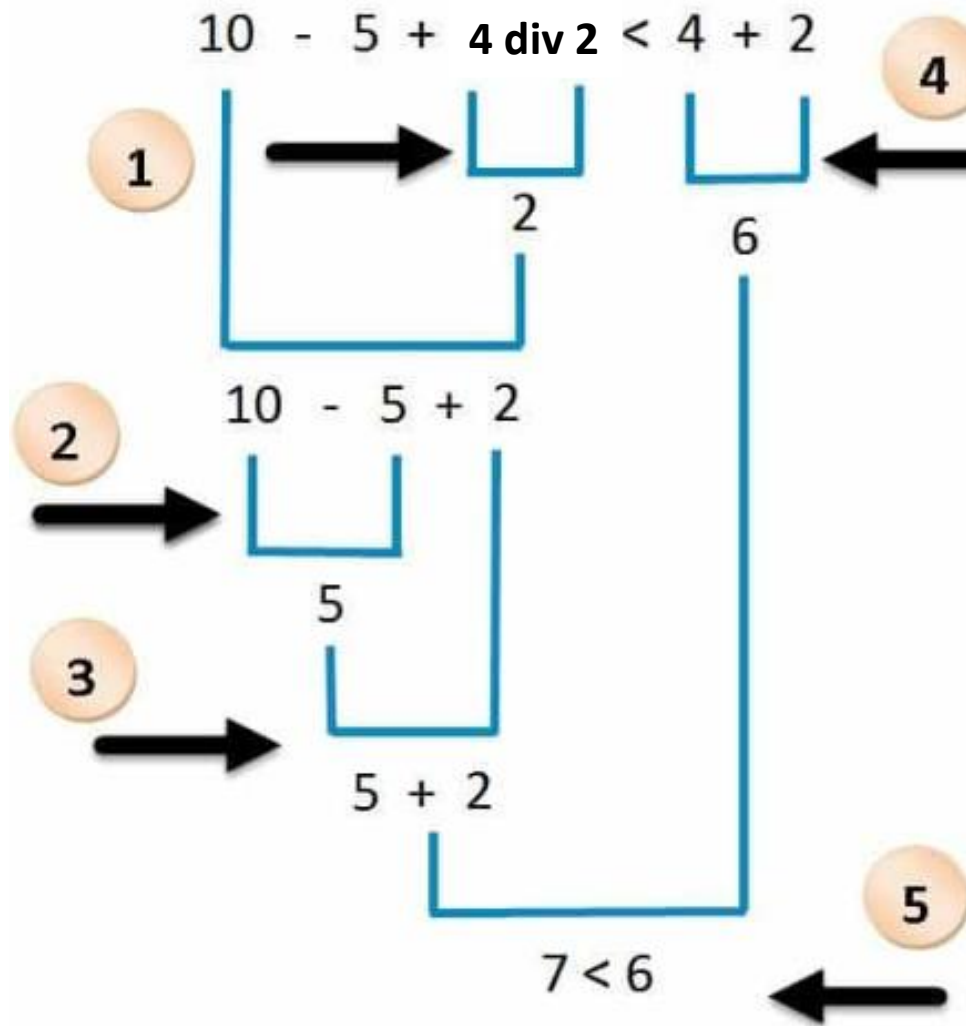
0

$$z = 1 + 2 * 3$$


6

7

z: integer ;



```
main.pas
1 program VariableExample;
2 begin
3   WriteLn('result=', 10-5+4 div 2 < 4+2);
4 end.
```

OprEvl.pas

Ans : FALSE

Type conversion & assignment

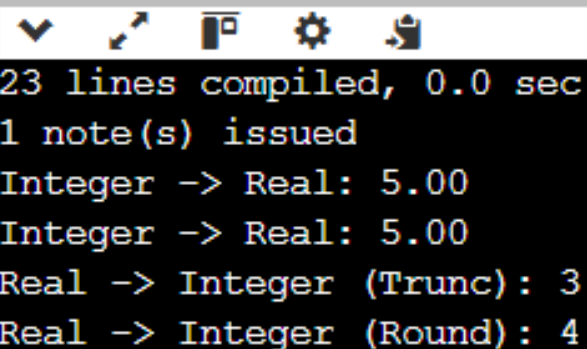
- กระบวนการเปลี่ยนชนิดของค่าข้อมูล (**Data Type**) จากชนิดหนึ่งไปเป็นอีกชนิดหนึ่ง เพื่อให้สามารถใช้งานหรือประมวลผลข้อมูลได้อย่างเหมาะสมในโปรแกรม

การแปลง	ฟังก์ชัน / วิธี	ตัวอย่างโค้ด	ผลลัพธ์
Integer → Real	Implicit	<code>r := i;</code>	<code>i = 5 → r = 5.00</code>
Integer → Real	<code>Real(i)</code>	<code>r := real(i);</code>	<code>i = 5 → r = 5.00</code>
Real → Integer (ตัดทศนิยม)	<code>Trunc(r)</code>	<code>i := Trunc(r);</code>	<code>r = 3.7 → i = 3</code>
Real → Integer (ปัดเลขใกล้เคียง)	<code>Round(r)</code>	<code>i := Round(r);</code>	<code>r = 3.7 → i = 4</code>
Real -> Integer	error	<code>r := i;</code>	Compile error

`r : real , i: integer , s: string`

ความสามารถในการแปลงขึ้นอยู่กับภาษาโปรแกรม เพื่อความง่ายใน course จะใช้คุณสมบัติของ free pascal

```
1 program TypeConversionExample;
2 var
3   i: integer;
4   r: real;
5 begin
6   // 1. Integer -> Real
7   i := 5;
8   r := i; // ใน Pascal สามารถกำหนดตรง ๆ ก็ได้
9   WriteLn('Integer -> Real: ', r:0:2); // แสดง 5.00
10  //
11  r := real(i); // ใน Pascal สามารถกำหนดตรง ๆ ก็ได้
12  WriteLn('Integer -> Real: ', r:0:2); // แสดง 5.00
13
14  // 2. Real -> Integer
15  r := 3.7;
16  i := Trunc(r); // ตัดทศนิยม
17  WriteLn('Real -> Integer (Trunc): ', i);
18  i := Round(r); // ปัดเป็นจำนวนเต็ม
19  WriteLn('Real -> Integer (Round): ', i);
20  //i := r; { compiler error }
21 end.
22
```



23 lines compiled, 0.0 sec
1 note(s) issued
Integer -> Real: 5.00
Integer -> Real: 5.00
Real -> Integer (Trunc): 3
Real -> Integer (Round): 4

TypeConv.pas

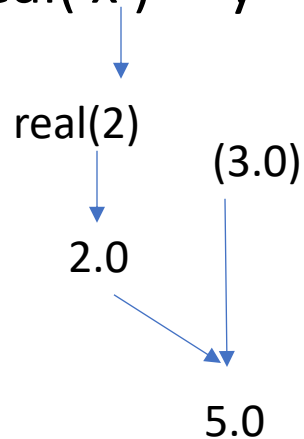
Mixed mode Operation

mixedm_1.pas

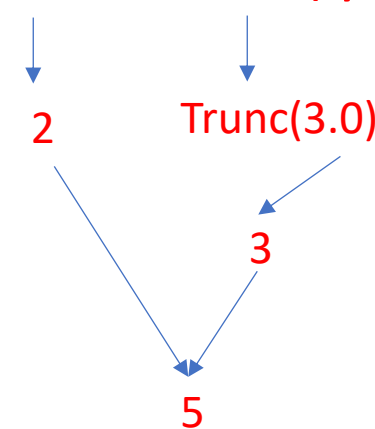
- ภาษาโปรแกรมหลายๆโปรแกรมสนับสนุนเรื่องการแปลง **data type** เราเรียกว่า “**explicit type conversion**” มีการใช้คำสั่งที่แน่นอนในการ

x := 2 ; y := 3.0

real(x) + y



x + Trunc(y)



```
main.pas
1 program VariableExample;
2 var
3   x:integer;
4   y:real;
5 begin
6   x := 2;
7   y := 3.0;
8   WriteLn('result-1=', x + y );
9   WriteLn('result-2=', x + Trunc(y) );
10  WriteLn('result-3=', real(x) + y );
11  WriteLn('result-4 =', x / y );
12 end.
```

```
result-1= 5.0000000000000000E+000
result-2=5
result-3= 5.0000000000000000E+000
result-4 = 6.6666666666666663E-001
```

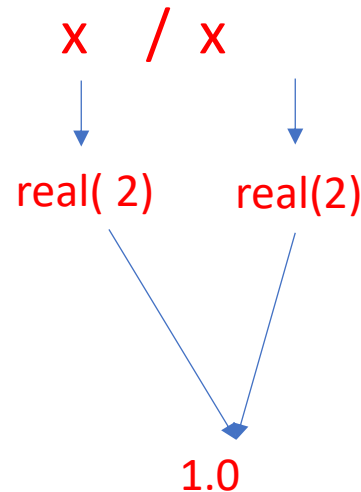
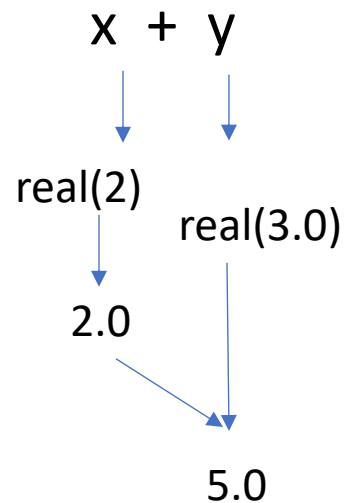
explicit type conversion

impConv.pas

Mixed mode Operation

- ภาษาโปรแกรมหลาย ๆ โปรแกรมสนับสนุนเรื่องการแปลง **data type** เราเรียกว่า **“implicit conversion”**
- integer + real -> real**
Integer / integer -> real

x := 2 ; y := 3.0



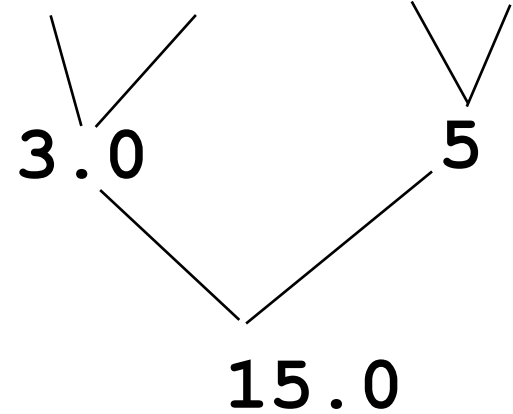
```
main.pas
1 program VariableExample;
2 var
3   x:integer;
4   y:real;
5 begin
6   x := 2;
7   y := 3.0;
8   WriteLn('result-1=', x + y );
9   WriteLn('result-2=', x / x );
10  WriteLn('result-3=', y / y );
11 end.
12
```

```
Linking a.out
11 lines compiled, 0.0 sec
< result-1= 5.0000000000000000E+000
result-2= 1.0000000000000000E+000
result-3= 1.0000000000000000E+000
```

Implicit type conversion

ตัวอย่าง # 1. Implicit Type conversion

`(1.0+2.0) * (2+3) ;`



Precedence	Operator
	<code>++ --</code>
	<code>()</code>
1	<code>[]</code> <code>.</code> <code>-></code> <code>(type){list}</code>
2	<code>++ --</code> <code>+ -</code> <code>! ~</code> <code>(type)</code> <code>*</code> <code>&</code> <code>sizeof</code> <code>_Alignof</code>
3	<code>* / %</code>
4	<code>+ -</code>
5	<code><< >></code>
6	<code>< <=</code> <code>> >=</code>
7	<code>== !=</code>
8	<code>&</code>
9	<code>^</code>
10	<code> </code>
11	<code>&&</code>
12	<code> </code>
13	<code>? :</code>

ตัวอย่าง #2. Implicit Type conversion

$(4+2.1) + (2+3.0) ;$

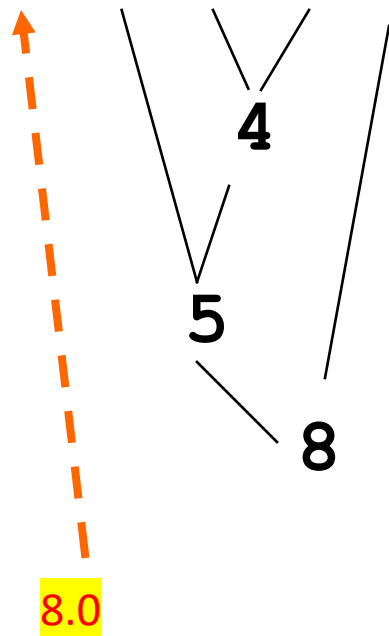
Diagram illustrating implicit type conversion:

- $4 + 2.1$ results in 6.1 (double).
- $2 + 3.0$ results in 5.0 (double).
- The final result is 11.1 (double), as the integer results are converted to double before the final addition.

