컴퓨팅사고와 파이썬 프로그래밍

Ch 3. 파이썬 프로그램 실행 제어, 기본 자료형과 연산

교수 김 영 탁

영남대학교 정보통신공학과

(Tel : +82-53-810-2497; E-mail : ytkim@yu.ac.kr)

Outline

<파이썬 프로그램 실행제어>

- 프로그램 실행 제어, 조건식과 조건문
- while-반복문
- for-반복문
- 반복문 블록 내부의 break, continue

<파이썬 기본 자료형>

- 기본 숫자 자료형 – bool, int, float, complex
- 기본 시퀀스 (sequence) 자료형 개요 – list, range

프로그램 실행 제어 - 조건문, 조건식
(if, if-elif-else)

프로그램 실행제어

◆ 프로그램 실행 제어

- 프로그램 실행 중간의 다양한 상황에 따라 다른 기능을 수행할 수 있도록 구성

◆ 프로그램 실행 제어 구조

- 조건식 (conditional expression), 조건문(conditional statement)
- 반복문 – while-loop, for-loop
- 예외처리

조건식 관련 연산자

◆ 조건식 관련 연산자

연산자의 분류	연산자	의미, 예
관계연산자 (relation-ship)	>	a > b : a가 b보다 크면 True, 아니면 False
	>=	a >= b : a가 b보다 같거나 크면 True, 아니면 False
	<	a < b : a가 b보다 작으면 True, 아니면 False
	<=	a <= b : a가 b보다 같거나 작으면 True, 아니면 False
	==	a == b : a와 b가 같으면 True, 아니면 False
	!=	a != b : a와 b가 다르면 True, 아니면 False
논리연산자 (logical)	and (논리 곱)	A and B : A와 B가 모두 True이면 True, 아니면 False
	or (논리 합)	A or B : A나 B 둘 중 하나가 True이면 True, 아니면 (즉, A와 B모두 False이면) False
	not (논리 역)	not A : A가 True이면 False, A가 False이면 True
Ternary selection	x if condition else y	조건에 따라 선택 max = x if x > y else y; (만약 x가 y보다 크면 x를 선택, 아니면 y를 선택)

조건식의 표현

◆ 논리 연산자를 사용한 조건식의 표현

주어진 조건	산술 연산자를 사용한 수식의 표현
성적 (score)이 90보다 같거나 높고, 95 보다 낮은 경우	if 90 <= score < 95: print(“Your grade is A“)

윤년 (leap year)의 조건:
연도가 4의 배수이며
100의 배수가 아니거나,
또는 400의 배수이면 윤년
	if ((year % 4 == 0) and (year % 100 != 0)) or(year % 400 == 0):
	  print(“Year(%d) is a leap year“%(year))
	else:
	  print(“Year(%d) is not a leap year“%(year))

기온이 30도 이상이며, 날씨가 화창할 때	if ((temp >= 30) and (weather == “SUNNY”)): print(“It’s good for picnic !!”)

