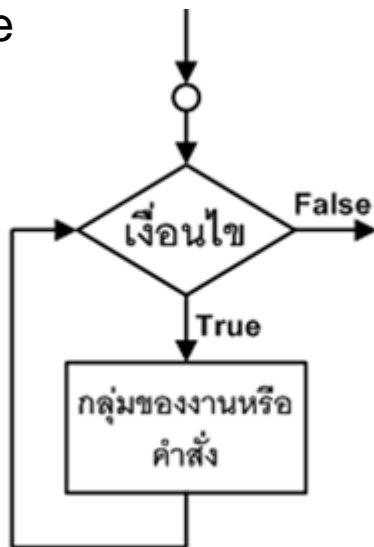


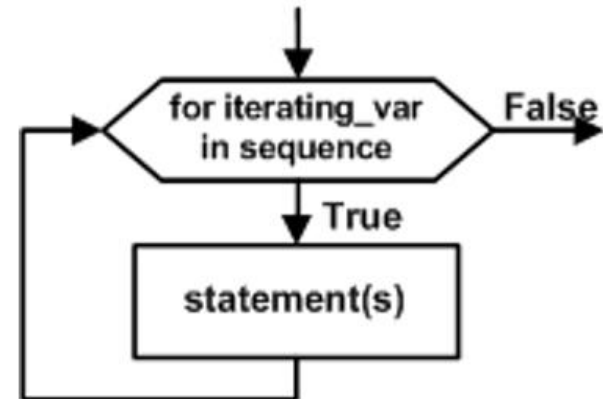
# คำสั่งวนซ้ำ (Loop)

While , For

while



for



# 1. Loop : While

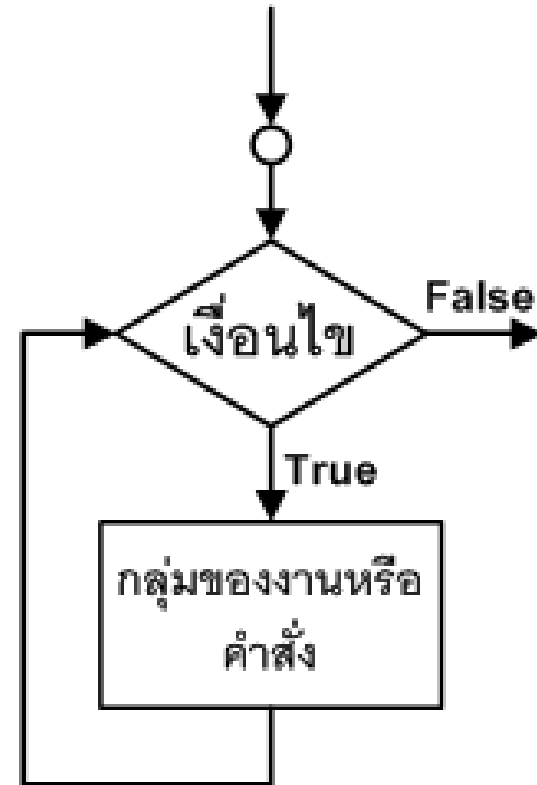
งานที่ต้องทำซ้ำเป็นจำนวนรอบ ใช้ได้ทั้งการวนรอบที่มีจำนวนรอบที่แน่นอนและไม่แน่นอน