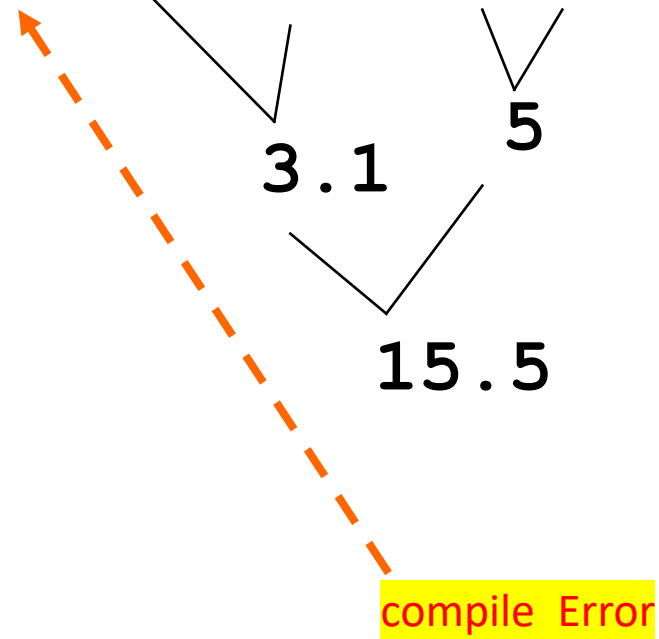
Precedence	Operator
	$++ --$
	$()$
1	$[]$ $.$ $->$ $(type)\{list\}$
2	$++ --$ $+ -$ $! ~$ $(type)$ $*$ $\&$ $sizeof$ $_Alignof$
3	$* / \%$
4	$+ -$
5	$<< >>$
6	$< <=$ $> >=$
7	$== !=$
8	$\&$
9	\wedge
10	$ $
11	$\&\&$
12	$ $
13	$? :$

x : real
 y : integer

$x := 1 + 2 * 2 + 3 ;$



$y := (1 + 2.1) * (2 + 3) ;$



Understand read() in pascal

main.pas

```
1 program CalculateNet;  
2 var  
3   w,x,y: real;  
4 begin  
5   Read( w , x , y);  
6   Writeln('W = ' , w:0:2 , ',X = ' , x:0:2 , ',Y = ' , y:0:2);  
7 end.  
8
```

readwxy.pas

Read ()

Read ()

Space or enter เท่ากัน

7 lines compiled, 0.0 sec

10 20 30

W = 10.00,X = 20.00,Y = 30.00

...Program finished with exit code

space

7 lines compiled, 0.0 sec

10

20

30

W = 10.00,X = 20.00,Y = 30.00

...Program finished with exit code 0

Press ENTER to exit console.

enter

main.pas

```
1 program CalculateNet;  
2 var  
3   w,x,y: real;  
4 begin  
5   Read( w );  
6   Read( x );  
7   Read( y );  
8   Writeln('W = ' , w:0:2 , ',X = ' , x:0:2 , ',Y = ' , y:0:2);  
9 end.  
10
```

```
Free Pascal Compiler version 3.2.2 [2021-01-10] 64-bit  
Copyright (c) 1993-2021 by Florian E. Schwaiblmair  
Target OS: Linux for x86-64  
Compiling main.pas  
Linking a.out  
9 lines compiled, 0.0 sec  
10 20 30  
W = 10.00,X = 20.00,Y = 30.00  
...Program finished with exit code 0  
Press ENTER to exit console.
```

Space

```
Linking a.out  
9 lines compiled, 0.0 sec  
20  
30  
< 40  
W = 20.00,X = 30.00,Y = 40.00  
...Program finished with exit code 0  
Press ENTER to exit console.
```

Enter

Readln()

readln.pas

main.pas

```
1 program CalculateNet;
2 var
3     w,x,y: real;
4 begin
5     Readln( w );
6     Readln( x );
7     Readln( y );
8     Writeln('W = ' , w:0:2 , ',X = ' , X:0:2 , ',Y = ' , y:0:2);
9 end.
10
```

ไม่ได้

```
Free Pascal Compiler version 3.2.2+
Copyright (c) 1993-2021 by Florian
Target OS: Linux for x86-64
Compiling main.pas
Linking a.out
9 lines compiled, 0.0 sec
10
20
30
W = 10.00,X = 20.00,Y = 30.00
```

```
Target OS: Linux for x86-64
Compiling main.pas
Linking a.out
9 lines compiled, 0.0 sec
10 11
20
30
W = 10.00,X = 20.00,Y = 30.00

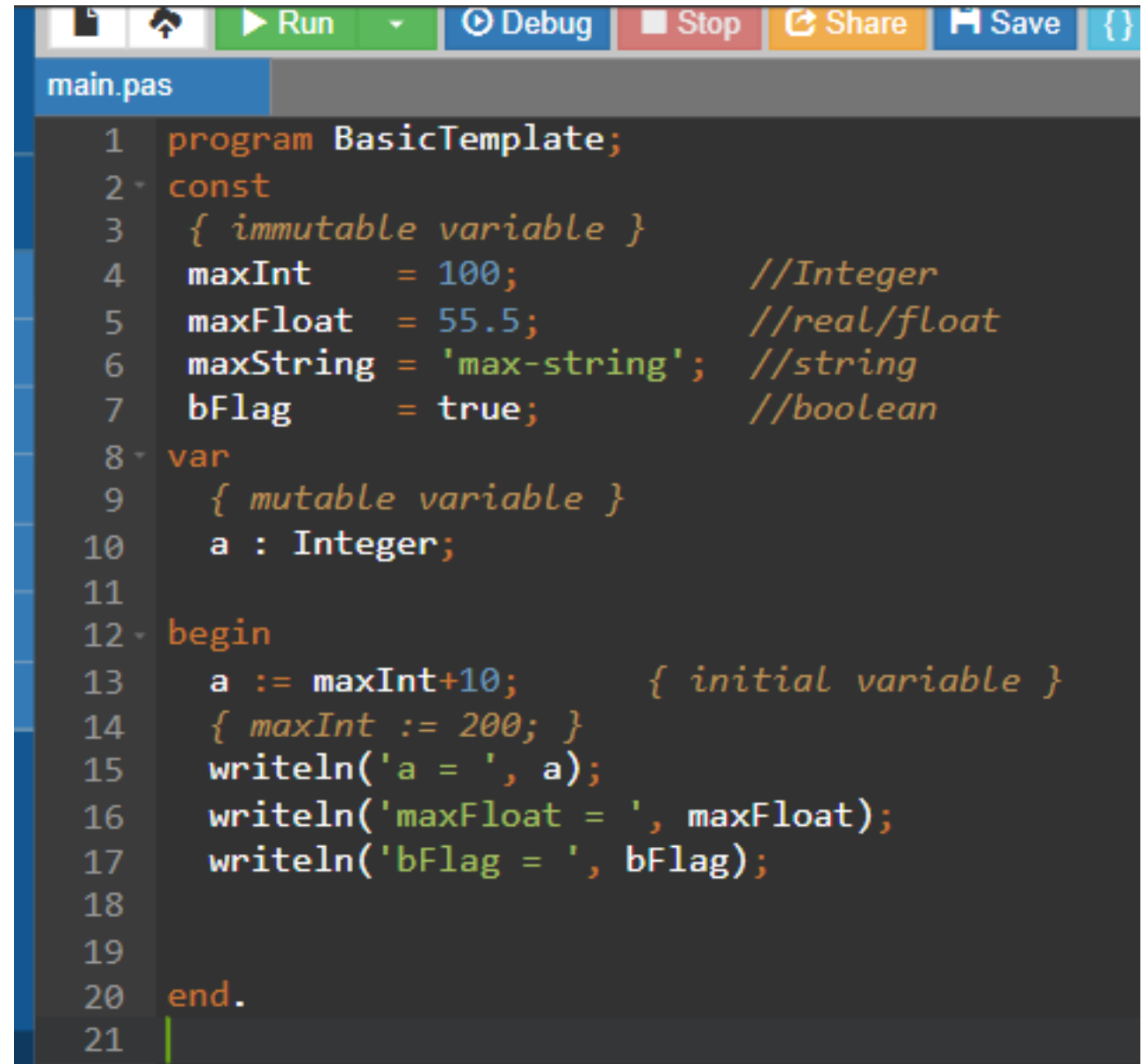
...Program finished with exit code 0
Press ENTER to exit console.
```

Pascal Program Template

Immutable & mutable variable

const.pas

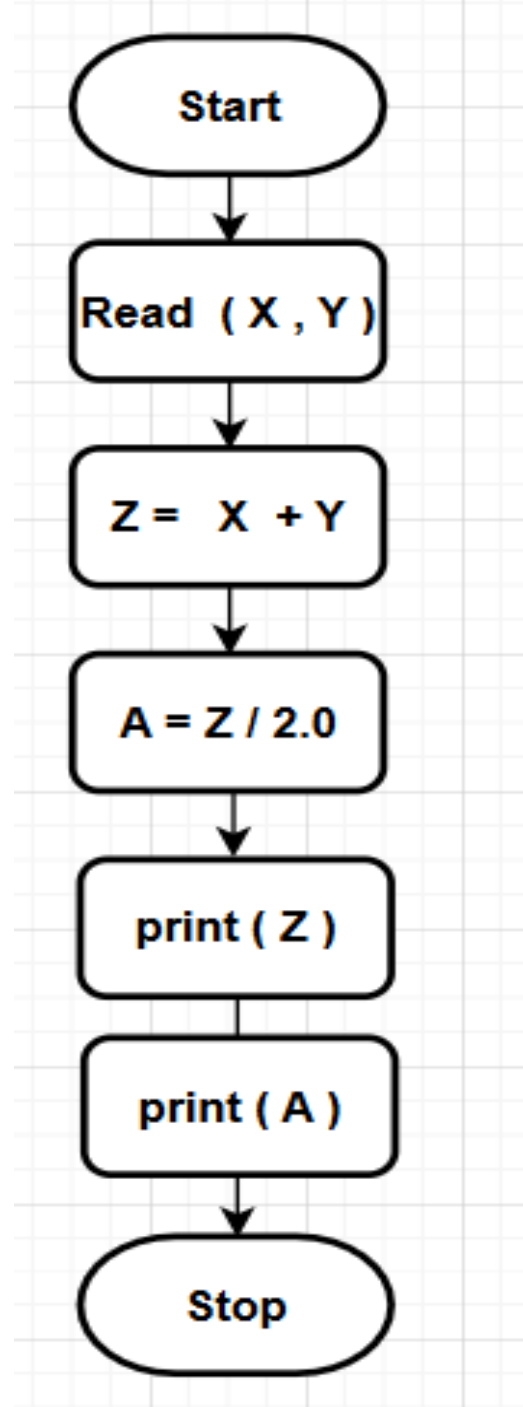
```
const
{ immutable variable }
maxInt   = 100;      //Integer
maxFloat = 55.5;     //real/float
maxString = 'max-string'; //string
bFlag    = true;     //boolean
```



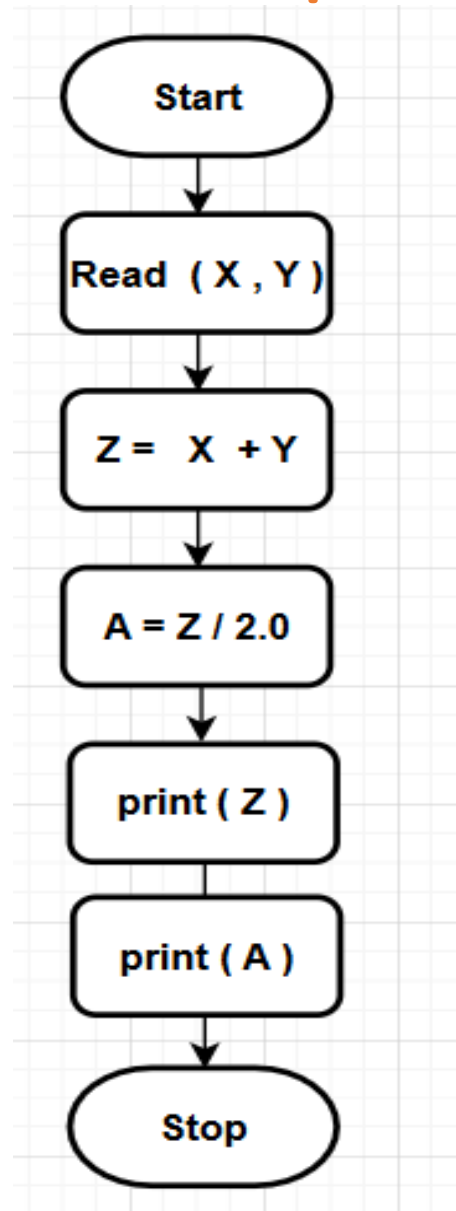
The screenshot shows a Pascal IDE window titled 'main.pas'. The code is a Pascal program template with the following structure:

```
1 program BasicTemplate;
2 const
3   { immutable variable }
4   maxInt   = 100;      //Integer
5   maxFloat = 55.5;     //real/float
6   maxString = 'max-string'; //string
7   bFlag    = true;     //boolean
8 var
9   { mutable variable }
10  a : Integer;
11
12 begin
13  a := maxInt+10;      { initial variable }
14  { maxInt := 200; }
15  writeln('a = ', a);
16  writeln('maxFloat = ', maxFloat);
17  writeln('bFlag = ', bFlag);
18
19
20 end.
21
```

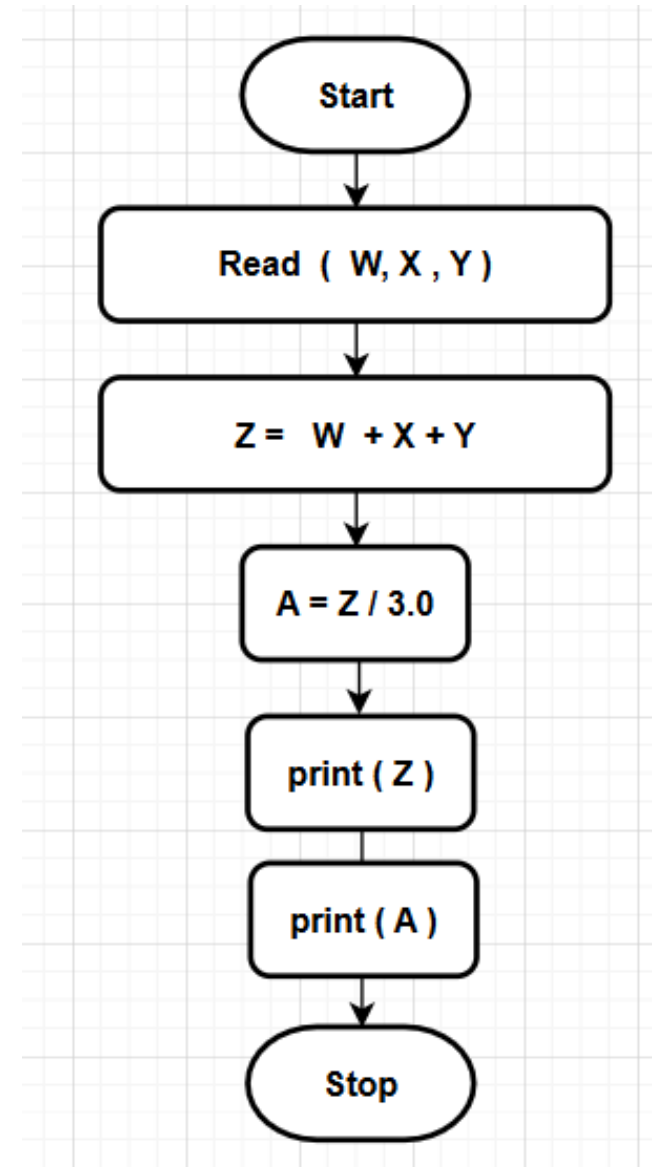
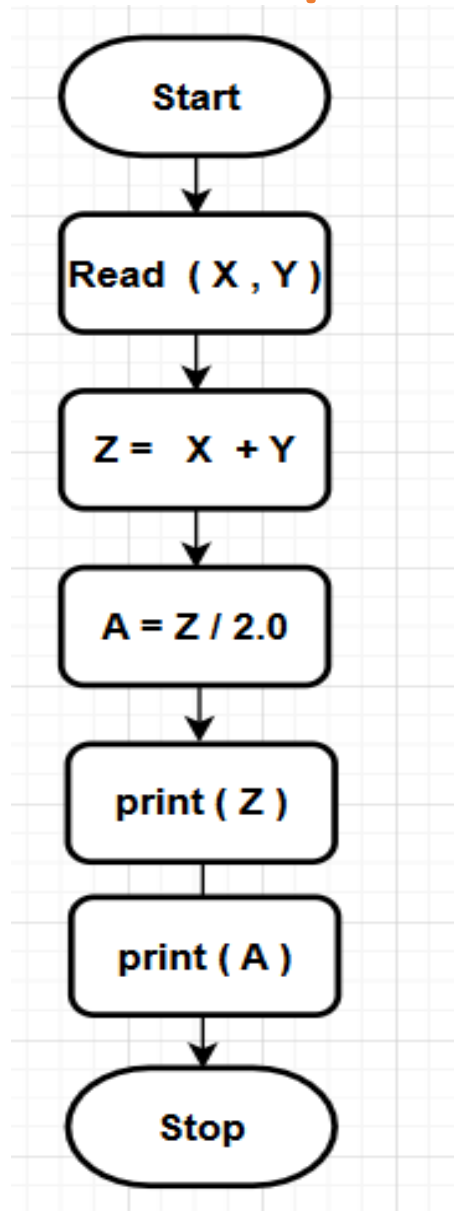
ฝึกอ่าน Sequential Flow