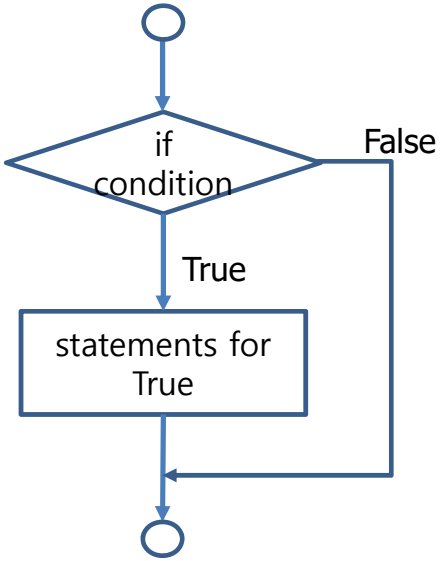
if

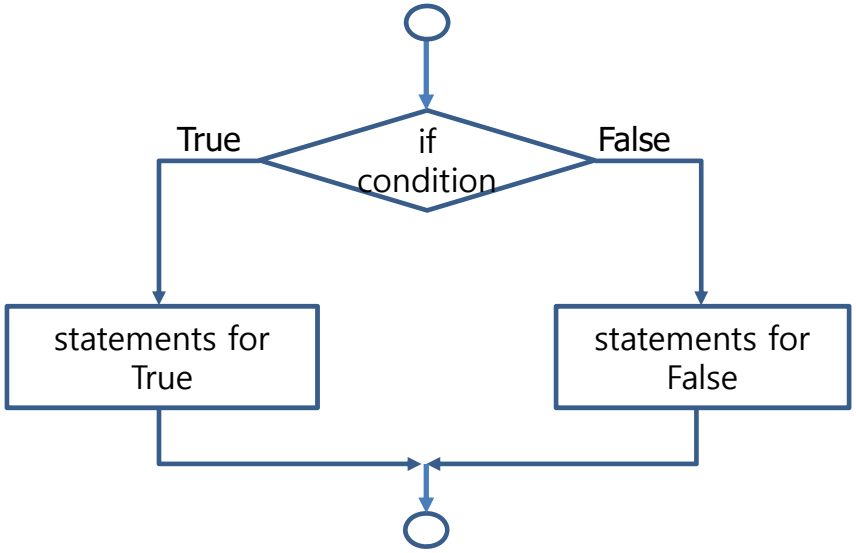
◆ Simple Branch

```
# simple program to input two integers and compare
x, y = map(int, input('input two integers (x, y) to compare : ').split())
if x == y:
    print("x(%d) is equal to y(%d)"%(x, y))
if x < y:
    print("x(%d) is less than y(%d)"%(x, y))
if x > y:
    print("x(%d) is greater than y(%d)"%(x, y))
```

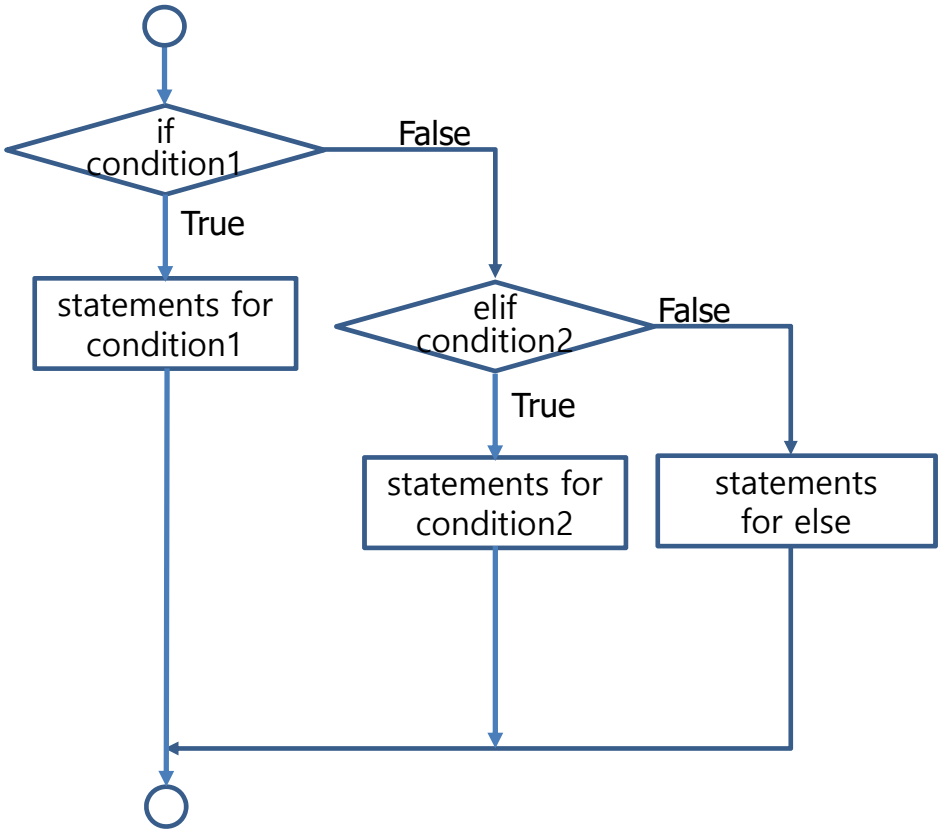
```
============= RESTART: C:\YTK-Progs\2018 Book (Python)\ch 2-1 Python Overview\
mpleCondition_if.py =============
input two integers (x, y) to compare : 3 5
x(3) is less than y(5)
>>>
============= RESTART: C:\YTK-Progs\2018 Book (Python)\ch 2-1 Python Overview\
mpleCondition_if.py =============
input two integers (x, y) to compare : 5 7
x(5) is less than y(7)
>>>
============= RESTART: C:\YTK-Progs\2018 Book (Python)\ch 2-1 Python Overview\
mpleCondition_if.py =============
input two integers (x, y) to compare : 7 5
x(7) is greater than y(5)
>>>
============= RESTART: C:\YTK-Progs\2018 Book (Python)\ch 2-1 Python Overview\
mpleCondition_if.py =============
input two integers (x, y) to compare : 5 5
x(5) is equal to y(5)
```