มีเงื่อนไขในการหยุดการทำงาน

ตรวจเงื่อนไขก่อนการทำงานทุกครั้ง  
ถ้าใช้ตามที่เงื่อนไขต้องการ จะทำงานซ้ำต่อไป

อาจไม่ได้ทำเลยแม้แต่ครั้งเดียว

Flow diagram



# 1. Loop : While

คำสั่ง **while** จะทดสอบเงื่อนไขก่อนทำงาน จะวนรอบทำงานเมื่อเงื่อนไขเป็นจริง

รูปแบบ

```
while condition :  
    statement(s)
```

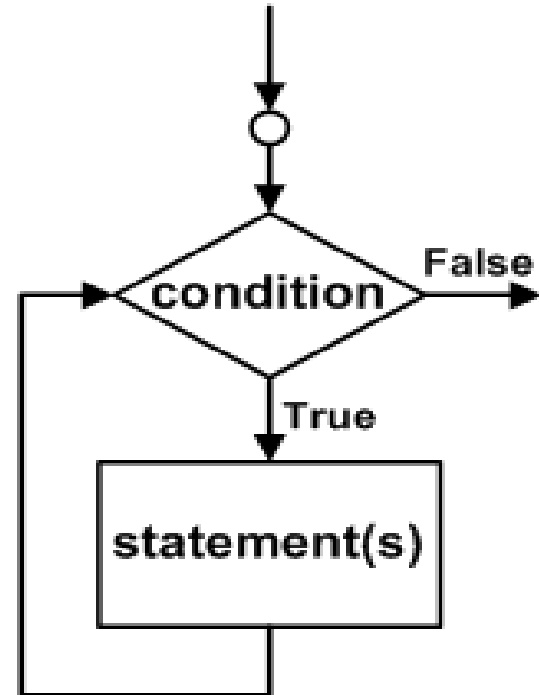
อธิบาย

**Condition** คือ เงื่อนไขในการตัดสินใจว่า  
ต้องการให้ทำงานหรือไม่

ถ้าเงื่อนไขเป็นจริง (**True**) จึงจะทำงาน

**statement** คือคำสั่งที่ต้องการให้ทำงานในกรณีที่ เงื่อนไขเป็นจริง

**Flow diagram**



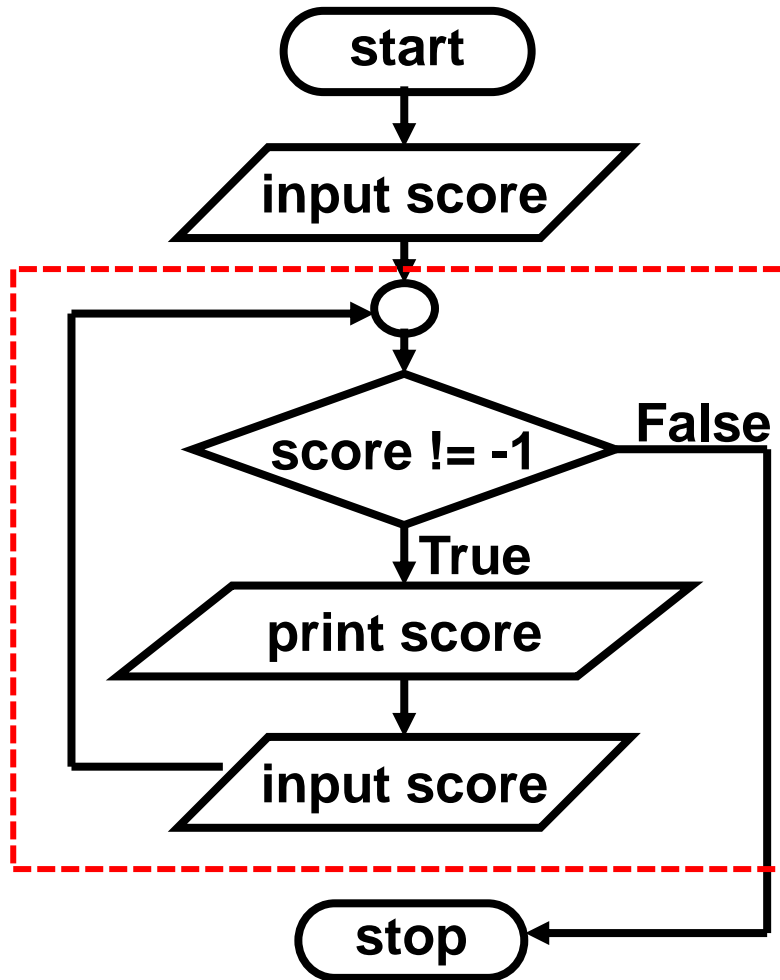
# ตัวอย่าง 1 รับและแสดงคะแนน

ให้รับคะแนนของนักศึกษาในกลุ่มหนึ่งแล้วทำการแสดงผล  
โดยกำหนดให้การป้อนคะแนนเป็น -1 คือสิ้นสุดการรับคะแนน  
(ให้ใช้คำสั่ง while)

## วิเคราะห์โจทย์

ผลลัพธ์	คือ แสดงคะแนนของนักศึกษาในกลุ่มหนึ่ง
ข้อมูลเข้า	คือ คะแนนของนักศึกษาในกลุ่มหนึ่ง (score)
การประมวลผล	จำนวนรอบไม่แน่นอน ผู้ใช้ข้อมูล -1 สิ้นสุดการรับ เป็นเงื่อนไขในการหยุด

# ตัวอย่าง 1 รับและแสดงคะแนน(ต่อ)



เริ่มต้นทำงาน

รับค่าคะแนนจากผู้ใช้

ถ้าคะแนนที่รับมาไม่เท่ากับ -1 ให้แสดงผลคะแนนนั้น แล้วเริ่มต้นรับคะแนนใหม่

สิ้นสุดทำงาน

# ตัวอย่าง 1 รับและแสดงคะแนน(ต่อ)

เขียนโปรแกรมภาษาไพทอน `score.py` ได้ดังนี้

```
inp = input("score :")  
score = float(inp)  
while score != -1 :  
    print(score)  
  
inp = input("score :")  
score = float(inp)
```

ตัวอย่างการทำงานและผลลัพธ์ที่ได้

```
>>>  
score :45  
45.0  
score :40  
40.0  
score :39.2  
39.2  
score :37  
37.0  
score :-1  
>>> |
```

