บทที่ 5 การเขียนโปรแกรมแบบวนซ้ำ(for)

สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

Definite Loops

Iterating over a set of items…

Definite Loops

Quite often we have a list of items of the lines in a file - effectively a finite set of things

We can write a loop to run the loop once for each of the items in a set using the Python for construct

These loops are called “definite loops” because they execute an exact number of times

We say that “definite loops iterate through the members of a set”

A Simple Definite Loop


```
for i in [5, 4, 3, 2, 1] :
    print(i)
print('Blastoff!')
```

```
5
4
3
2
1
Blastoff!
```

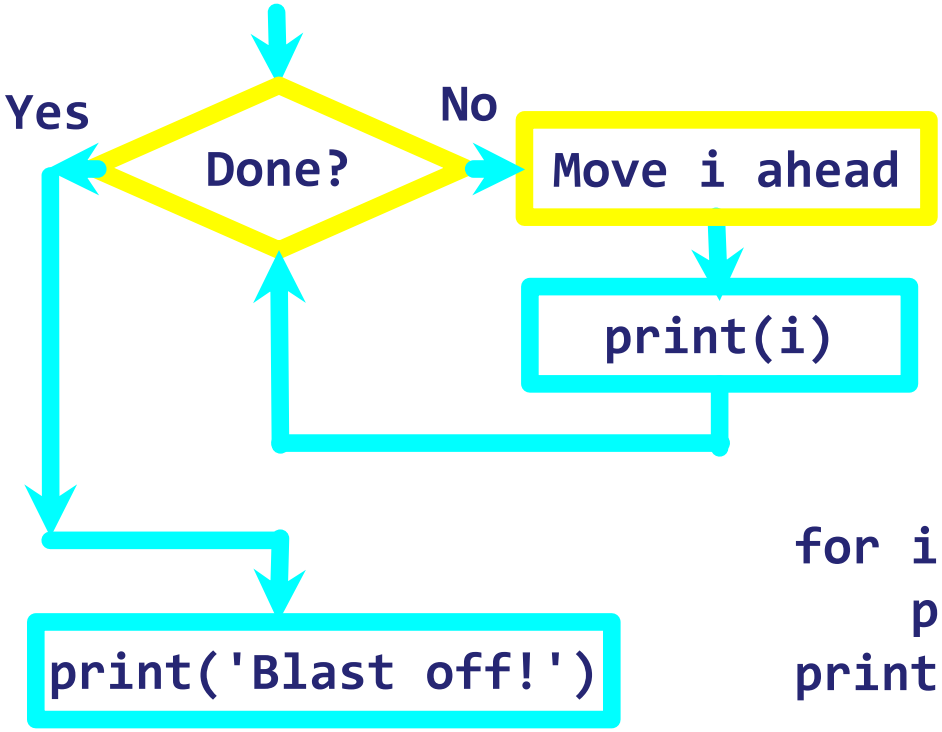
A Definite Loop with Strings


```
friends = ['Joseph', 'Glenn', 'Sally']
for friend in friends :
    print('Happy New Year:', friend)
print('Done!')
```

```
Happy New Year: Joseph
Happy New Year: Glenn
Happy New Year: Sally

Done!
```

A Simple Definite Loop




```
for i in [5, 4, 3, 2, 1] :
    print(i)
print('Blastoff!')
```

```
5
4
3
2
1
Blastoff!
```

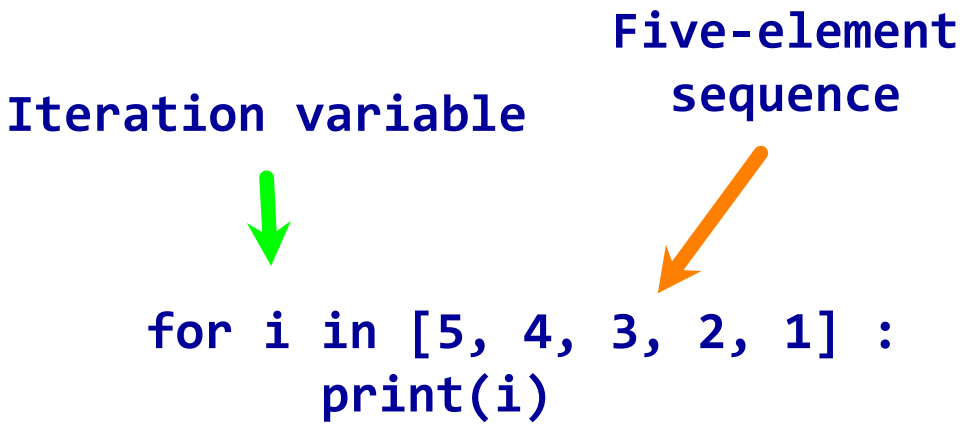
Definite loops (for loops) have explicit **iteration variables** that change each time through a loop. These **iteration variables** move through the sequence or set.