if, if-else statement





if-elif-else statement



if ~ else

◆ Conditional branch with if - else

```
# conditional branch with if - else
x, y = map(int, input('input two integers (x, y) : ').split())
if x>y:
    Max = x
    Min = y
else:
    Max = y
    Min = x
print("x = %d, y = %d"%(x, y))
print("Max = %d, Min = %d"%(Max, Min))
```

```
========== RESTART: C:\YTK-Progs\2018 Book (Python)\ch 2-1 Python Overview\
input two integers (x, y) : 3 5
x = 3, y = 5
Max = 5, Min = 3
>>>
========== RESTART: C:\YTK-Progs\2018 Book (Python)\ch 2-1 Python Overview\
input two integers (x, y) : 7 5
x = 7, y = 5
Max = 7, Min = 5
```

if ~ elif ~ else

◆ Conditional branch with if - else

```
# conditional branch with if - elif - else
score = int(input('course score [0..99] = '))
if 90 <= score <= 100:
    grade = 'A'
elif 80 <= score:
    grade = 'B'
elif 70 <= score:
    grade = 'C'
elif 60 <= score:
    grade = 'D'
else:
    grade = 'F'
print("score = %d, grade = %s" %(score, grade))
```

```
============== RESTART: C:/YTK-PythonProg/2_3 if_elif_else.py ==============
course score [0..99] = 95
score = 95, grade = A
>>>
============== RESTART: C:/YTK-PythonProg/2_3 if_elif_else.py ==============
course score [0..99] = 84
score = 84, grade = B
>>>
============== RESTART: C:/YTK-PythonProg/2_3 if_elif_else.py ==============
course score [0..99] = 65
score = 65, grade = D
>>>
============== RESTART: C:/YTK-PythonProg/2_3 if_elif_else.py ==============
course score [0..99] = 50
score = 50, grade = F
>>>
```

while 반복문

while 반복문 기본 구조

◆ while 반복문 기본 구조

- 조건식에서 사용되는 조건의 초기값 설정
- 조건식이 만족하는 동안 while-반복구문 실행
- while 반복구문 내부에서 조건식의 update가 반드시 있어야 함

```
조건의 초기값 설정
while 조건식 (condition):
    반복 구문
    반복 구문
    . . . .
    반복 구분
    조건의 갱신 (update)
```

while loop

◆ while-loop

```
# while-loop

L = list()
print("Input integers (-1 to end)")
x = int(input("data : "))
n = 0
while x >= 0:
    L.append(x)
    n += 1
    x = int(input("data : "))

print("Input data : ", L)
```

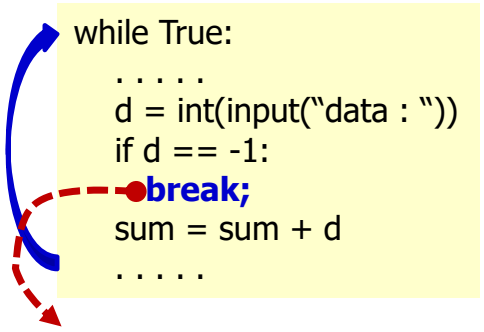
```
================ RESTART: C:/YTK-Py
Input integers (-1 to end)
data : 2
data : 3
data : 4
data : 5
data : 6
data : 7
data : -1
Input data :  [2, 3, 4, 5, 6, 7]
>>>
```