## ตัวอย่าง 2

ให้รับค่าเลขจำนวนจริงค่าหนึ่งแล้วทำการแสดงผลของค่ากำลังสองของ  
เลขนั้น โดยกำหนดให้ถ้ามีการป้อนค่าลบให้สิ้นสุดการรับข้อมูล  
(กำหนดให้ใช้ คำสั่ง while )

### วิเคราะห์โจทย์

**ผลลัพธ์**

คือ ค่ากำลังสองของตัวเลขที่รับเข้ามา

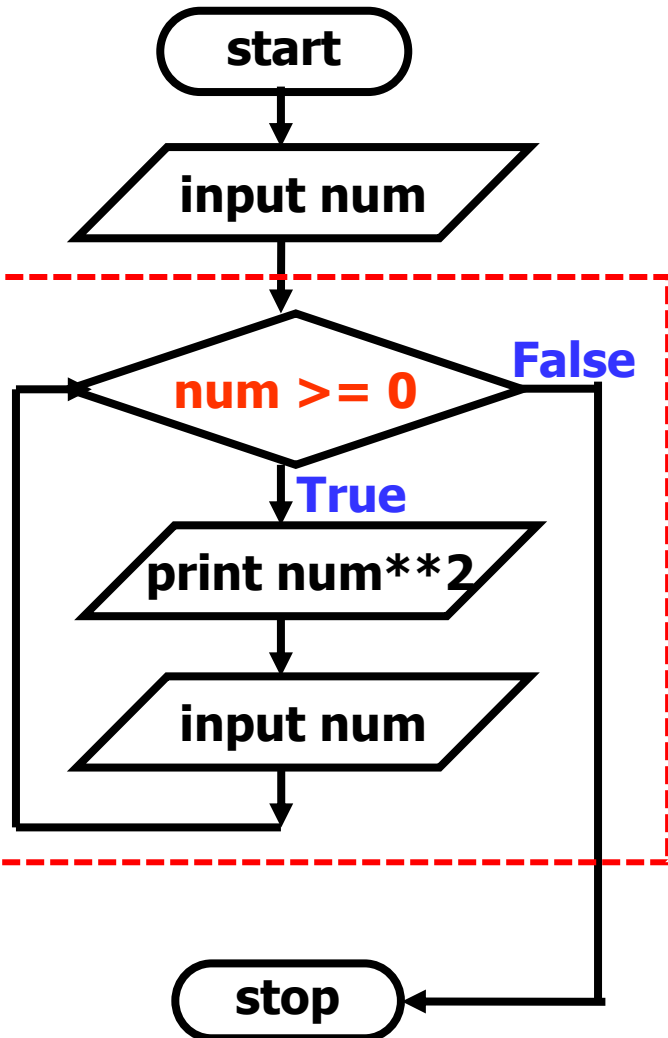
**ข้อมูลเข้า**

คือ เลขจำนวนจริง (num)

**การประมวลผล** จำนวนรอบไม่แน่นอน

ป้อนค่าข้อมูลเป็นเลข**ลบ** เงื่อนไขในการ**หยุด**วนทำซ้ำ

## ตัวอย่าง 2 (ต่อ)



เขียนโปรแกรมภาษาไพทอน power.py

```
inp=input('Input float number : ')
num = float(inp)
while num >= 0 :
    print('Power of num=', num**2)
    inp=input('Input float number : ')
    num = float(inp)
```

ผลลัพธ์ที่ได้

```
>>>
Input float number : 2.5
Power of num= 6.25
Input float number : 5.5
Power of num= 30.25
Input float number : -1
>>> |
```



# Infinite Loop

**ข้อควรระวัง!!** เราอาจเขียนโปรแกรมแล้วเกิด วนซ้ำไม่จบสิ้น (infinite loop) ดังตัวอย่าง

```
var = 1
while var == 1 :      # This constructs an infinite loop
    num = input("Enter a number :")
    print ("You entered: ", num)
print ("Good bye!")
```

เมื่อโปรแกรมทำงาน สมมติเราใส่ข้อมูลดังตัวอย่าง

Enter a number :20

You entered: 20

Enter a number :29

You entered: 29

Enter a number :3

You entered: 3

จากตัวอย่างข้างต้น โปรแกรมจะวนทำงานไม่จบสิ้น เราต้อง กด **CTRL+C** เพื่อจบการทำงาน  
การทำงานของโปรแกรม : Keyboard Interrupt

## 2. Loop : For

ปกติภาษาที่เราคุ้นเคย เมื่อทำงาน Loop ด้วยคำสั่ง For มักจะใช้ตัวเลขเป็นตัวนับการวนซ้ำ คำสั่ง For ในภาษาไพทอน จะใช้ ลำดับ(sequence) เป็นตัววนซ้ำ ตัวอย่าง เป็นการเปรียบเทียบการเขียนโปรแกรมเพื่อ loop โดยใช้

คำสั่ง while และคำสั่ง for

```
counter = 0
while counter <= 10 :
    print (counter)
    counter +=1
```