ฝึกเขียน Sequential Flow (Average 3 ตัวแปร)



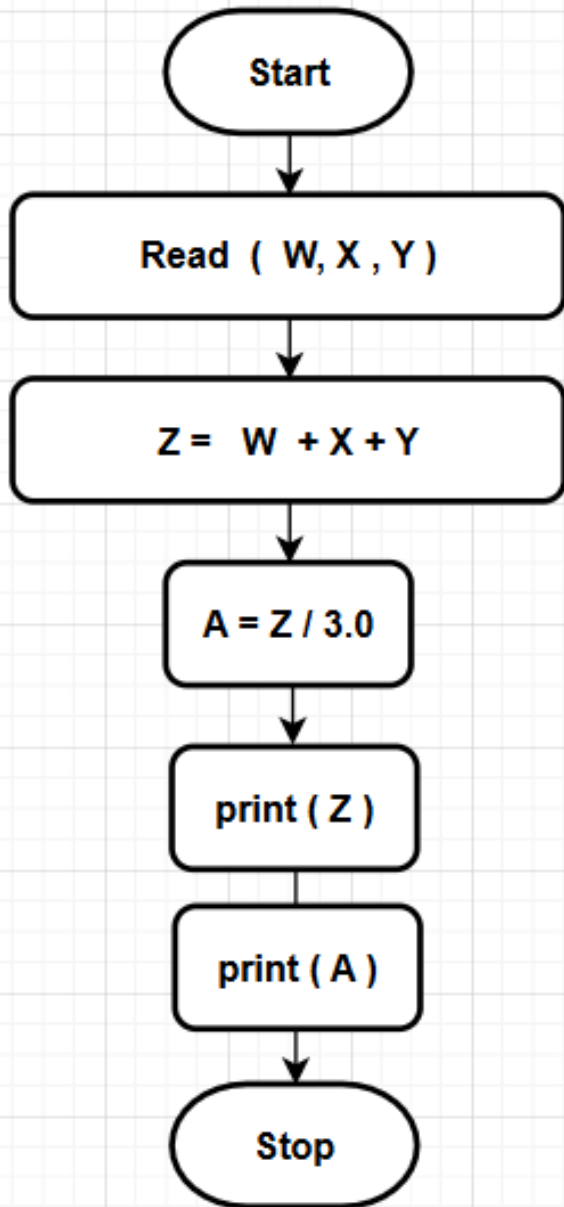
ฝึกเขียน Sequential Flow (Average 3 ตัวแปร)



Convert to pascal code

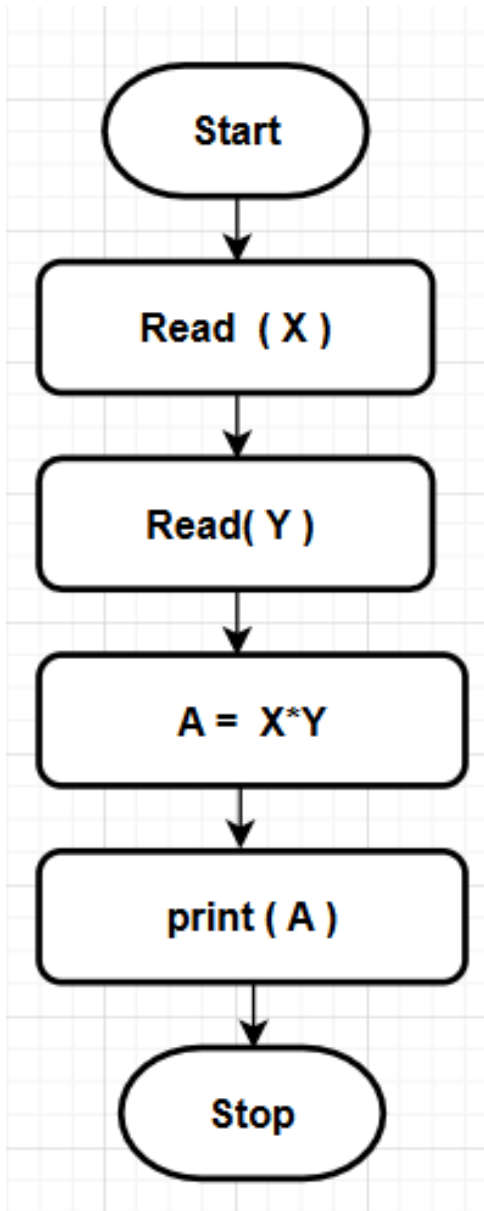
avgWXY.pas

Assume All variables are real/float



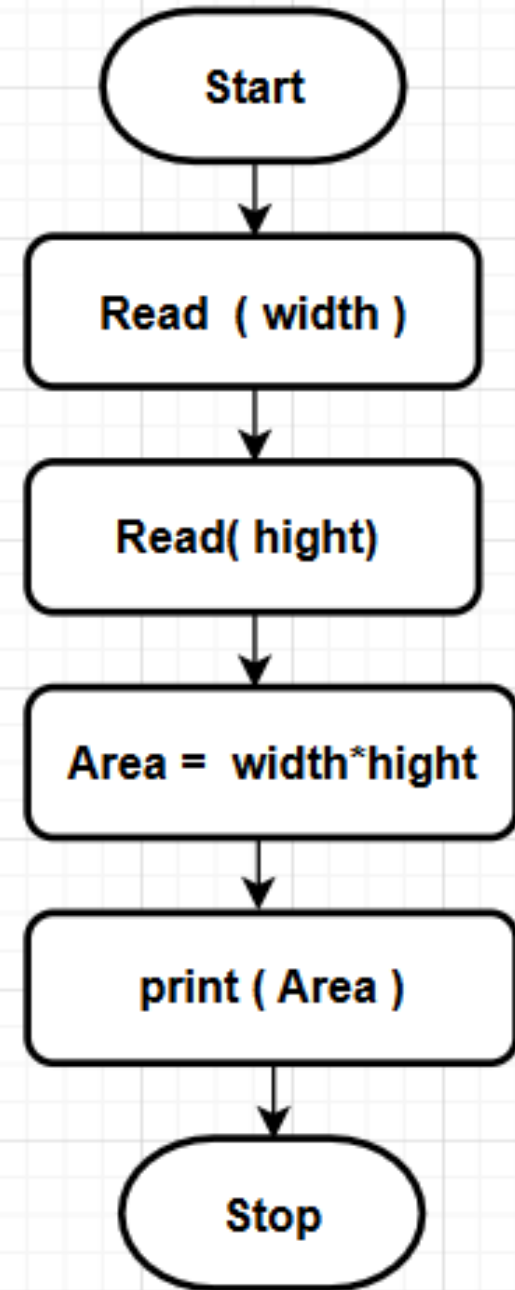
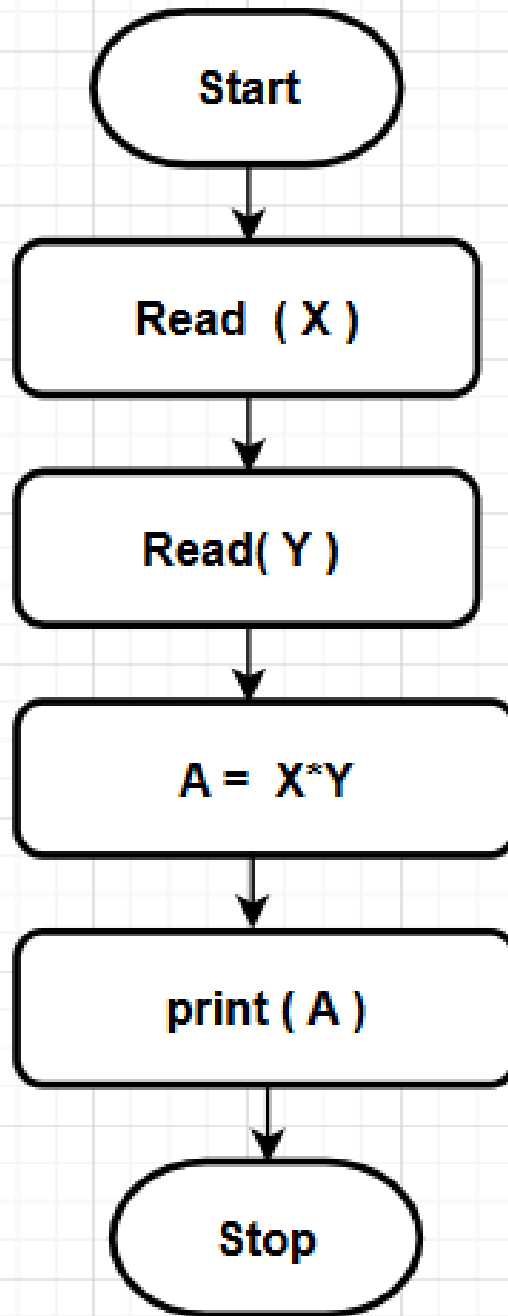
```
program CalculateNet;  
var  
  
begin  
  
end.
```

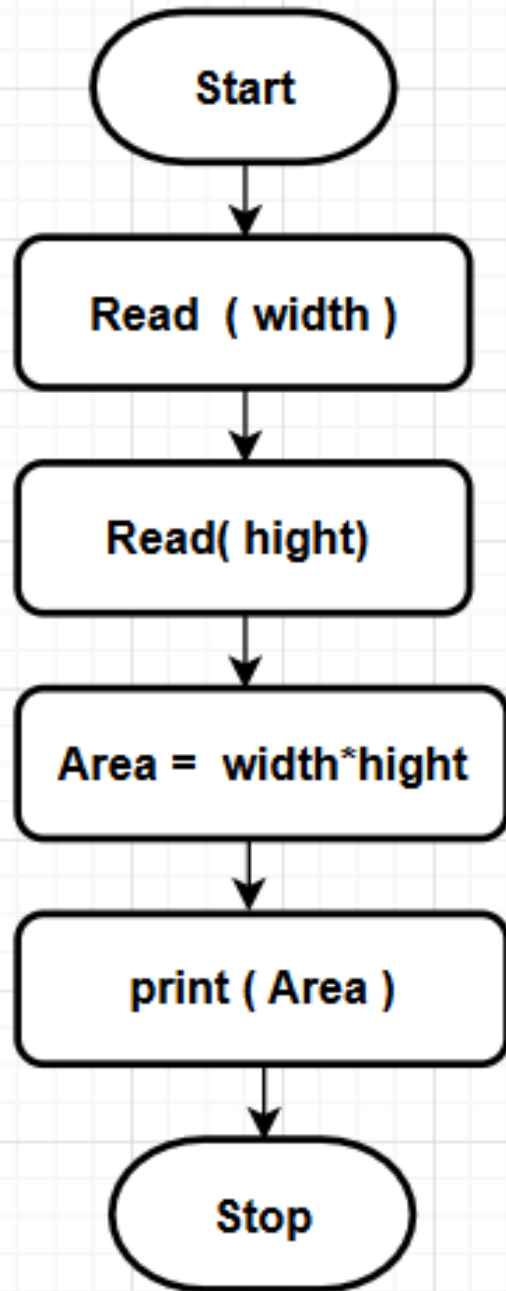

ฝึกอ่าน Sequential Flow



Change name of variable

- A คือ ผลคูณของ X และ Y
- ข้อมูลของ X และ Y ได้มาจากแป้นพิมพ์



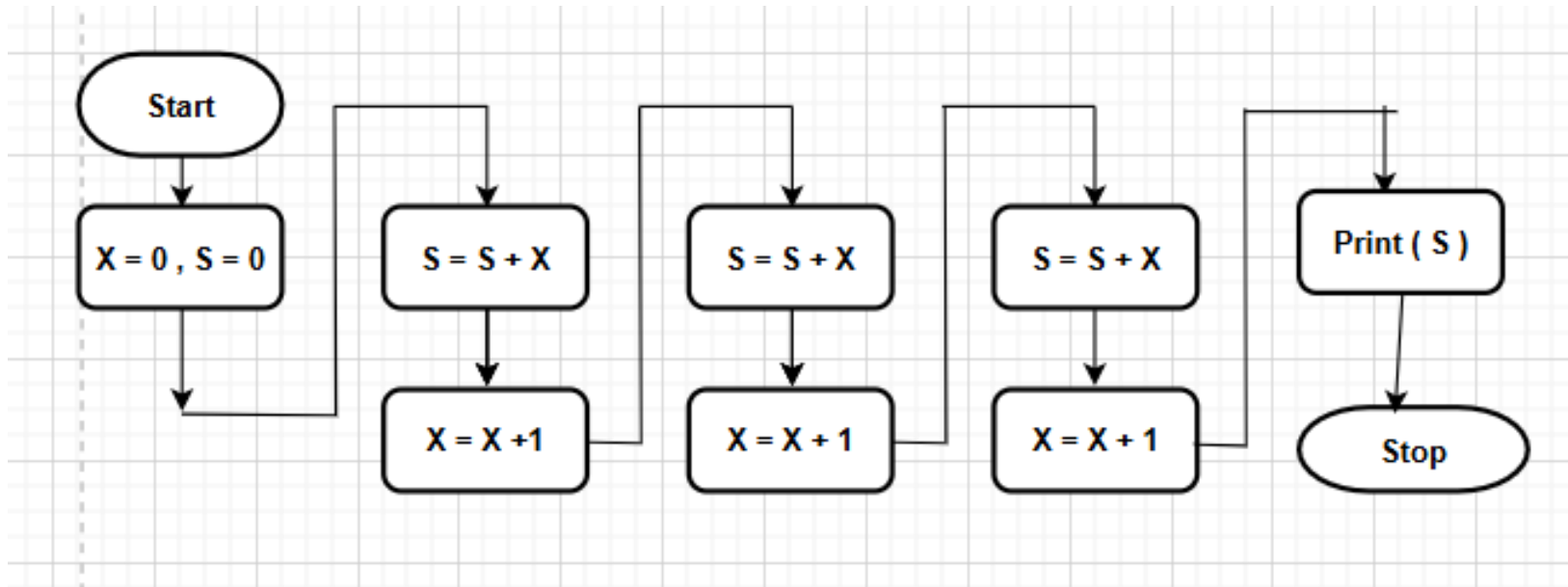


Convert to Pascal
Assume all variables are real

```
program CalculateNet;  
var  
  
begin  
  
end.
```