while-반복문과 break, continue

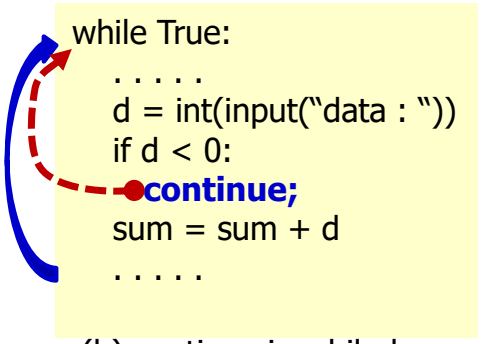
반복문	설 명

condition initialize
while condition:
  statements
  condition update
	▪ 먼저 while 반복문 조건식의 초기화를 실행
	▪ 조건식의 연산 결과가 True이면 while 반복 구문을 실행
	▪ while 반복 구문 내에서 조건식을 update

while condition1:
  if condition2:
  break
  if condition3:
  continue
  statements
	▪ 만약 condition1이 True이면 while 반복 구문 실행
	▪ 만약 conditon2가 True이면 while 반복문을 중단하고 빠져 나감
	▪ 만약 condition3가 True이면 continue 이후 구간을 생략하고 while 반복문을 계속 실행



(a) break in while-loop



(b) continue in while-loop

while-loop구조의 데이터 입력, 리스트 저장, 통계 분석

```
# while_loop with list and finding max and min

n = int(input("Input the number of data to process: "))
print("Input %d integers"%(n))
L = [] # create an empty list
count = 0
while count < n:
    d = int(input())
    L.append(d)
    if count == 0:
        Max = Min = d
    else:
        if d > Max:
            Max = d
        if d < Min:
            Min = d
    count = count + 1
print("Input data : ", L)
print("Max : ", Max)
print("Min : ", Min)
```

```
Input the number of data to process: 5
Input 5 integers
1
9
-5
2
-9
Input data :  [1, 9, -5, 2, -9]
Max :  9
Min :  -9
```

Sentinel 데이터를 사용한 while 반복문 제어

◆ Sentinel 데이터

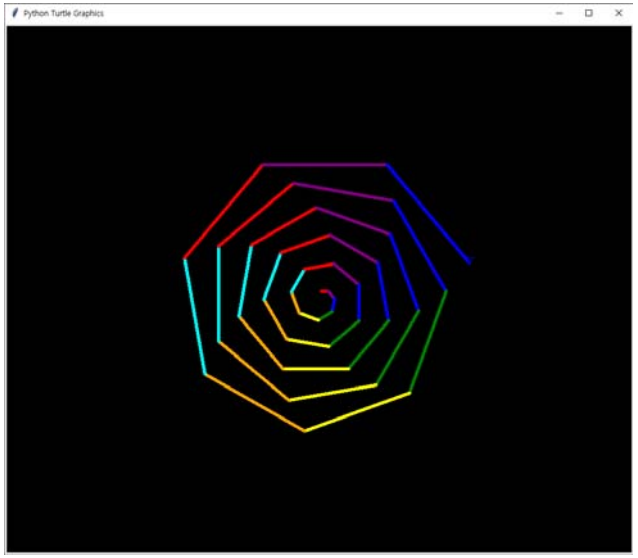
- 입력 데이터의 마지막을 표시하는 특별한 데이터
- 정상적인 데이터 입력에서 사용되지 않는 값을 사용

```
# while-loop

L = list()
while True:
    data = int(input("Input positive integers (-1 to stop) : "))
    if data == -1 :
        break
    L.append(data)
    print("L = ", L)
```

```
Input positive integers (-1 to stop) : 1
L =  [1]
Input positive integers (-1 to stop) : 2
L =  [1, 2]
Input positive integers (-1 to stop) : 3
L =  [1, 2, 3]
Input positive integers (-1 to stop) : 4
L =  [1, 2, 3, 4]
Input positive integers (-1 to stop) : 5
L =  [1, 2, 3, 4, 5]
Input positive integers (-1 to stop) : -1
```

while-loop과 터틀 그래픽

```
#turtle graphic and while-loop
import turtle
colors = ["red", "purple", "blue", "green", "yellow",\
          "orange", "cyan", "white", “violet", "brown"]

t = turtle.Turtle()

turtle.bgcolor("black")
t.width(5)
num_vertices = int(input("input num_vertices = "))
length = 10
count = 0
turn_angle = (360 // num_vertices) - 1
while length < 200:
    t.pencolor(colors[count % num_vertices])
    t.forward(length)
    t.right(turn_angle)
    length += 5
    count += 1
```