ผลลัพธ์

```
0
1
2
3
4
5
6
7
8
9
10
```

```
for counter in range (10):
    print (counter)
```

ผลลัพธ์

```
0
1
2
3
4
5
6
7
8
9
```

## 2. Loop : For

คำสั่ง `for` ในภาษาไพทอนจะวนซ้ำเพื่อทำงาน โดยใช้ลำดับ (sequence) เช่น ลำดับที่เราระบุโดยตรง , ลิสต์ , สตริง , `range()`

- รูปแบบ `for... in ...` :

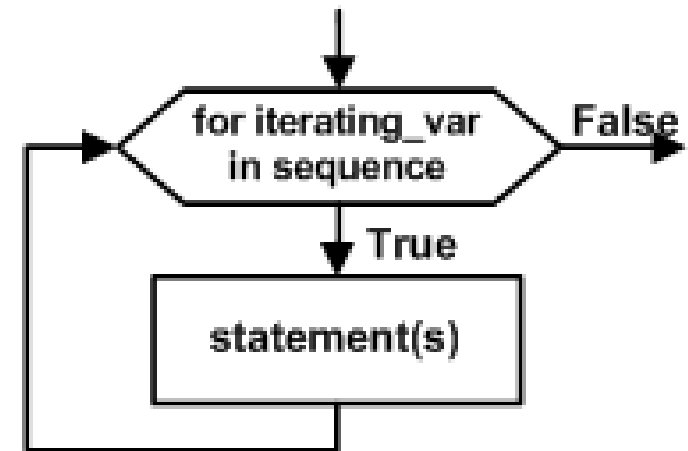
```
for iterating_var in sequence :  
    statements(s)
```

อธิบาย `iterating_var` คือ ตัวแปรที่ใช้วนซ้ำ

`sequence` คือ ลำดับ

`statement(s)` คือคำสั่งในขอบเขตของคำสั่ง `for` จะวนรอบทำซ้ำ(True) สำหรับ next item from sequence และออกจากการวนรอบ(False) เมื่อ no more item in sequence

Flow diagram



## 2. Loop : For

ใช้ลำดับ (sequence) : ลำดับที่เราระบุโดยตรง, ลิสต์, สตริง, range()

(1) คำสั่ง for วนซ้ำทำงาน ลำดับที่ใช้คือ : กำหนดลำดับโดยตรง

ตัวอย่าง

```
for i in 1,2,3,4,5 :  
    print(i)
```

ผลลัพธ์  
1  
2  
3  
4  
5

```
for i in 'one','two','three':  
    print(i)
```

ผลลัพธ์  
one  
two  
three

## 2. Loop : For

ใช้ลำดับ (sequence) : ลำดับที่เราระบุโดยตรง, ลิสต์, สตริง, range()

(2) คำสั่ง for วนซ้ำทำงาน ลำดับที่ใช้คือ : สตริงหรือข้อความ

```
for letter in 'Python' :  
    print (letter)
```

```
ผลลัพธ์  
P  
y  
t  
h  
o  
n
```

(3) คำสั่ง for วนซ้ำทำงาน ลำดับที่ใช้คือ : ลิสต์

```
fruits = ['banana', 'apple', 'mango']
```

```
for word in fruits :  
    print(word)
```

```
ผลลัพธ์  
banana  
apple  
mango
```

```
a = ['cat', 'window', 'defenestrate']  
for x in a :  
    print (x)
```

```
ผลลัพธ์  
cat  
window  
defenestrate
```

## 2. Loop : For

ใช้ลำดับ (sequence) : ลำดับที่เราระบุโดยตรง, ลิสต์, สตริง, range()

(4) คำสั่ง for วนซ้ำทำงาน ลำดับที่ใช้คือ : range()

```
for i in range(10) :  
    print (i)
```

ผลลัพธ์

0  
1  
2  
3  
4  
5  
6  
7  
8  
9

การวนซ้ำ คำสั่ง for ดังตัวอย่างข้างต้น

มีการเรียกใช้ ฟังก์ชัน range() อธิบายฟังก์ชัน range() ได้ดังนี้ →

## 2. Loop : For

ใช้ลำดับ (sequence) : ลำดับที่เราระบุโดยตรง, ลิสต์, สตริง, range()

- รูปแบบ `range([start,] stop [,step])`      argument เป็นเลขจำนวนเต็ม

ฟังก์ชัน `range()` มีการส่งหรือ pass parameter ให้ฟังก์ชัน `range` ได้ 3 argument

- **pass 1 arg.** → called **stop**      ในกรณีนี้ฟังก์ชัน `range` จะ return sequence ในขอบเขต ตั้งแต่ 0 – (end-1)

- **pass >=2 arg.** ตัวแรกเรียก start ตัวถัดมาเรียก stop ตัวสุดท้ายเรียก step

ลำดับจะถูกสร้าง ด้วยการเพิ่มค่าจาก start ไป end โดยเพิ่มขึ้นด้วยค่าของ step

(ถ้า step เป็นบวก ค่าสุดท้ายใน sequence คือ largest multiple less than end)