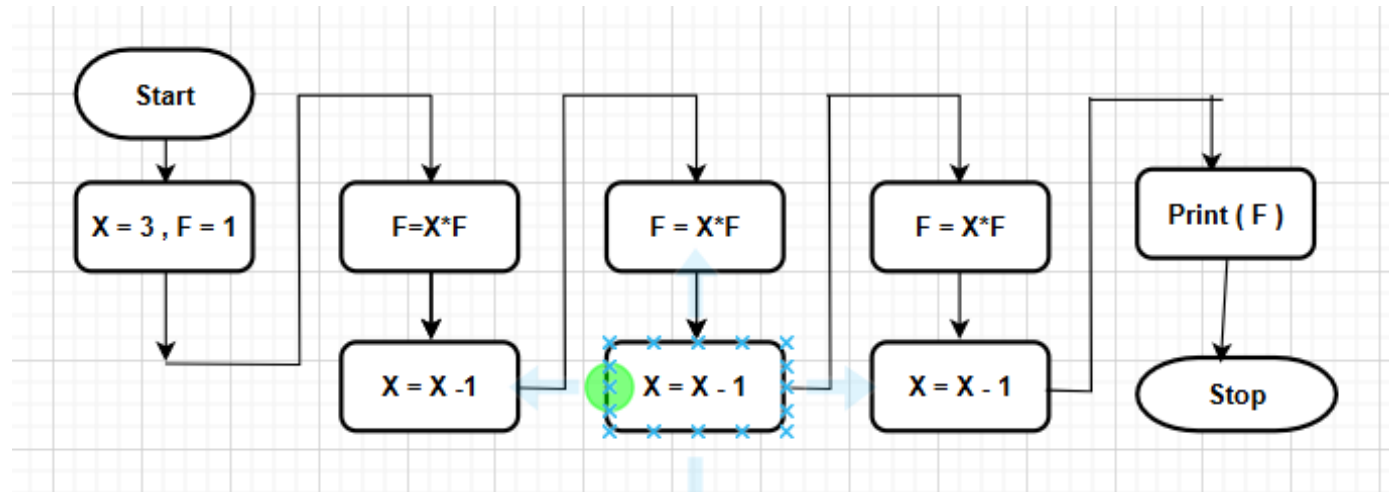
areaWH.pas

ฝึกอ่าน Sequential Flow



$X = ?$
 $S = ?$

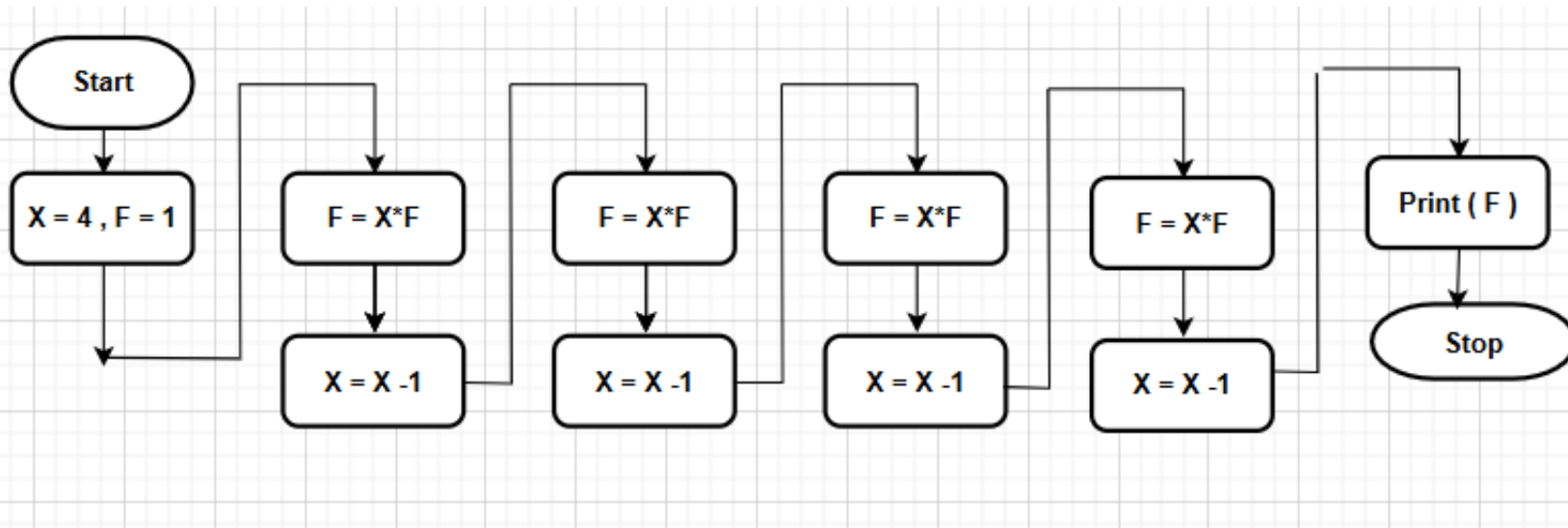
ฝึกอ่าน Sequential Flow



X = ?
F = ?

ถ้า $X = 4$: ต้องเพิ่มอะไร

ฝึกอ่าน Sequential Flow : $X = 4$ & convert to pascal



$X = ?$
 $F = ?$

```
program CalculateNet;  
var  
  
begin  
  
end.
```

1. แมวชื่อมีมี เกิดปี 2019 แต่ปัจจุบันปี 2025 อยากรู้ว่าน้องแมวมีอายุเท่าไร?
myCat **bornYear** **currentYear** **catAge**

myCart มีประเภทข้อมูลคือ String ค่าเริ่มต้น คือ "memi"

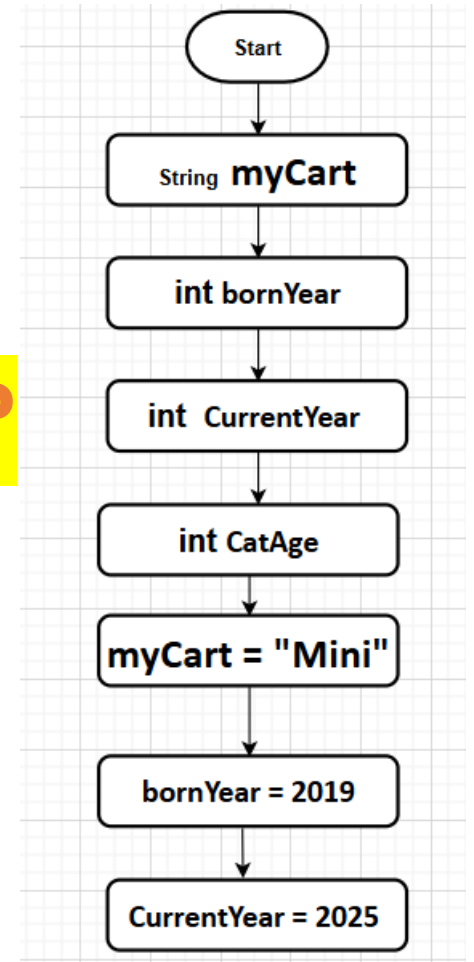
bornYear มีประเภทข้อมูลคือ Int ค่าเริ่มต้น คือ "2019"

CurrentYear มีประเภทข้อมูลคือ Int ค่าเริ่มต้น คือ "2025"

CatAge มีประเภทข้อมูลคือ Int ไม่มีค่าเริ่มต้น ?

Write Flowchart เพื่ออายุของแมว?

```
main.pas
1  program Hello;
2  var myCat :string;
3      bornYear:integer;
4      currentYear:integer;
5      catAge: integer;
6  begin
7      myCat := "Memi";
8      bornYear := 2019;
9      currentYear := 2015;
10 end.
11
12
```



1. แมวชื่อมีมี เกิดปี 2019 แต่ปัจจุบันปี 2025 อยากรู้ว่าน้องแมวมีอายุเท่าไร?
myCat **bornYear** **currentYear** **catAge**

myCart มีประเภทข้อมูลคือ **String** ค่าเริ่มต้น คือ “Memi”

bornYear มีประเภทข้อมูลคือ **Int** ค่าเริ่มต้น คือ “2019”

CurrentYear มีประเภทข้อมูลคือ **Int** ค่าเริ่มต้น คือ “2025”

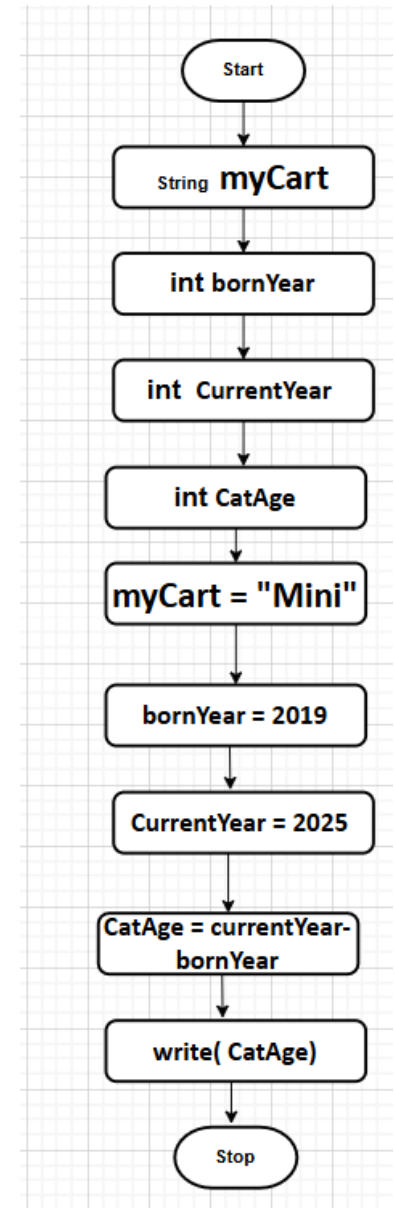
CatAge มีประเภทข้อมูลคือ **Int** ไม่มีค่าเริ่มต้น ?

Convert to Pascal =>

```
program CalculateNet;  
var
```

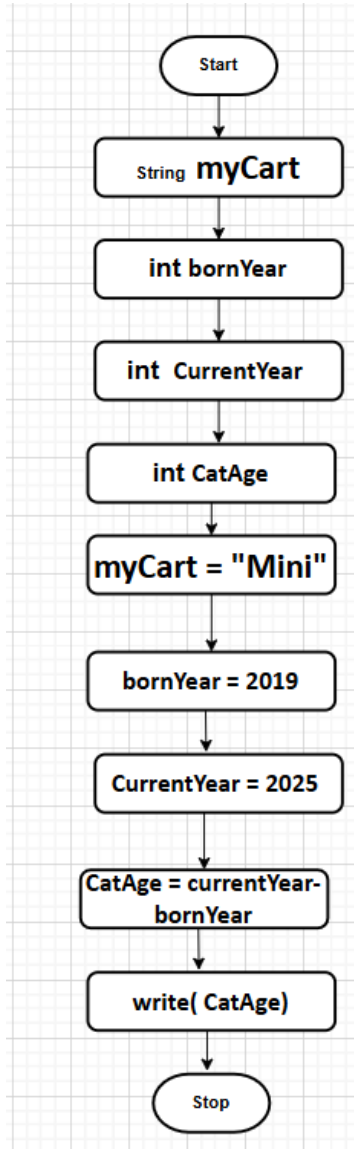
```
begin
```

```
end.
```

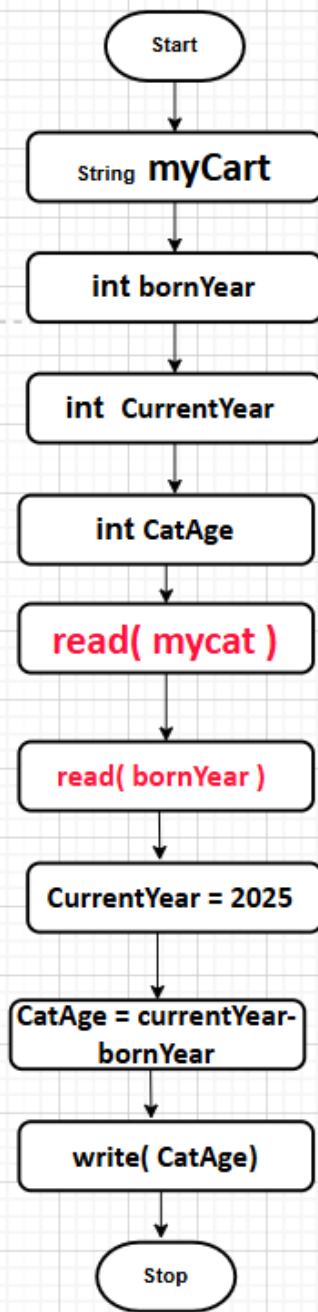
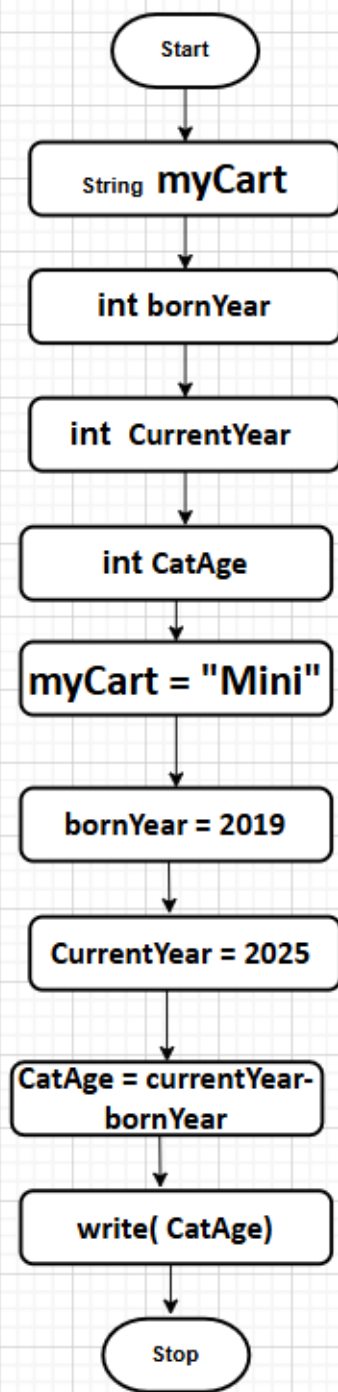