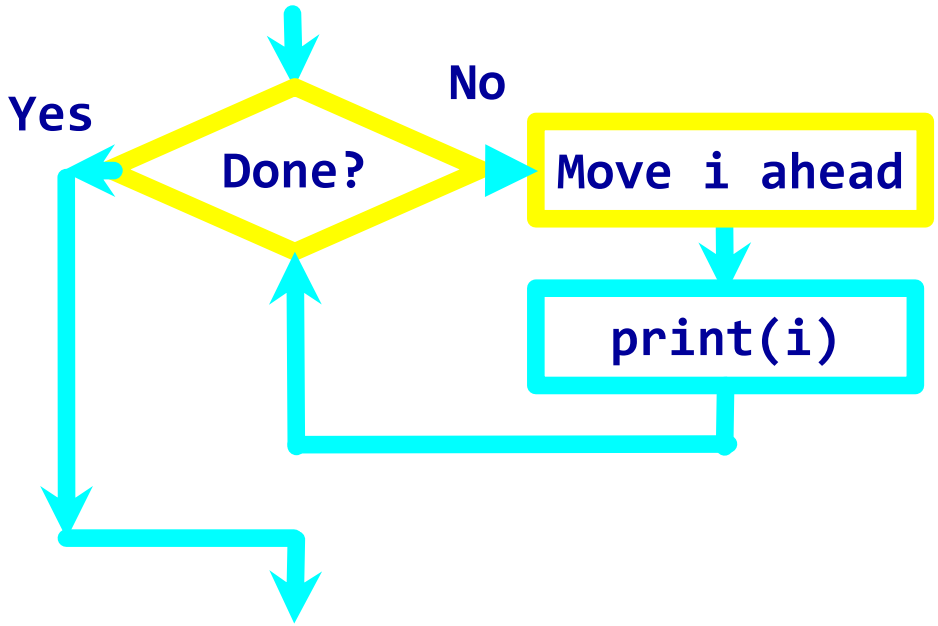
Looking at in...

The iteration variable "iterates" through the sequence (ordered set)

The block (body) of code is executed once for each value in the sequence

The iteration variable moves through all of the values in the sequence



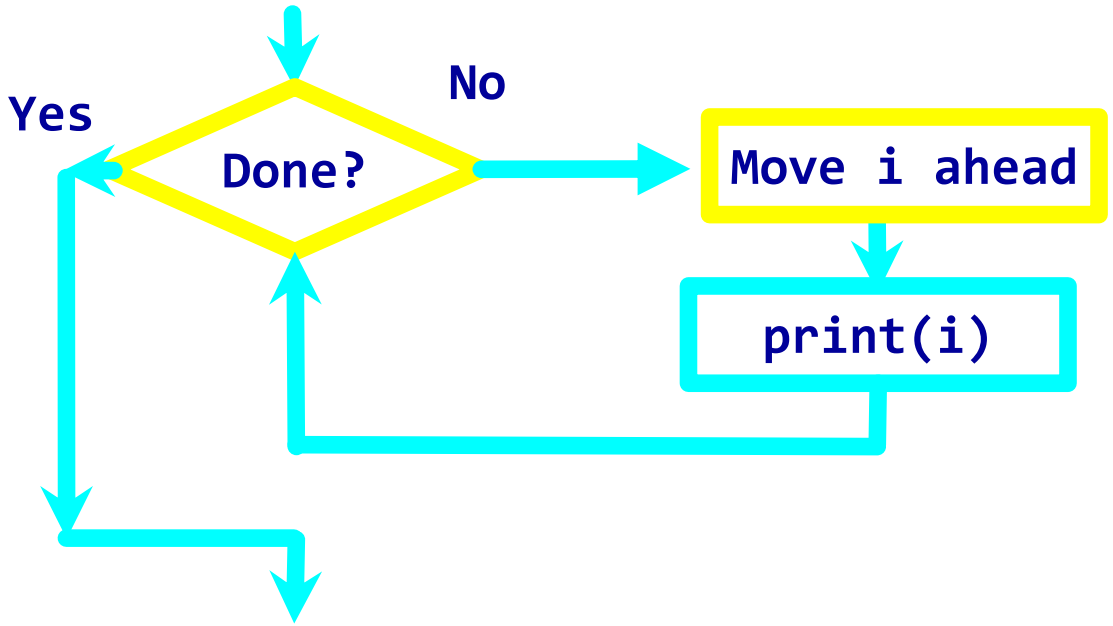



```
for i in [5, 4, 3, 2, 1] :
    print(i)
```