ตัวอย่าง

สรุป รูปแบบการใช้ function `range()` มี 3 รูปแบบ คือ

1 arg → `range(10)`

2 arg → `range(0,10)`

3 arg → `range(0,10,1)`

`range(10)` `range(0,10)` `range(0,10,1)` ได้ผลลัพธ์เหมือนกันคือตัวเลขตั้งแต่ 0 ถึง 9

## 2. Loop : For

รูปแบบ `range([start,] stop [,step])`

ตัวอย่าง คำสั่ง for และการเรียกใช้ ฟังก์ชัน range()

```
for counter in range (1,101) :
```

```
    # vary control variable from 1 to 100 increments of 1
```

```
for counter in range (100, 0 ,-1) :
```

```
    # vary control variable from 100 to 1 increments of -1
```

```
for counter in range (7,78,7) :
```

```
    # vary control variable from 7 to 77 in steps of 7
```

```
for counter in range (2,21,3) :
```

```
    # vary control variable over following sequence of value 2 5 8 11 14 17 20
```

```
for counter in range (1,10,4) :
```

```
    # vary control variable over following sequence of value 1,5,9
```

```
for counter in range (99,-1,-11) :
```

```
    # vary control variable over following sequence of value 99 88 77 66 55 44 33 22 11 0
```



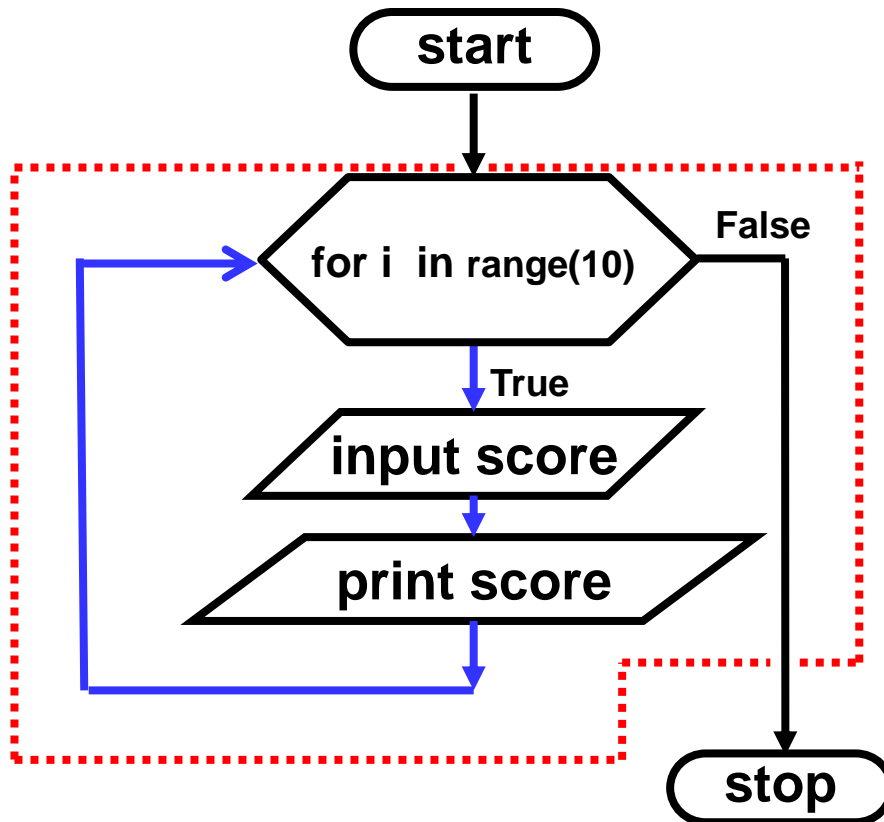
# ตัวอย่าง 1

ให้รับคะแนนของนักศึกษา 10 คนแล้วทำการแสดงผล (ใช้คำสั่ง for)