1. แมวชื่อมีมี เกิดปี 2019 แต่ปัจจุบันปี 2025 อยากรู้ว่าน้องแมวมีอายุเท่าไร?
myCat bornYear currentYear catAge

ปรับโจทย์: จงเขียนโปรแกรมรับชื่อแมว ปีที่เกิด แต่ปัจจุบันปี 2025 อยากรู้ว่าน้องแมวอายุเท่าไร



rewrite Flowchart



จงเขียน โปรแกรมหาความยาวของวงกลม โดยรับรัศมีจากผู้ใ้

$$\text{ความยาว} = 2 * \pi * \text{รัศมี}$$

คำถาม

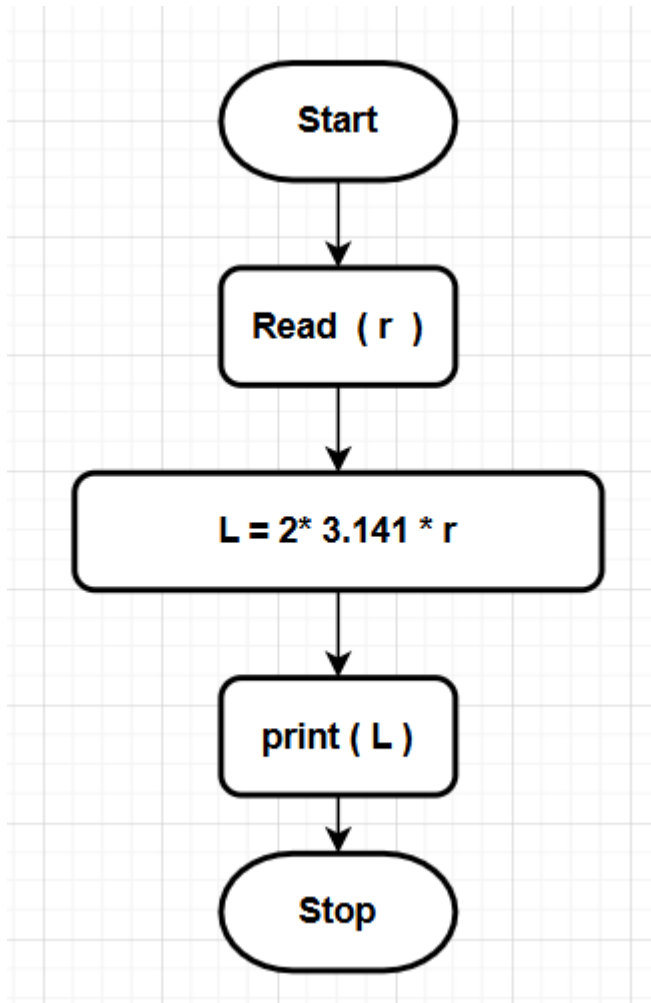
1. ความยาวของวงกลม (L) ไม่ทราบค่า
2. รัศมี (r) รับข้อมูลจากแป้นพิมพ์ => แทนด้วยคำสั่ง `read (r)`
3. สูตร

$$L = 2 * \pi * r$$

π มีค่าคือ 22.0/7.0 หรือประมาณ 3.141

จงเขียน โปรแกรมหาความยาวของวงกลม (L) โดยรับรัศมี (r)จากผู้ใ้

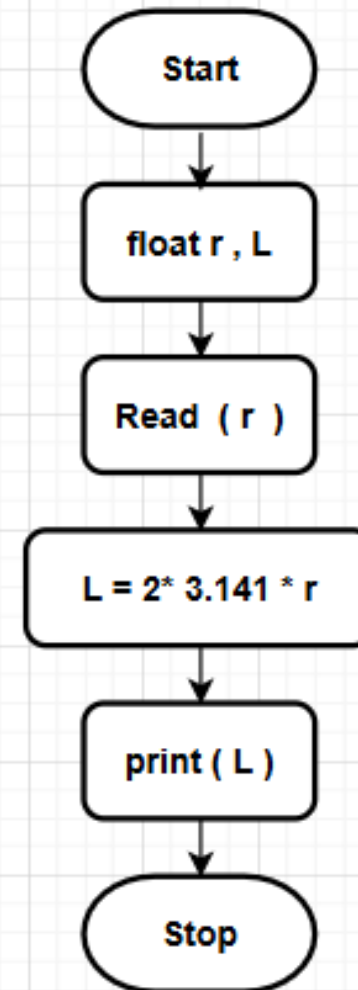
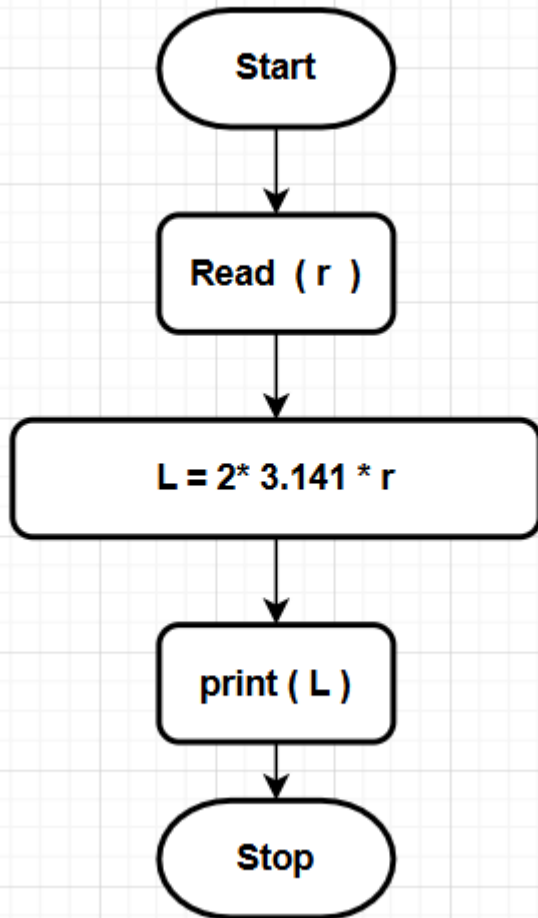
$$\text{ความยาว} = 2 * \pi * \text{รัศมี}$$



จงเขียน โปรแกรมหาความยาวของวงกลม โดยรับรัศมีจากผู้ใ้

$$L = 2\pi r$$

Convert to Pascal



เขียนโปรแกรมคำนวณเงินเดือนสุทธิของพนักงาน โดย

- รับเงินเดือนพื้นฐาน
- หักภาษี 5% ของเงินเดือนพื้นฐาน
- รับค่าคอมมิชชั่นเพิ่มเติม
- แสดงเงินเดือนสุทธิ = (เงินเดือน - ภาษี) + ค่าคอมมิชชั่น

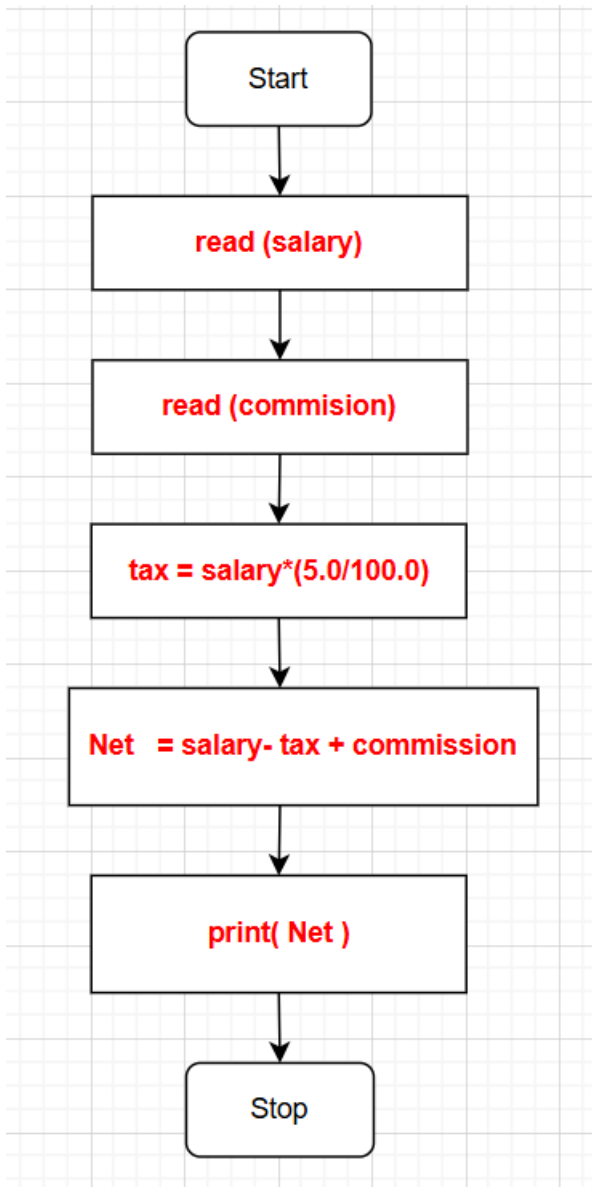
เขียนโปรแกรมคำนวณเงินเดือนสุทธิของพนักงาน โดย

- รับเงินเดือนพื้นฐาน (**salary**)
- หักภาษี 5% ของเงินเดือนพื้นฐาน

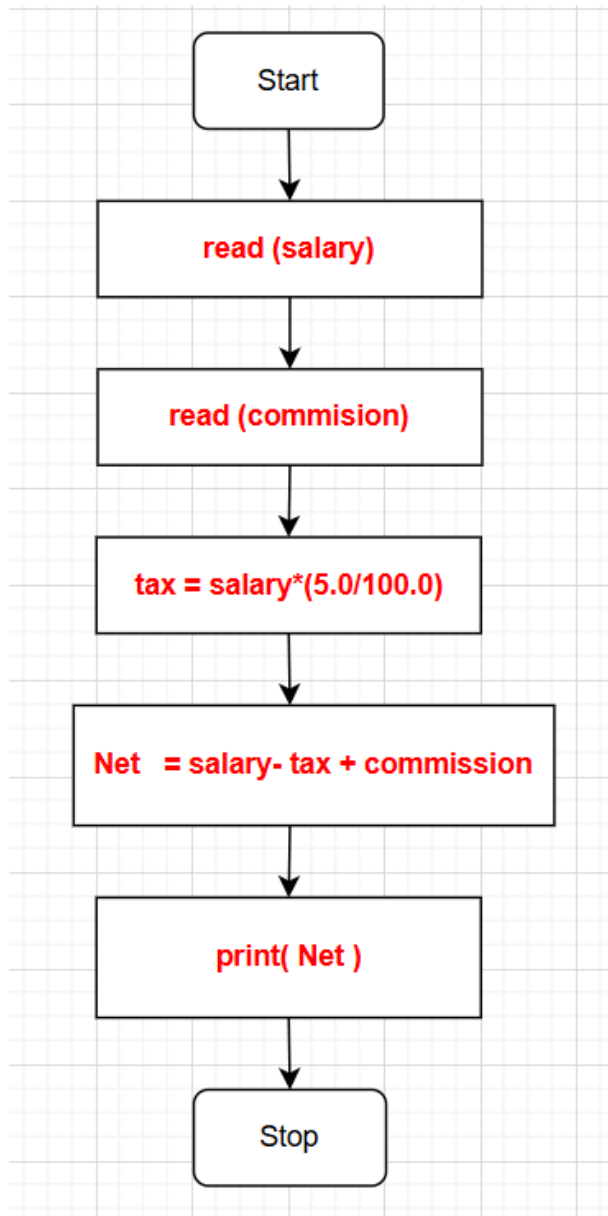
$$\text{tax} = (\text{salary} * 5.0 / 100.0)$$

- รับค่าคอมมิชชั่นเพิ่มเติม (**commission**)
- แสดงเงินเดือนสุทธิ = (เงินเดือน - ภาษี) + ค่าคอมมิชชั่น

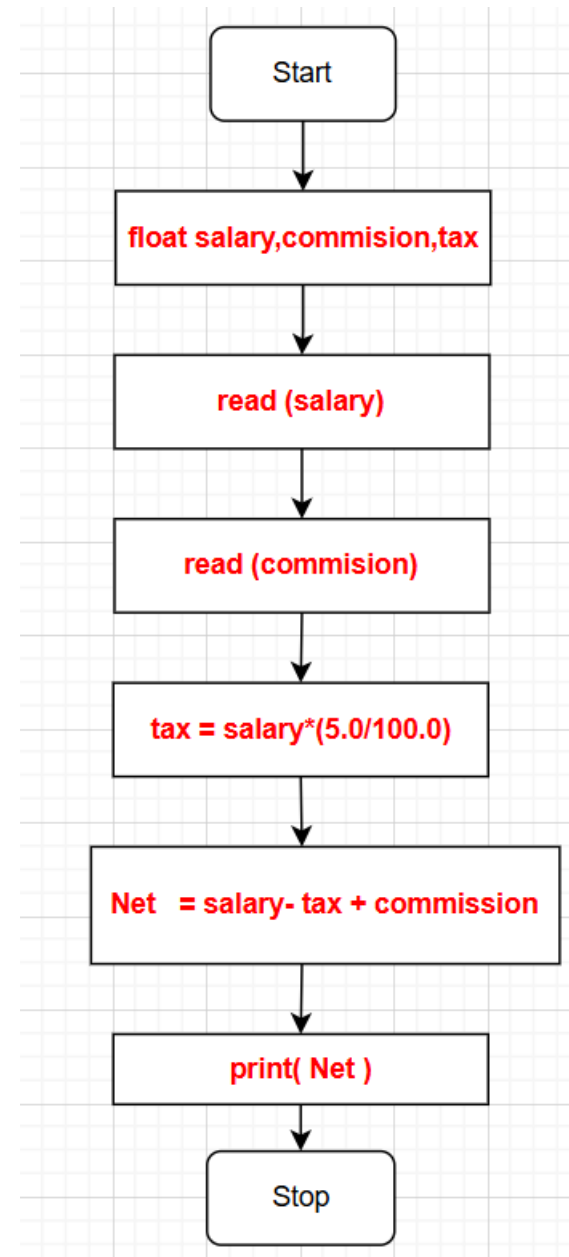
$$\text{Net} = \text{salary} - \text{tax} + \text{commission}$$



เพิ่มประเภทของตัวแปร ?