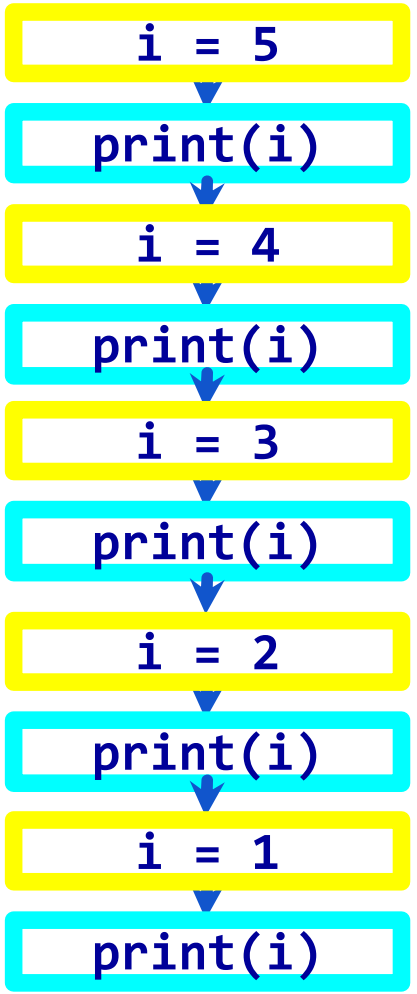
The iteration variable "iterates" through the sequence (ordered set)

The block (body) of code is executed once for each value in the sequence

The iteration variable moves through all of the values in the sequence




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



Loop Idioms: What We Do in Loops

Note: Even though these examples are simple, the patterns apply to all kinds of loops

Making "smart" loops

The trick is "knowing" something about the whole loop when you are stuck writing code that only sees one entry at a time

Set some variables to initial values

for thing in data:

Look for something or do something to each entry separately, updating a variable

Look at the variables

Looping Through a Set


```
print('Before')
for thing in [9, 41, 12, 3, 74, 15] :
     print(thing)
print('After')
```

```
Before
9
41
12
3
74
15
After
```

What is the Largest Number?

3 41 12 9 74 15

What is the Largest Number?

3 41 12 9 74 15

74

Finding the Largest Value


```
largest_so_far = -1
print('Before', largest_so_far)
for the_num in [9, 41, 12, 3, 74, 15] :
    if the_num > largest_so_far :
        largest_so_far = the_num
    print(largest_so_far, the_num)

print('After', largest_so_far)
```

```
Before -1
9  9
41  41
41 12
41  3
74  74
74  15
After 74
```

We make a variable that contains the largest value we have seen so far. If the current number we are looking at is larger, it is the new largest value we have seen so far.

More Loop Patterns…

Counting in a Loop


```
zork = 0
print 'Before', zork
for thing in [9, 41, 12, 3, 74, 15] :
    zork = zork + 1
    print zork, thing
print 'After', zork
```

```
Before 0
1 9
2 41
3 12
4 3
5 74
6 15
After 6
```

To count how many times we execute a loop, we introduce a counter variable that starts at 0 and we add one to it each time through the loop.

Summing in a Loop


```
zork = 0
print('Before', zork )
for thing in [9, 41, 12, 3, 74, 15] :
    zork = zork + thing
    print zork  thing
print('After', zork)
```

```
Before 0
9 9
50 41
62 12
65 3
139 74
154 15
After 154
```

To add up a value we encounter in a loop, we introduce a sum variable that starts at 0 and we add the value to the sum each time through the loop.