for 반복문

for 반복문

◆ for 반복문의 기본 구조

- 변수가 지정된 영역에 있는 경우 반복구문 수행
- 변수는 반복문을 실행할 때 마다 갱신되어야 함
- 변수가 주어진 조건을 만족하지 않을 때 for 반복문을 벗어남

```
for 변수 (variable) in 시퀀스 객체 (range, list, tuple, str, bytes, bytearray 등) :
    반복 구문
    반복 구문
    ....
    반복 구분
```

Control loop with for

◆ for loop with range()

```
# control loop with for
n = int(input('Input n to calculate sum of [0..n] : '))
nSum = 0
for i in range(0, n+1): #sSum = sum(range(0, n+1))
    nSum += i
print("Sum of [0..%d] = %d" %(n, nSum))
```

```
================== RESTART: C:\YTK-Progs\2018 Book (Python)\ch 2-1
Input n to calculate sum of [0..n] : 1
Sum of [0..1] = 1
>>>
================== RESTART: C:\YTK-Progs\2018 Book (Python)\ch 2-1
Input n to calculate sum of [0..n] : 10
Sum of [0..10] = 55
```

for-loop with str

```
# for-loop with str
testStr = "Python, 파이선, 12345, *?!!"
nAlphabet = nHangul = nNumber = nSymbol = nOther = 0
#
print('Test string: ', testStr)
#
for c in testStr:
    #print(c)
    if 0x41<= ord(c) <=0x5A or 0x61 <= ord(c) <= 0x7A:
        print(c, ' : Alphabet')
        nAlphabet += 1
    elif 0xAC00 <= ord(c) <= 0xD7A3:
        print(c, ' : Hangul')
        nHangul +=1
    elif 0x30 <= ord(c) <= 0x39:
        print(c, ' : Number')
        nNumber += 1
    elif 0x21 <= ord(c) <= 0x2F or 0x3A <= ord(c) <= 0x40:
        print(c, ' : Symbol')
        nSymbol += 1
    else:
        print(c, ' : Other')
        nOther += 1
totalChar = nAlphabet + nHangul + nNumber + nSymbol + nOther
#
print('Total %d characters'%totalChar)
print('nAlphabet: ', nAlphabet)
print('nHangul: ', nHangul)
print('nNumber: ', nNumber)
print('nSymbol: ', nSymbol)
print('nOther: ', nOther)
```

```
================ RESTART: C:\YTK-Progs\2(
Test string:  Python, 파이선, 12345, *?!!
P  : Alphabet
y  : Alphabet
t  : Alphabet
h  : Alphabet
o  : Alphabet
n  : Alphabet
,  : Symbol
   : Other
파  : Hangul
이  : Hangul
선  : Hangul
,  : Symbol
   : Other
1  : Number
2  : Number
3  : Number
4  : Number
5  : Number
,  : Symbol
   : Other
*  : Symbol
?  : Symbol
!  : Symbol
!  : Symbol
Total 24 characters
nAlphabet:  6
nHangul:  3
nNumber:  5
nSymbol:  7
nOther:  3
>>>
```

Getting Max and Min from list

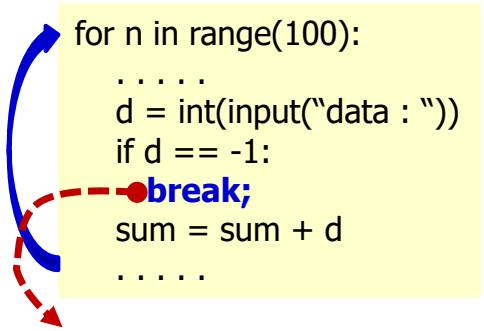
```
#Find Max and Min from List
L = [70, 85, 15, 55, 30, 90, 45, 10, 5, 60]
Max = Min = L[0]
for n in L:
    if Min > n:
        Min = n
    if Max < n:
        Max = n
print("Data (before data change) : %s"%L)
print("Min: %d, Max: %d"%(Min, Max))

#data change with index of list
for i in range(len(L)):
    L[i] *= 5
Min = min(L)
Max = max(L)
print("Data (after data changes) : %s"%L)
print("Min: %d, Max: %d"%(Min, Max))
```

```
>>>
========== RESTART: C:/YTK-PythonProg/2_6 Max and Min of List.py ==========
Data (before data change) : [70, 85, 15, 55, 30, 90, 45, 10, 5, 60]
Min: 5, Max: 90
Data (after data changes) : [350, 425, 75, 275, 150, 450, 225, 50, 25, 300]
Min: 25, Max: 450
>>>
```