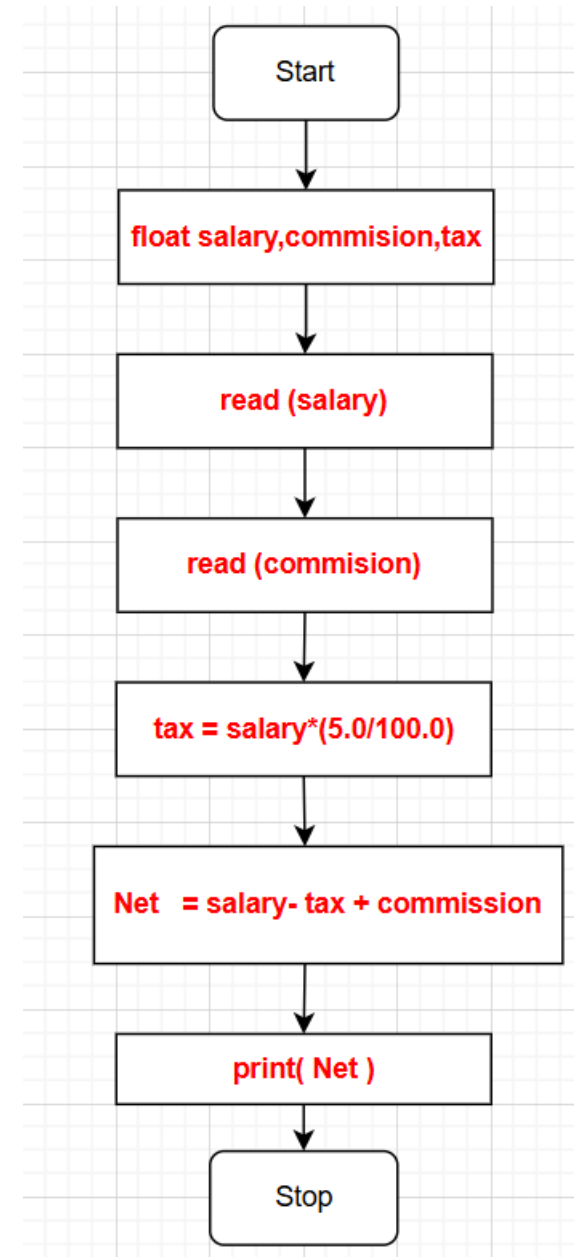
เพิ่ม data type
➔

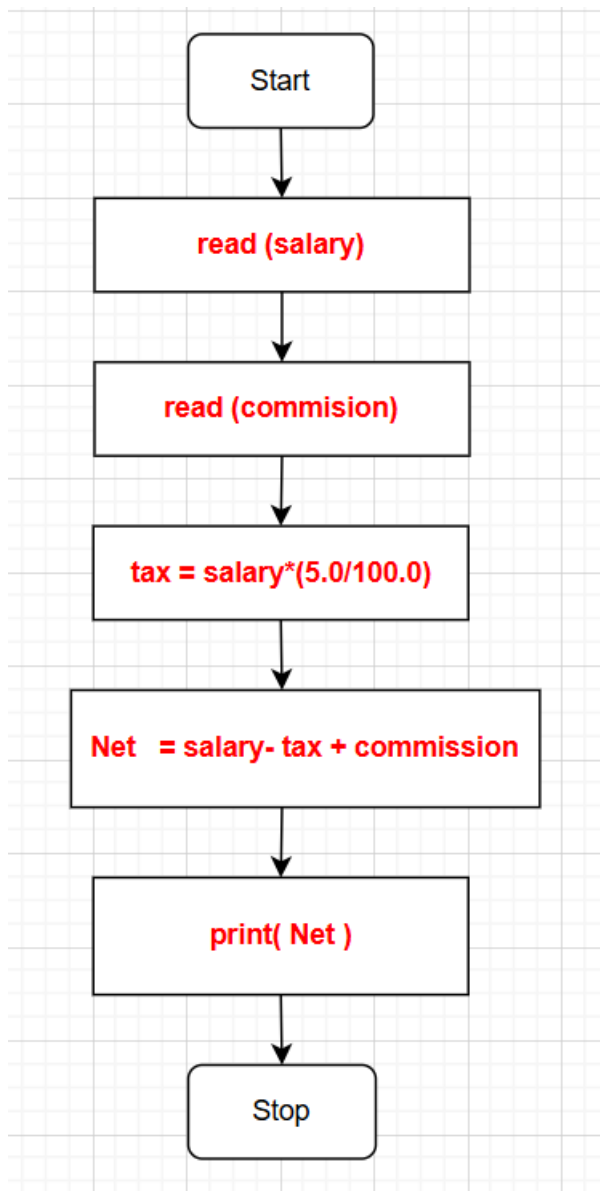


การแปลงให้เป็นภาษาโปรแกรม

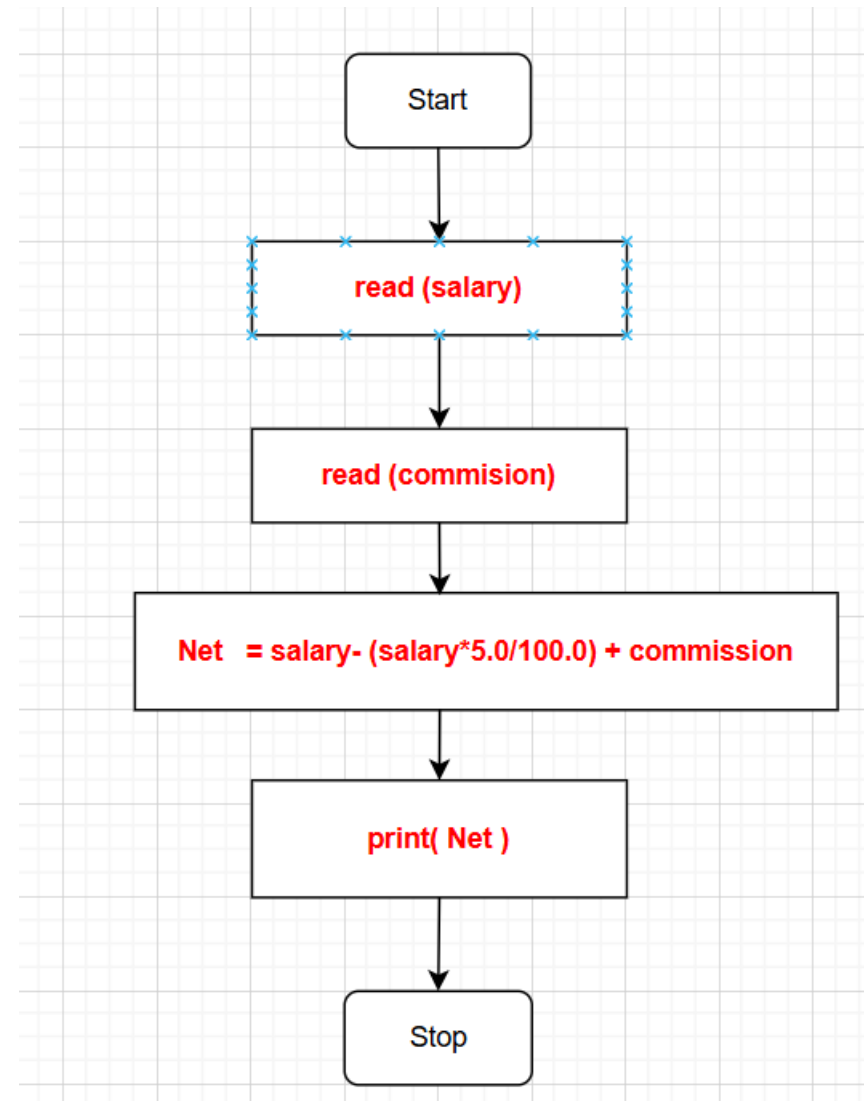
```
main.pas
1 program CalculateNet;
2 var
3   salary, commision, tax, Net: real;
4 begin
5   Write('salary = ');
6   ReadLn(salary);
7
8   Write('commision = ');
9   ReadLn(commision);
10
11   tax := salary * 0.05;
12   Net := salary - tax + commision;
13
14   Write('Net = ');
15   WriteLn(Net:0:2); // แสดงผลทศนิยม 2 ตำแหน่ง
16 end.
17
```