วิเคราะห์โจทย์

10 คน

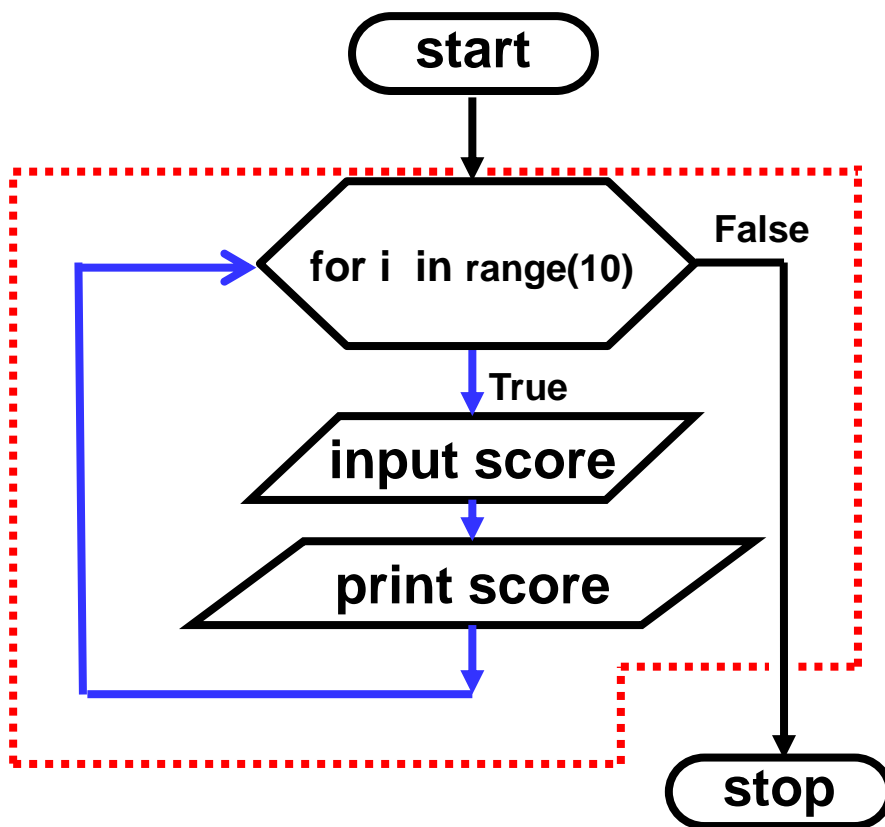
จำนวนรอบแน่นอน



- กำหนดขอบเขตการวนซ้ำ
- ในแต่ละรอบที่ทำงานตัวแปร  $i$  จะเพิ่มขึ้นทีละหนึ่ง
- รวมทำงาน 10 รอบ

# ตัวอย่าง 1

เขียนโปรแกรมภาษาไพทอน ScoreFor.py ได้ดังนี้



```
for i in range(10) :  
    inp=input("score :")  
    score=float(inp)  
    print(score)
```

## ตัวอย่าง 2

ให้รับคะแนนของนักศึกษา  $n$  คนแล้วทำการคำนวณและแสดงผลคะแนนเฉลี่ย โดยที่ค่า  $n$  ให้รับจากผู้ใช้ (For)