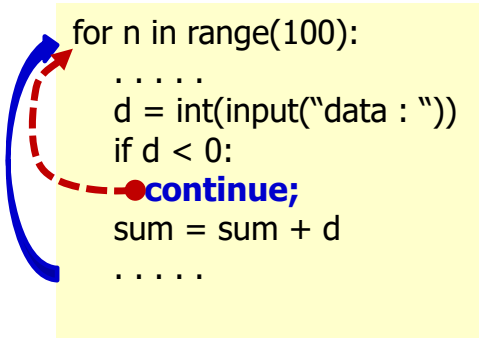
for-반복문과 break, continue

반복문	설 명
for n in sequence_type_object: statements	▪ 시퀀스 객체에 있는 원소들을 차례로 사용하면서 for 반복 구문을 실행

for n in sequence_type_object:
  if condition1:
  break
  if condition2:
  continue
  statements
	▪ 시퀀스 객체에 있는 원소들을 차례로 사용하면서 for 반복 구문을 실행
	▪ 만약 condition1이 True이면 for 반복문을 중단하고 빠져 나감
	▪ 만 약 conditon2 가 True 이 면 continue 이 후 구 간 을 생 략 하 고 for 반복문을 계속 실행



(a) break in for-loop



(b) continue in for-loop

break, continue

◆ Example of for-loop with break and continue

```
#for_loop with break and continue

n = int(input("Input number of data to process: "))
L = list()
sum = 0
print("Input %d non-negative integers"%(n))
for i in range(n):
    d = int(input())
    if d == 0:
        continue
    elif d < 0:
        break
    L.append(d)
    sum += d
print("Input data : ", L)
print("Sum = ", sum)
```

```
====== RESTART: C:/YTK-PythonProg/3_
Input number of data to process: 5
Input 5 non-negative integers
2
3
0
4
5
Input data :  [2, 3, 4, 5]
Sum =  14
>>>
====== RESTART: C:/YTK-PythonProg/3_
Input number of data to process: 5
Input 5 non-negative integers
2
-1
Input data :  [2]
Sum =  2
>>>
```

for-loop with list, set and dict

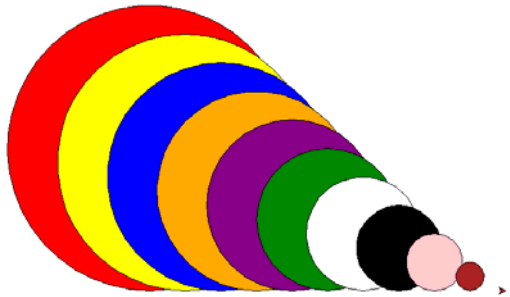
◆ for-loop with list, set and dict

```
# for-loop with list, set and dict
L = [1, 2, 3, 4, 5] #list
print('List L: ', L)
for d in L:
    print("{}".format(d), end=' ')
print()

S = {'A', 'B', 'C', 'D', 'E'} #set
print('Set S: ', S)
for a in S:
    print("{}".format(a), end= ' ')
print()

D = {1:'January', 2:'February', 3:'March', 4:'April', 5:'May'} #dict
print('Dictionary D:\n ', D)
for key, value in D.items():
    print("key = {0}: value = {1}".format(key, value))
```

```
List L:  [1, 2, 3, 4, 5]
1 2 3 4 5
Set S:  {'C', 'E', 'B', 'D', 'A'}
C E B D A
Dictionary D:
  {1: 'January', 2: 'February', 3: 'March', 4: 'April', 5: 'May'}
key = 1: value = January
key = 2: value = February
key = 3: value = March
key = 4: value = April
key = 5: value = May
```

for-loop과 터틀 그래픽

```
# drawing with for-loop
import turtle
t = turtle.Turtle()
t.shape("classic")

radius = 200
start_pos = (-250, 0)
t.up()
t.goto(start_pos)
t.down()
color_list = ["red", "yellow", "blue", "orange", "purple",\
              "green", "white", "black", "pink", "brown"]
for color in color_list:
    t.fillcolor(color)
    t.begin_fill()
    t.circle(radius)
    t.end_fill()
    radius -= 20
    t.up()
    t.forward(50)
    t.down()
```



for 반복문과 iter(), next(), enumerate()