CalculateNet.Pas



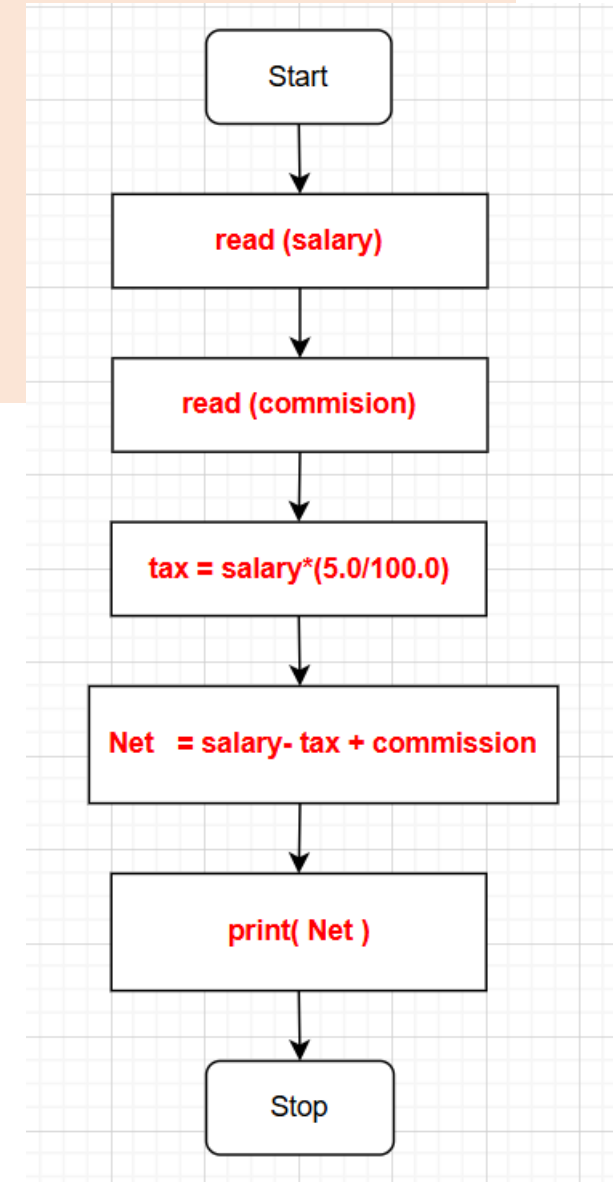


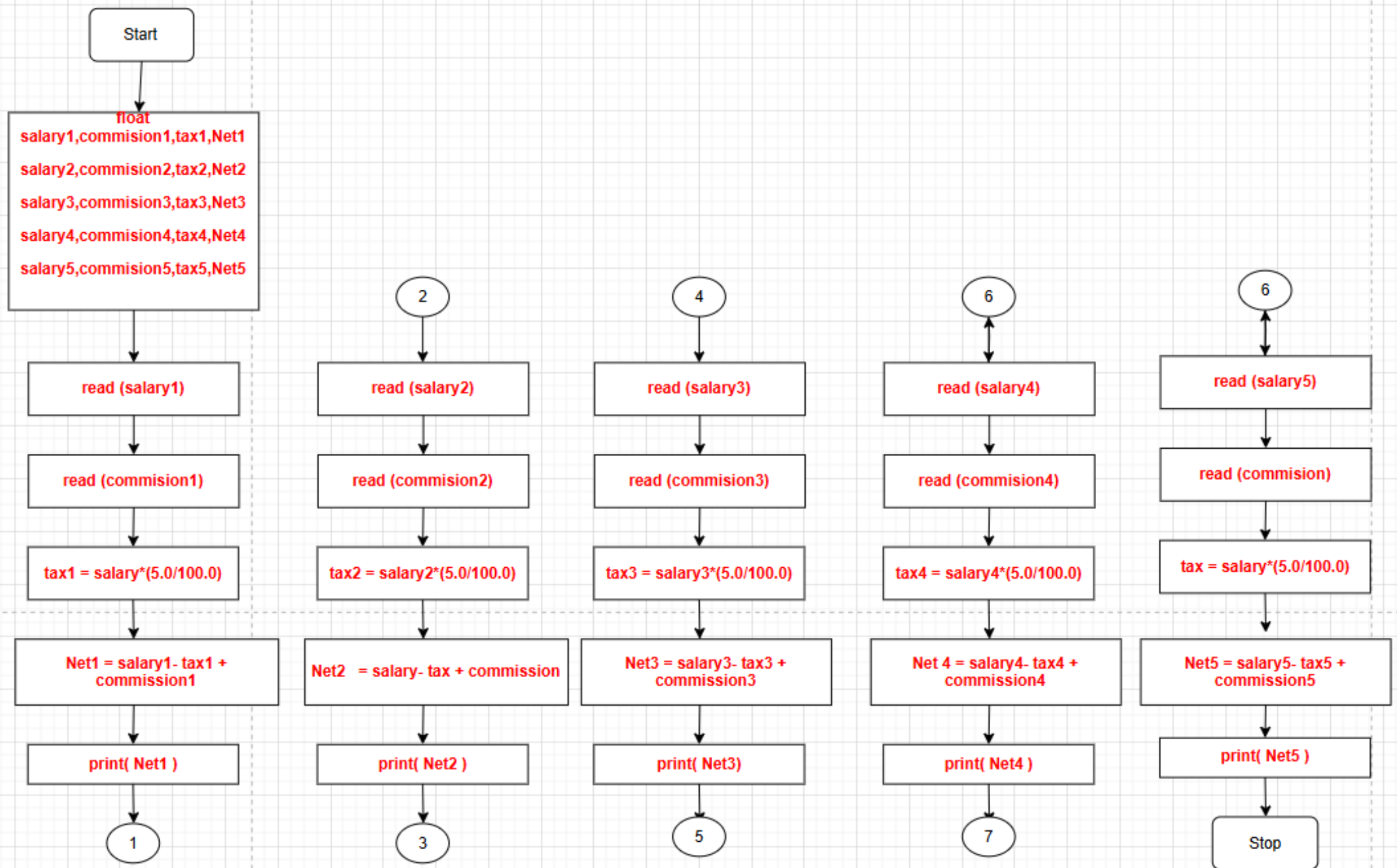
ลดตัวแปร

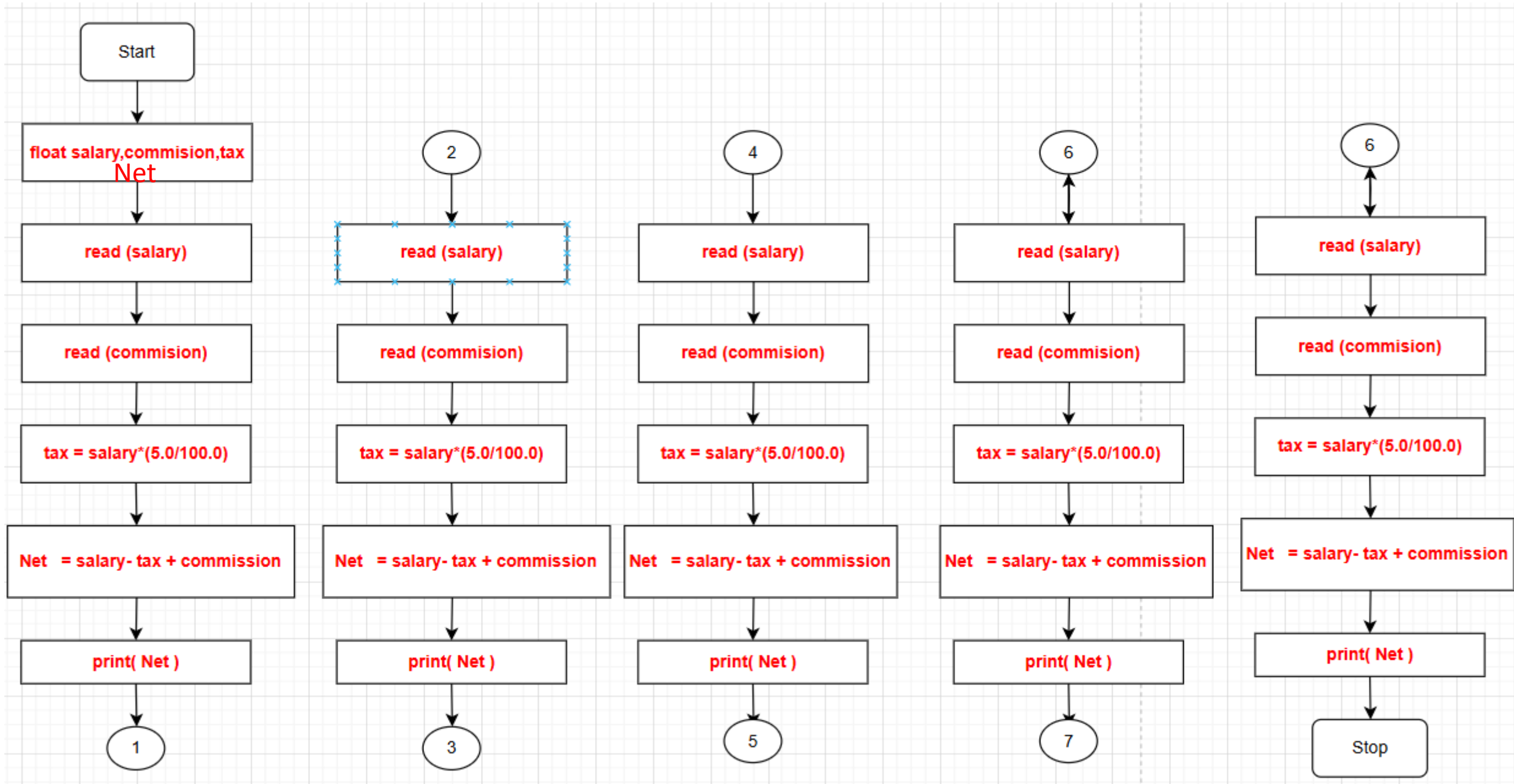


เขียนโปรแกรมรับข้อมูลของพนักงานจำนวน 5 คน เพื่อคำนวณเงินเดือนสุทธิของพนักงาน โดยมีละเอียดข้อมูลดังนี้

- รับเงินเดือนพื้นฐาน (**salary**)
- หักภาษี 5% (**salary* 5.0/100.0**)
- รับค่าคอมมิชชั่นเพิ่มเติม (**commission**)
- แสดงเงินเดือนสุทธิ = (เงินเดือน - ภาษี) + ค่าคอมมิชชั่น
Net = salary- (salary* 5.0/100.0) + commission







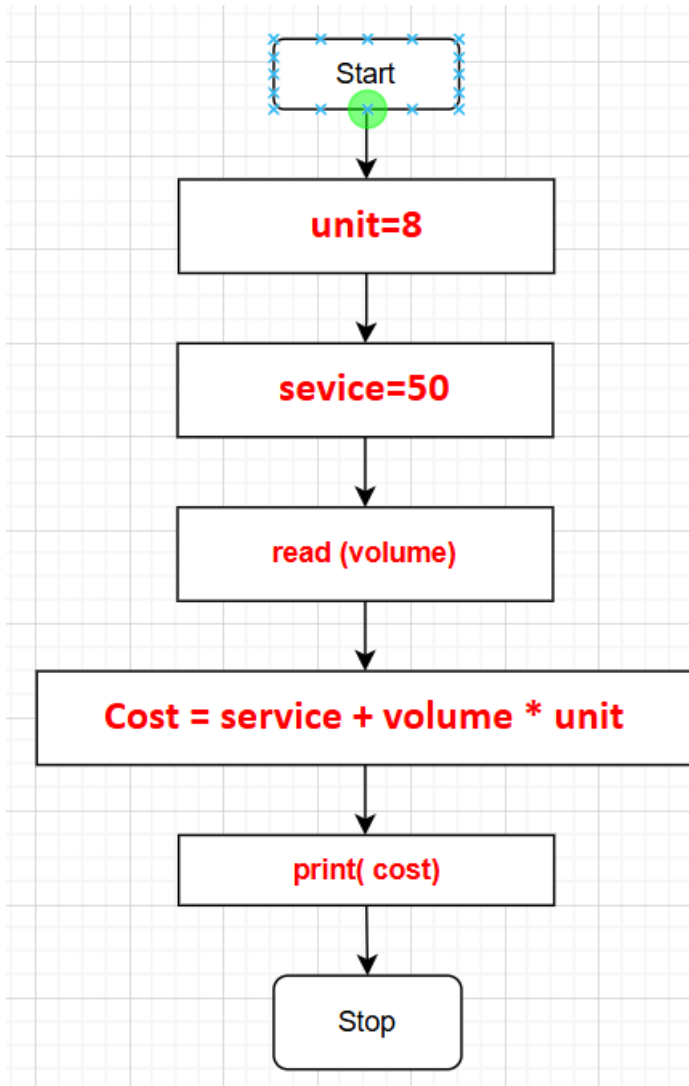
เขียนโปรแกรมเพื่อคำนวณค่าน้ำประปาของบ้านหลังหนึ่ง
โดยกำหนดให้

- ค่าน้ำหน่วยละ 8 บาท
- ค่าบริการพื้นฐาน = 50 บาท
- **รับ** ค่าการใช้น้ำจากผู้ใช้ (เป็นจำนวนหน่วย)
- แสดงผลค่าน้ำรวมทั้งหมด

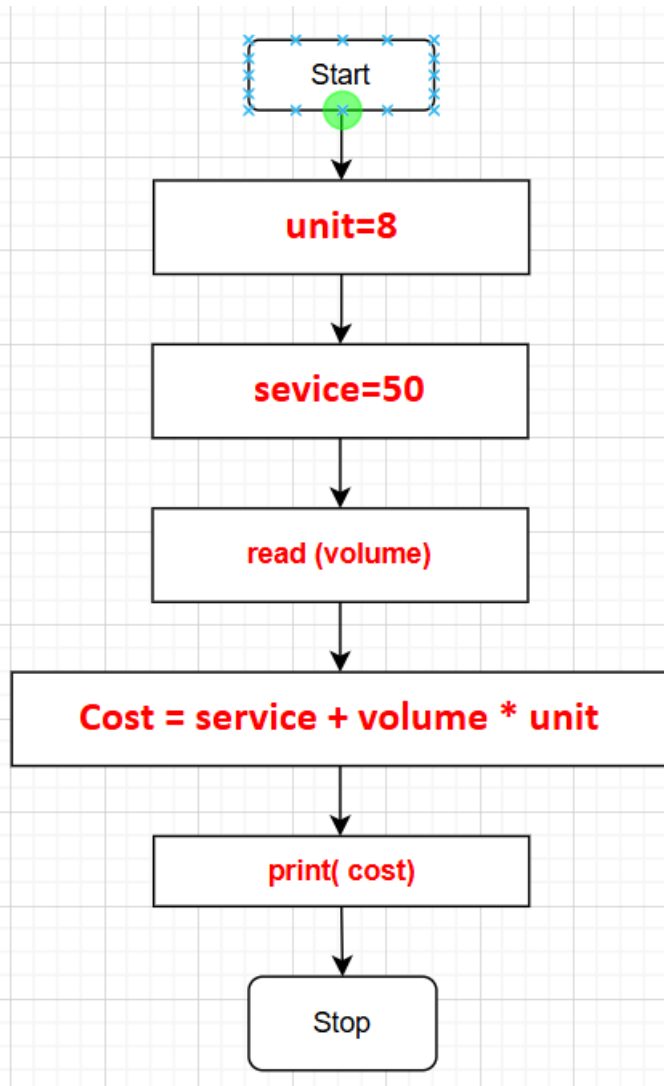
Hit : ใช้น้ำในการหาตัวแปร

เขียนโปรแกรมเพื่อคำนวณค่าน้ำประปาของบ้านหลังหนึ่ง
โดยกำหนดให้

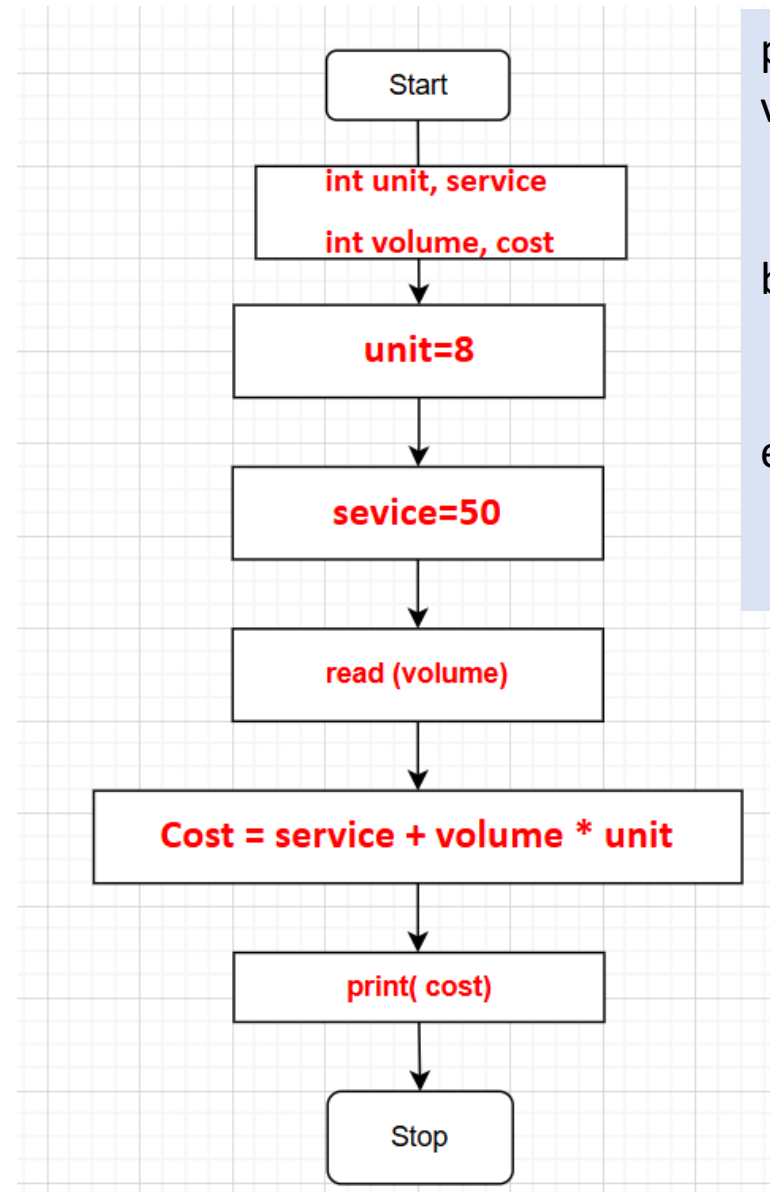
- ค่าน้ำหน่วยละ 8 บาท (**unit=8**)
- ค่าบริการพื้นฐาน = 50 บาท (**service=50**)
- **รับ** ค่าการใช้น้ำจากผู้ใช้ (เป็นจำนวนหน่วย) (**volume**)
- แสดงผลค่าน้ำรวมทั้งหมด **$\text{Cost} = \text{service} + \text{volume} * \text{unit}$**



เพิ่ม data type ?