Finding the Average in a Loop


```
count = 0
sum = 0
print('Before', count, sum)
for value in [9, 41, 12, 3, 74, 15] :
    count = count + 1
    sum = sum + value
    print(count, sum, value)
print('After', count, sum, sum / count)
```

```
Before 0 0
1 9 9
2 50 41
3 62 12
4 65 3
5 139 74
6 154 15
After 6 154 25.666
```

An average just combines the counting and sum patterns and divides when the loop is done

Filtering in a Loop


```
print 'Before'
for value in [9, 41, 12, 3, 74, 15] :
    if value > 20:
        print('Large number',value)
print('After')
```

```
Before
Large number 41
Large number 74
After
```

We use an if statement in the loop to catch / filter the values we are looking for.

Search Using a Boolean Variable


```
found = False
print('Before', found)
for value in [9, 41, 12, 3, 74, 15] :
    if value == 3 :
        found = True
    print found, value
print('After', found)
```

```
Before False
False 9
False 41
False 12
True 3
True 74
True 15
After True
```

If we just want to search and know if a value was found, we use a variable that starts at False and is set to True as soon as we find what we are looking for.

How to Find the Smallest Value


```
largest_so_far = -1
print('Before', largest_so_far)
for the_num in [9, 41, 12, 3, 74, 15] :
    if the_num > largest_so_far :
        largest_so_far = the_num
    print(largest_so_far, the_num)

print('After', largest_so_far)
```

```
Before -1
9  9
41  41
41 12
41  3
74  74
74  15
After 74
```

How would we change this to make it find the smallest value in the list?

Finding the Smallest Value


```
smallest_so_far = -1
print('Before', smallest_so_far)
for the_num in [9, 41, 12, 3, 74, 15] :
    if the_num < smallest_so_far :
        smallest_so_far = the_num
    print(smallest_so_far, the_num)

print('After', smallest_so_far)
```

We switched the variable name to smallest_so_far
and switched the > to <

Finding the Smallest Value


```
smallest_so_far = -1
print('Before', smallest_so_far)
for the_num in [9, 41, 12, 3, 74, 15] :
    if the_num < smallest_so_far :
        smallest_so_far = the_num
    print(smallest_so_far, the_num)

print('After', smallest_so_far)
```

```
Before -1
-1  9
-1  41
-1 12
-1  3
-1  74
-1  15
After -1
```

We switched the variable name to smallest_so_far
and switched the > to <

Finding the Smallest Value


```
smallest = None
print 'Before'
for value in [9, 41, 12, 3, 74, 15] :
    if smallest is None :
        smallest = value
    elif value < smallest :
        smallest = value
    print smallest, value
print 'After', smallest
```

```
Before
9 9
9 41
9 12
3 3
3 74
3 15
After 3
```

We still have a variable that is the smallest so far. The first time through the loop smallest is None, so we take the first value to be the smallest.

The is and is not Operators


```
smallest = None
print('Before')
for value in [3, 41, 12, 9, 74, 15] :
    if smallest is None :
        smallest = value
    elif value < smallest :
        smallest = value
    print(smallest, value)

print('After', smallest)
```

- Python has an is operator that can be used in logical expressions
- Implies “is the same as”
- Similar to, but stronger than ==
- is not also is a logical operator

โปรแกรมแสดงผล a – z

จงเขียนโปรแกรมสำหรับแสดงผลอักษร a – z ออกทางจอภาพ

```
*** Show Alphabet ***
a b c d e f g h i j k l m n o p q r s t u v w x y z
===== End of Program =====
```

function range [range(start, stop, step)]


```
for n in range(5):
    print(n, end=" ")
```

```
0 1 2 3 4
```

```
for n in range(10):
    print(n, end=" ")
```

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

```
for n in range(3,9):
    print(n, end=" ")
```

```
3 4 5 6 7 8
```

```
for n in range(3, 9, 2):
    print(n, end=" ")
```

```
3 5 7
```

```
for n in range(13, 5, -2):
    print(n, end=" ")
```

```
13 11 9 7
```

ord, chr function


```
ch, num = "A", ord("A")
print(f'ch = {ch}, num = {num:d} = 0x{num:X}')
```

```
ch = A, num = 65 = 0x41
```

```
num = ord("Abx")
print(f'ch = {ch}, num = {num:d} = 0x{num:X}')
```

```
Traceback (most recent call last):
File "<string>", line 1, in <module>
TypeError: ord() expected a character, but string of length 3 found
```

โปรแกรมแสดงผล a – z | for


```
print(" *** Show Alphabet ***")
for i in range(26):
    num = ord("a") + i
    char = chr(num)
    print(f"{char} ", end="")
print("\n===== End of Program =====")
```

```
*** Show Alphabet ***
a b c d e f g h i j k l m n o p q r s t u v w x y z
===== End of Program =====
```