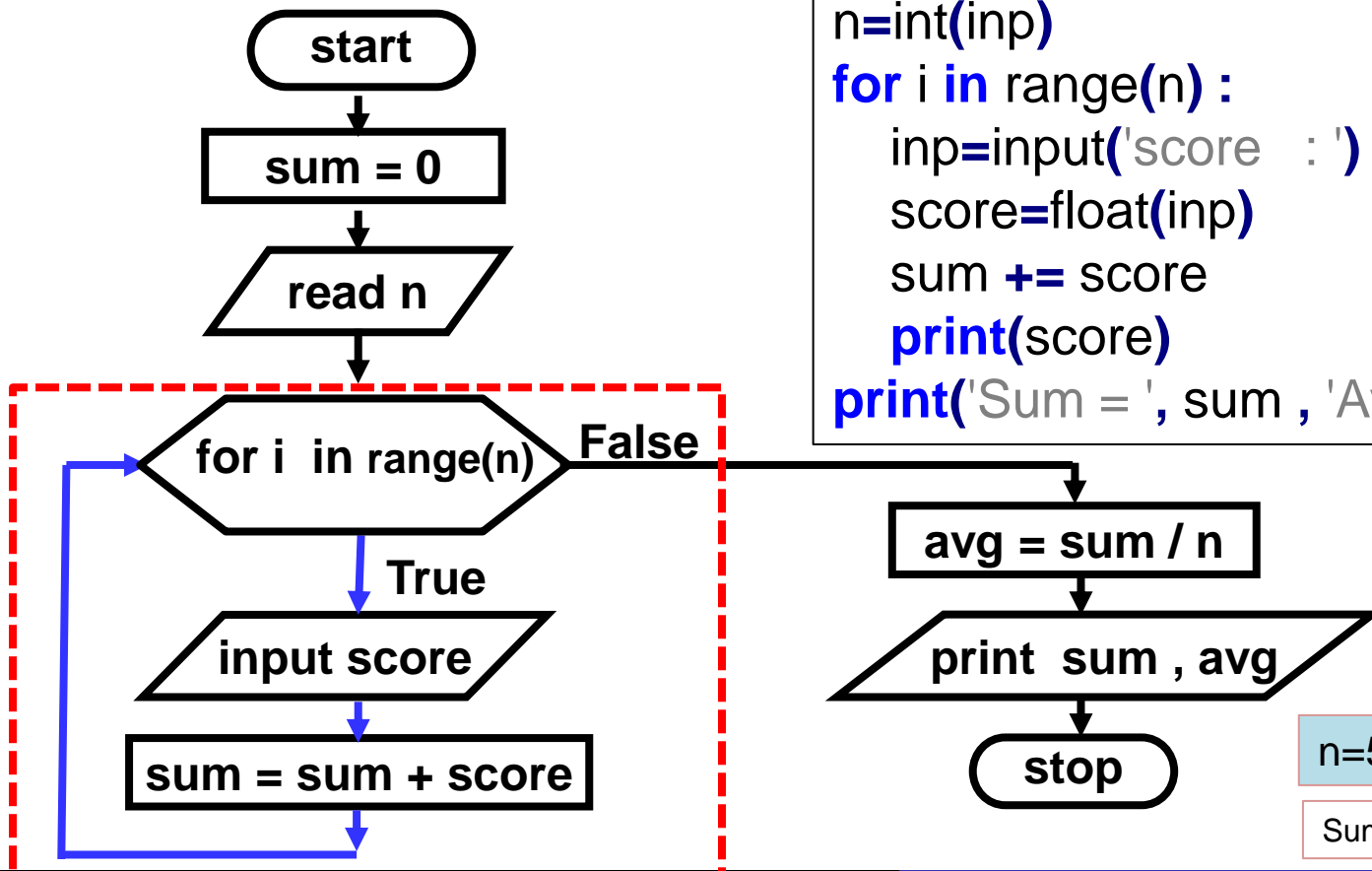
### วิเคราะห์โจทย์

- $n$  คน จำนวนรอบแน่นอน
- กำหนดตัวแปร sum ไว้สำหรับเก็บค่าผลรวม (ตัวแปรที่เก็บค่าผลรวมก่อนใช้งานต้องให้ค่าเริ่มต้นเป็น 0 ก่อน)

# ตัวอย่าง 2 (ต่อ)

เขียนโปรแกรมภาษาไพทอน ScoreForAvg.py ได้ดังนี้

```
sum=0.0
inp=input(' number student : ')
n=int(inp)
for i in range(n) :
    inp=input('score : ')
    score=float(inp)
    sum += score
    print(score)
print('Sum = ', sum , 'Average = ' ,sum/n)
```



Test data

n=5	55	42	75	85	90
Sum = 347	Average = 69.4				

# ตัวอย่าง 3

โจทย์ คุณ อะเล็กซิส ฝากเงินในบัญชีสะสมทรัพย์ \$1000 ได้รับอัตราดอกเบี้ย 5% ต่อปี และดอกเบี้ยที่ได้ในแต่ละปีก็สะสมเข้าในบัญชีดังกล่าว เมื่อฝากครบ 10 ปี จงเขียนผังงานและโปรแกรมแสดงเงินในบัญชีตั้งแต่ปีที่ 1 จนถึงปีที่ 10 กำหนดสูตรดังนี้