เพิ่ม data type
➔

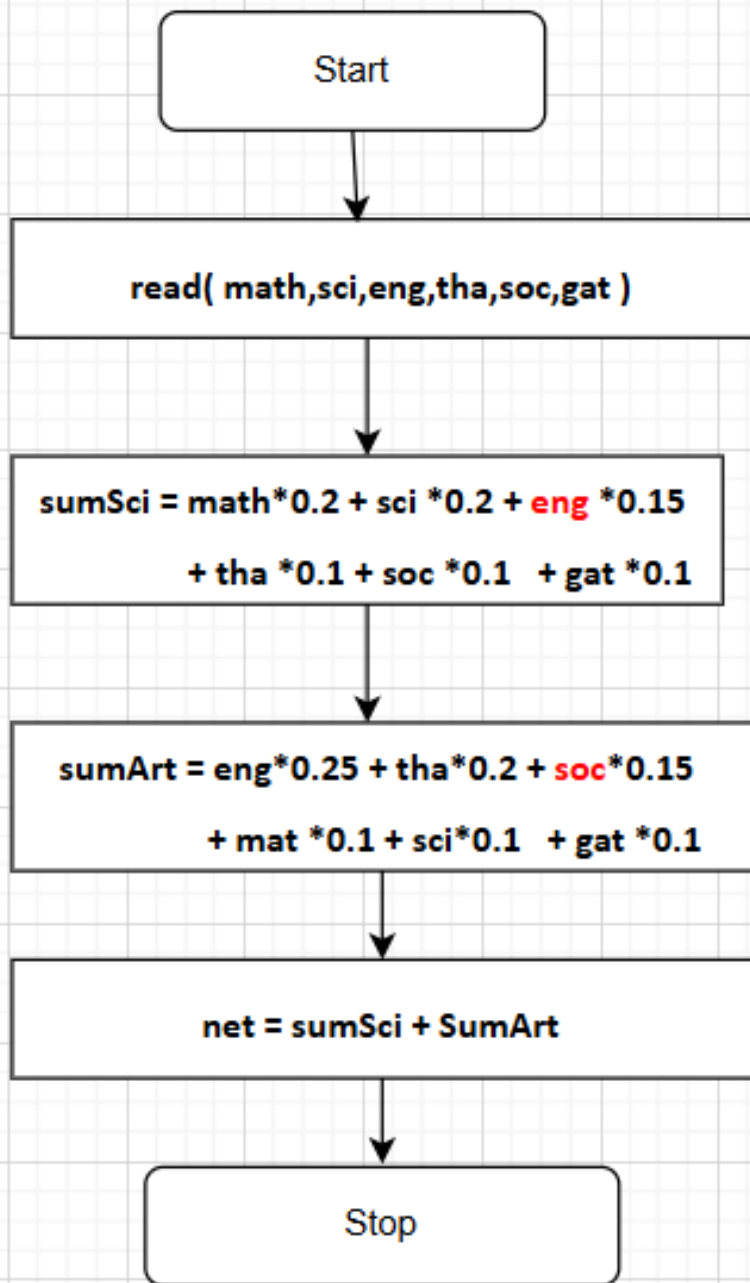


```
program plumbingCost;
var
begin
end.
```

- จงเขียนโปรแกรมคำนวณค่าสอบเข้าเรียนระดับมหาวิทยาลัย รับค่าคะแนนจำนวน 6 วิชาดังนี้
 - คณิตศาสตร์ (เต็ม 100) วิทยาศาสตร์ (เต็ม 100)
 - ภาษาอังกฤษ (เต็ม 100) ภาษาไทย (เต็ม 100)
 - สังคมศึกษา (เต็ม 100) ความถนัดทั่วไป (**GAT**) (เต็ม 150)
- ค่ารวมคะแนนรวมโดยคิดน้ำหนักแบบนี้:
 - กลุ่มสายวิทย์:
 - คณิต (20%), วิทย์ (20%), อังกฤษ (15%), ไทย (10%), สังคม (10%), **GAT (10%)**
 - กลุ่มสายศิลป์:
 - อังกฤษ (25%), ไทย (25%), สังคม (20%), คณิต (10%), วิทย์ (10%), **GAT (10%)**
- สิ่งที่ต้องแสดงผล:
 - คะแนนรวมสายวิทย์
 - คะแนนรวมสายศิลป์
 - คะแนนเฉลี่ยของทั้ง 6 วิชา

- จงเขียนโปรแกรมคำนวณค่าสอบเข้าเรียนระดับมหาวิทยาลัย รับค่าคะแนนจำนวน 6 วิชาดังนี้
 - คณิตศาสตร์ (เต็ม 100) (math) วิทยาศาสตร์ (เต็ม 100) (sci)
 - ภาษาอังกฤษ (เต็ม 100) (eng) ภาษาไทย (เต็ม 100) (thai)
 - สังคมศึกษา (เต็ม 100) (social) ความถนัดทั่วไป (GAT) (เต็ม 150) (gat)
- คำนวณคะแนนรวมโดยคิณน้ำหนักแบบนี้:
 - กลุ่มสายวิทย์: (sumSci)
 - คณิต (20%), วิทย์ (20%), อังกฤษ (15%), ไทย (10%), สังคม (10%), GAT (10%)
 - กลุ่มสายศิลป์: (sumArt)
 - อังกฤษ (25%), ไทย (25%), สังคม (20%), คณิต (10%), วิทย์ (10%), GAT (10%)
- สิ่งที่ต้องแสดงผล:
 - คะแนนรวมสายวิทย์ (sumSci)
 - คะแนนรวมสายศิลป์ (sumArt)
 - คะแนนเฉลี่ยของทั้ง 6 วิชา (net = (SumSci+SumArt) / 6.0)

- จงเขียนโปรแกรมคำนวณค่าสอบเข้าเรียนระดับมหาวิทยาลัย รับค่าคะแนนจำนวน 6 วิชาดังนี้
 - คณิตศาสตร์ (เต็ม 100) (math) วิทยาศาสตร์ (เต็ม 100) (sci)
 - ภาษาอังกฤษ (เต็ม 100) (eng) ภาษาไทย (เต็ม 100) (tha)
 - สังคมศึกษา (เต็ม 100) (soc) ความถนัดทั่วไป (GAT) (เต็ม 150) (gat)
- คำนวณคะแนนรวมโดยคิดน้ำหนักแบบนี้:
 - กลุ่มสายวิทย์: (sumSci)
 - คณิต (20%), วิทย์ (20%), อังกฤษ (15%), ไทย (10%), สังคม (10%), GAT (10%)
 - $$\text{sumSci} = \text{math} * 0.2 + \text{sci} * 0.2 + \text{eng} * 0.15 + \text{tha} * 0.1 + \text{soc} * 0.1 + \text{gat} * 0.1$$
 - กลุ่มสายศิลป์: (sumArt)
 - อังกฤษ (25%) , ไทย (25%), สังคม (20%), คณิต (10%), วิทย์ (10%), GAT (10%)
 - $$\text{sumArt} = \text{eng} * 0.25 + \text{tha} * 0.25 + \text{soc} * 0.2 + \text{mat} * 0.1 + \text{sci} * 0.1 + \text{gat} * 0.1$$
- สิ่งที่ต้องแสดงผล:
 - คะแนนรวมสายวิทย์ (sumSci)
 - คะแนนรวมสายศิลป์ (sumArt)
 - คะแนนเฉลี่ยของทั้ง 6 วิชา ($\text{net} = (\text{SumSci} + \text{SumArt}) / 6.0$)

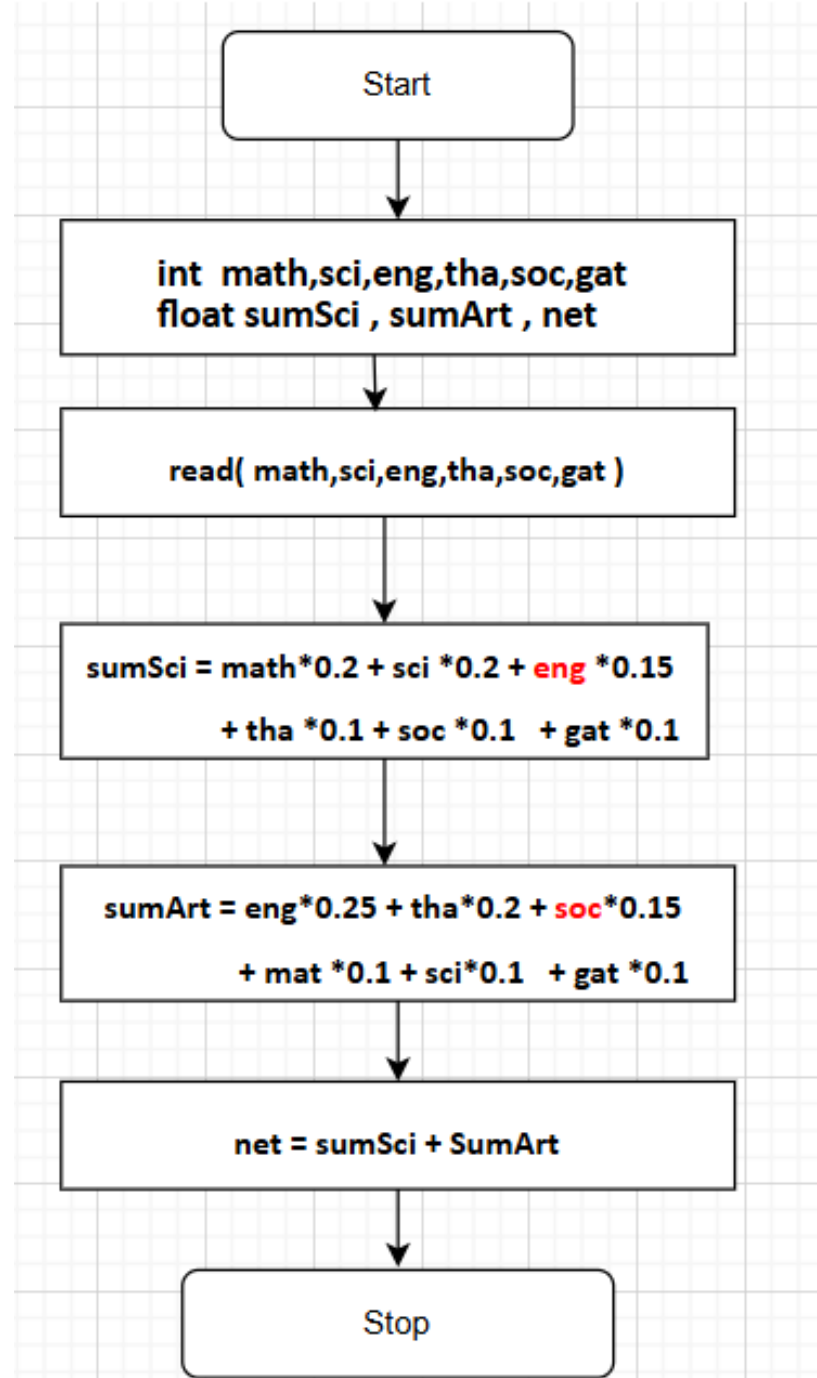


เพิ่ม data type



Convert flow to Pascal

```
program sumScore;  
var  
  
begin  
  
end.
```



คำสั่ง Modulo (%) : ใช้กับข้อมูล integer และมีค่าเป็นบวก

1 mod 3 -> 1

2 mod 3 -> 2

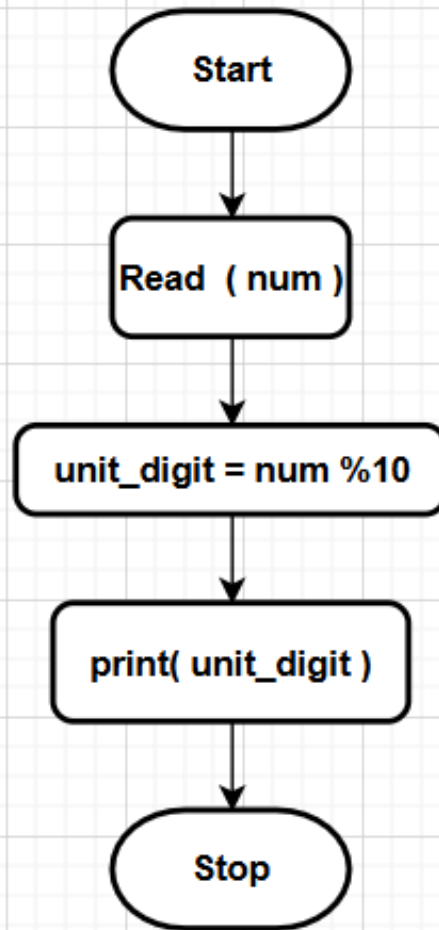
3 mod 3 -> 0

4 mod 3 -> 1

5 mod 3 -> 2

6 mod 3 -> 0

7 mod 1 -> 1



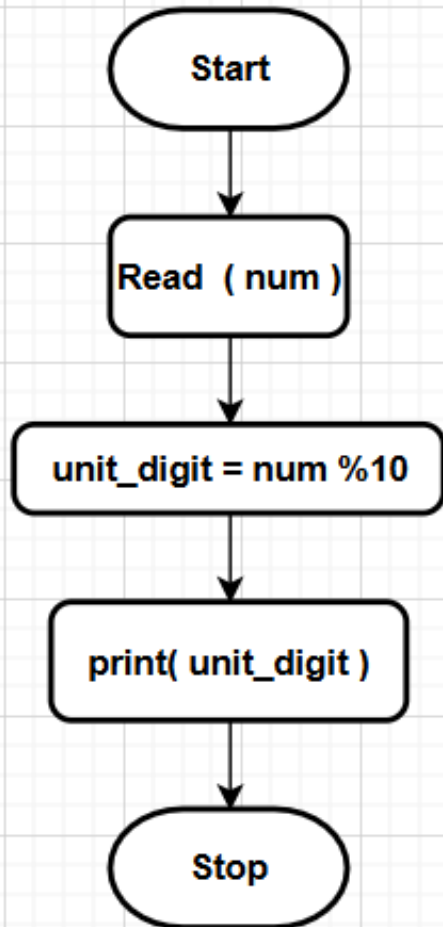
% \Leftrightarrow mod

modExample.pas

คำสั่ง Modulo (%) : ใช้กับข้อมูล integer และมีค่าเป็นบวก

% \Leftrightarrow mod

modExample.pas



main.pas

```
1  program modExample;
2
3  var
4      num, unit_digit: integer;
5
6  begin
7      Write('num = ');
8      ReadLn(num);
9
10     unit_digit := num mod 10;
11
12     WriteLn('unit_digit = ', unit_digit);
13 end.
14
15
```