◆ for loop with iter(), next(), enumerate() (1)

```
#iter(), next(), enumerate()
Months = ['January', 'February', 'March', 'April', \
          'May', 'June', 'July', 'August',\
          'September', 'October', 'November', 'December']
it = iter(Months) #iterator
for i in range(len(Months)):
    d = next(it)
    print(d, end=' ')
print() #new line
print() #new line

i = 0
for item in enumerate(Months):
    print("{0}".format(item), end=' ')
    i += 1
    if i%4 == 0:
        print()
print() #new line

i = 0
for item in enumerate(Months, 1):
    print(item, end =' ')
    i += 1
    if i%4 == 0:
        print()
print() #new line
```

```
>>>
========== RESTART: C:/YTK-PythonProg/2_9 iter, next, enumerate.py ==========
January February March April May June July August September October November December

(0, 'January') (1, 'February') (2, 'March') (3, 'April')
(4, 'May') (5, 'June') (6, 'July') (7, 'August')
(8, 'September') (9, 'October') (10, 'November') (11, 'December')

(1, 'January') (2, 'February') (3, 'March') (4, 'April')
(5, 'May') (6, 'June') (7, 'July') (8, 'August')
(9, 'September') (10, 'October') (11, 'November') (12, 'December')

>>>
```

for-loop과 enumerate()

◆ for-loop with enumerate()

```
# for-loop with enumerate()
Month_name = ["January", "February", "March", "April",\
              "May", "June", "July", "August",\
              "September", "October", "November", "December"]

#case A
for item in enumerate(Month_name):
    print(item)
print()

#case B
for item in enumerate(Month_name, 1):
    print(item)
print()

#case C
for i, item in enumerate(Month_name, 1):
    print("%2d"%i, item)
print()
```

(a) Case A

```
(0, 'January')
(1, 'February')
(2, 'March')
(3, 'April')
(4, 'May')
(5, 'June')
(6, 'July')
(7, 'August')
(8, 'September')
(9, 'October')
(10, 'November')
(11, 'December')
```

(b) Case B

```
(1, 'January')
(2, 'February')
(3, 'March')
(4, 'April')
(5, 'May')
(6, 'June')
(7, 'July')
(8, 'August')
(9, 'September')
(10, 'October')
(11, 'November')
(12, 'December')
```

(c) Case C

```
 1 January
 2 February
 3 March
 4 April
 5 May
 6 June
 7 July
 8 August
 9 September
10 October
11 November
12 December
```

중첩된 for-loop

◆ 2중 for-loop

```
# nested for_loop that generates multiplication table

for i in range (1, 10):
    for j in range (1, 10):
        print("{0}x{1} ={2:>3}, ".format(i, j, i*j), end= '')
    print() # new line
```

```
============= RESTART: C:/YTK-PythonProg/2_8 nested for_loop.py =============
1x1 =  1, 1x2 =  2, 1x3 =  3, 1x4 =  4, 1x5 =  5, 1x6 =  6, 1x7 =  7, 1x8 =  8, 1x9 =  9,
2x1 =  2, 2x2 =  4, 2x3 =  6, 2x4 =  8, 2x5 = 10, 2x6 = 12, 2x7 = 14, 2x8 = 16, 2x9 = 18,
3x1 =  3, 3x2 =  6, 3x3 =  9, 3x4 = 12, 3x5 = 15, 3x6 = 18, 3x7 = 21, 3x8 = 24, 3x9 = 27,
4x1 =  4, 4x2 =  8, 4x3 = 12, 4x4 = 16, 4x5 = 20, 4x6 = 24, 4x7 = 28, 4x8 = 32, 4x9 = 36,
5x1 =  5, 5x2 = 10, 5x3 = 15, 5x4 = 20, 5x5 = 25, 5x6 = 30, 5x7 = 35, 5x8 = 40, 5x9 = 45,
6x1 =  6, 6x2 = 12, 6x3 = 18, 6x4 = 24, 6x5 = 30, 6x6 = 36, 6x7 = 42, 6x8 = 48, 6x9 = 54,
7x1 =  7, 7x2 = 14, 7x3 = 21, 7x4 = 28, 7x5 = 35, 7x6 = 42, 7x7 = 49, 7x8 = 56, 7x9 = 63,
8x1 =  8, 8x2 = 16, 8x3 = 24, 8x4 = 32, 8x5 = 40, 8x6 = 48, 8x7 = 56, 8x8 = 64, 8x9 = 72,
9x1 =  9, 9x2 = 18, 9x3 = 27, 9x4 = 36, 9x5 = 45, 9x6 = 54, 9x7 = 63, 9x8 = 72, 9x9 = 81,
>>>
```