$$a = p(1+r)^n$$

$p$  = จำนวนเงินเริ่มต้น

$r$  = อัตราดอกเบี้ย

$n$  = จำนวนปีที่ฝาก

$a$  = จำนวนเงินทั้งหมดเมื่อครบ  $n$  ปี

# จากโจทย์ กำหนดตัวแปรและเขียนผังงานได้ดังนี้

กำหนดตัวแปร

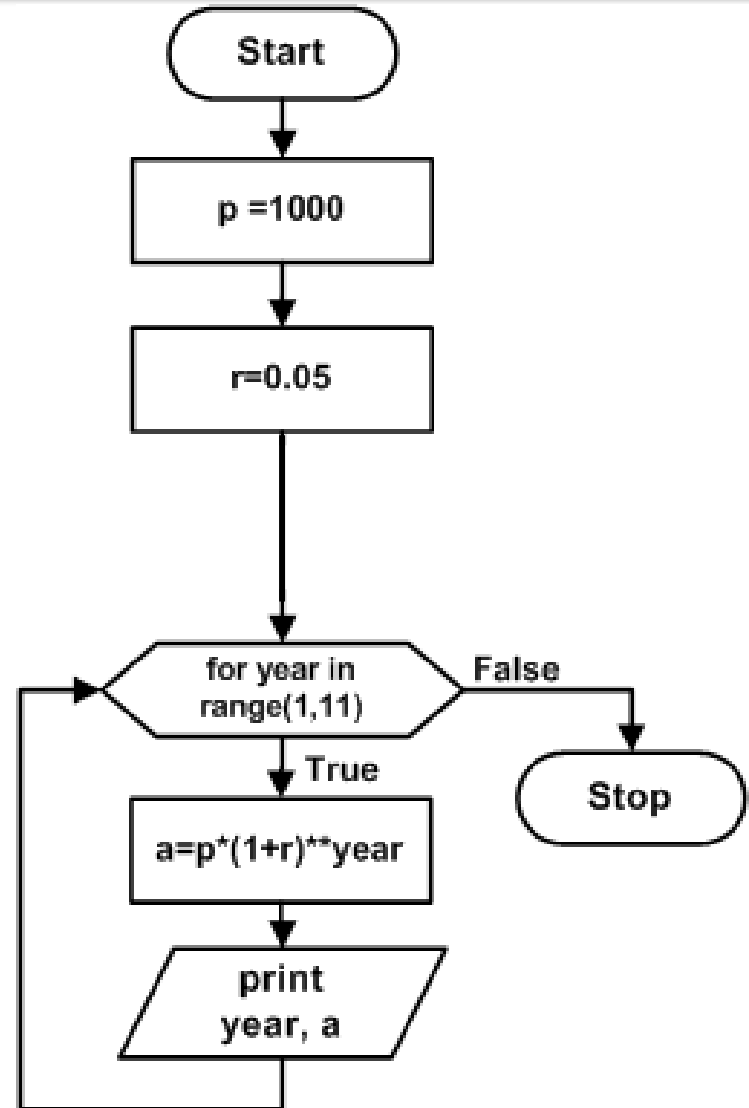
$p=1000$  จำนวนเงินเริ่มต้น

$r = 0.05$  อัตราดอกเบี้ย

$year = 1$  โปรแกรมวนรอบตามจำนวนปีที่โจทย์กำหนดคือ 10 ปี

$a$  = จำนวนเงินทั้งหมดที่ได้ ตั้งแต่ปีที่ 1 จนครบ 10 ปี

สูตร  $a=p*(1+r)**year$



# ตัวอย่าง 3 (ต่อ)

- จากโจทย์ การคำนวณเงินฝาก เขียนโปรแกรมภาษาไพทอนได้ดังนี้

```
#Calculate Amount
```

```
p=1000 # start principle
```

```
r=0.05 # interest rate
```

```
print ("Year %21s " % "Amount on deposit")
```

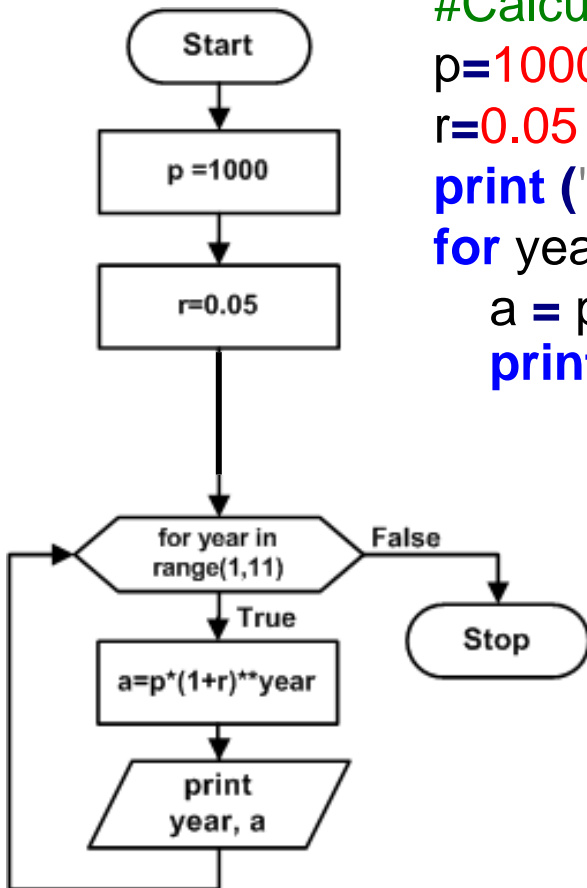
```
for year in range (1,11):
```

```
    a = p*(1+r)**year
```

```
    print ("%4d %21.2f" %(year , a))
```

จากโปรแกรมได้ผลลัพธ์ดังนี้

```
>>>
Year      Amount on deposit
 1          1050.00
 2          1102.50
 3          1157.63
 4          1215.51
 5          1276.28
 6          1340.10
 7          1407.10
 8          1477.46
 9          1551.33
10          1628.89
>>>
```



# ฟังก์ชัน print(): การแสดงผลแบบไม่ขึ้นบรรทัดใหม่

- The **keyword** argument *end* can be used to avoid the newline after the output, or end the output with a different string:

ตัวอย่าง

```
a = 0
```

```
b = 1
```

```
while b < 1000 :
```

```
    print(b, end=',')
```

```
    a = b
```

```
    b = a+b
```

ผลลัพธ์ที่ได้

```
1, 2, 4, 8, 16, 32, 64, 128, 256, 512,  
>>> |
```



# แบบฝึกหัด

1. จงเขียนผังงานและเขียนโปรแกรมสำหรับในการบวกเลขจำนวนเต็ม  $n$  จำนวน แล้วแสดงผลลัพธ์ออกทางจอภาพ

เช่น ระบุเลขจำนวนเต็มที่จะป้อน 5 ค่า

โดยค่าที่ป้อนได้แก่ 1 5 10 6 และ 9

ผลลัพธ์ที่ได้คือ 31