โปรแกรมแสดงผลอินพุทเป็นคำ | for


```
print(" *** Show splitted input ***")
words = input("Enter words : ")
count = 0
for word in words.split():
    print(f"{word} ", sep="=",end="=")
    count += 1
print("\ncount =",count)
print("===== End of Program =====")
```

```
 *** Show splitted input ***
Enter words : hello there
hello=there=
count = 2
===== End of Program =====
```

โปรแกรมแสดงผลอินพุทเป็นคำ | for


```
*** Show splitted input ***
Enter words : 1 2 3 4 5 6
1=2=3=4=5=6=
count = 6
===== End of Program =====
```

```
 *** Show splitted input ***
Enter words : 0 3 5 18 29
0=3=5=18=29=
count = 5
===== End of Program =====
```

โปรแกรมแสดงผลลูปข้อความ | for


```
message = "Let's party !!!"
n = 1
for x in message:
    print(f"{n}={x}",end=" ")
    n += 1
print("===== End Program =====")
```

```
1=L 2=e 3=t 4=' 5=s 6=  7=p 8=a 9=r 10=t 11=y 12=  13=! 14=! 15=!
===== End Program =====
```

โปรแกรมแสดงผลพีระมิด (version-2|for)


```
 *** Draw pyramid ***
Enter height : 5
    *
   ***
  *****
 *******
*********
===== End of program =====
```

```
 *** Draw pyramid ***
Enter height : 9
        *
       ***
      *****
     *******
    *********
   ***********
  *************
 ***************
*****************
===== End of program =====
```

โปรแกรมแสดงผลพีรามิด | for


```
print(" *** Draw pyramid (v-01) ***")
input_str = input("Enter height : ")
num = int(input_str)
space = ' '
star = '*'
for row in range(num):
    for col in range(row+num):
        if row+col < num-1:
            print(space,end="")
        else:
            print(star,end="")
    print()
print("===== End of program =====")
```

โปรแกรมแสดงผลพีรามิด | for


```
print(" *** Draw pyramid (v-02) ***")
input_str = input("Enter height : ")
num = int(input_str)
for row in range(num):
    line = ' '*(num-row-1)
    line += '*'*(2*row+1)
    print(line)
print("===== End of program =====")
```

โปรแกรมแสดงผลพีรามิด | for


```
print(" *** Draw pyramid (v-03) ***")
input_str = input("Enter height : ")
num = int(input_str)
space = ' '
star = '*'
for row in range(num):
    line = space * (num-row-1)
    line += star * (2*row+1)
    print(line)
print("===== End of program =====")
```

โปรแกรมแสดงผลพีรามิด | for


```
print(" *** Draw pyramid (v-04) ***")
input_str = input("Enter height : ")
num = int(input_str)
space = ' '
star = '*'
for row in range(num):
    num_of_space = num-row-1
    num_of_star = 2*row+1
    line = space * num_of_space
    line += star * num_of_star
    print(line)
print("===== End of program =====")
```

โปรแกรมแสดงผลพีรามิด | for


```
print(" *** Draw pyramid (v-05) ***")
input_str = input("Enter height : ")
num = int(input_str)
for row in range(num):
    print(f"{' '*(num-row-1)}{'*'*(2*row+1)}")
print("===== End of program =====")
```

โปรแกรมแสดงผลพีระมิด (0-9)


```
 *** Draw pyramid (0-9) ***
Enter height : 5
    0
   123
  45678
 9012345
678901234
===== End of program =====
```

```
  *** Draw pyramid (0-9) ***
Enter height : 9
        0
       123
      45678
     9012345
    678901234
   56789012345
  6789012345678
 901234567890123
4567890123456789O
===== End of program =====
```

โปรแกรมแสดงผลพีรามิด | for


```
print(" *** Draw pyramid (0-9) ***")
input_str = input("Enter height : ")
num = int(input_str)
n = 0
for row in range(num):
    line = ' '*(num-row-1)
    for ch in range(2*row+1):
        line += str(n%10)
        n += 1
    print(line)
print("===== End of program =====")
```

Summary (Loop)

- range function
- For loops (definite)
- Iteration variables
- None constants and variables
- is operator
- For loops (definite)
- Iteration variables
- Loop idioms
- Largest or smallest
- in str.split()
- in str