파이썬 기본 숫자 자료형과 연산
– bool, int, float, complex

자료형 클래스 (1)

◆ 파이썬 클래스 이름과 자료형

- 모든 파이썬 자료형은 클래스로 구현되어 있음
  (e.g., bool, int, float, complex, str, range, list, tuple, set, dict, etc.)

◆ 객체와 인스탄스

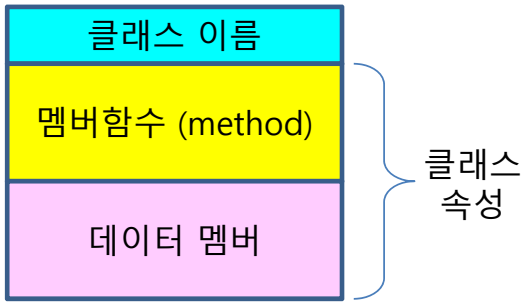
- class : 자료형을 구현하며, 속성 (멤머 데이터, 멤버 함수)를 가짐
- object : 객체 인스탄스 (object instance), 변수 (variable)
- e.g.)

class	Example of Object instance
int	i = 10 # integer
float	pi = 3.141592 # float
str	greeting = "Hello" # str

자료형 클래스 (2)

◆ 클래스 속성 (attribute)

- 클래스는 멤버함수 (메소드)와 멤버 데이터를 속성으로 가짐
- 클래스의 속성은 dot(.) 연산자를 사용하여 소속을 정의
- *e.g.)*
  - *object_name.attribute_identifier*



파이썬 자료형

◆ 파이썬 내장형 자료형 (embedded data type)

type		mutable	iterable	remarks
boolean	bool	no	no	True, False
numeric	int	no	no	
	float	no	yes	
	complex	no	yes	
sequence	str	no	yes	
	bytes	no	yes	
	bytearray	yes	yes	
	memoryview	no/yes	yes	
	list	no	yes	[0, 1, 2, 3],
	tuple	no	yes	(1, 2, 1, 2, 3)
	range	no	yes	
mapping	dict	yes	yes	{1:'A', 2:'B', 3:'C'}
set	set	yes	yes	{1, 2, 3}
	frozenset	no	yes	

정수 데이터의 리스트 생성 및 기본 연산

◆ list와 for-loop

```
# list

L = [1, 3, 5, 7, 9]
print("L = ", L)
print("len(L) = ", len(L))

sum_L = 0
for i in range(len(L)):
    sum_L = sum_L + L[i] #iterable
print("sum_L = ", sum_L)

L[0] = 100 # checking mutability
print("After L[0] = 100, L = ", L)
```

```
L =  [1, 3, 5, 7, 9]
len(L) =  5
sum_L =  25
After L[0] = 100, L =  [100, 3, 5, 7, 9]
```

range() 함수를 사용한 정수 리스트 생성

◆ range() 함수를 사용한 정수형 데이터 리스트 생성

```
# list, range(), append()

L1 = list()
for n in range(10):
    L1.append(n)
print("L1 : ", L1)

L2 = list()
for n in range(1, 10, 2):
    L2.append(n)
print("L2 : ", L2)

L3 = list()
for n in range(0, 15, 3):
    L3.append(n)
print("L3 : ", L3)
```

```
L1 :  [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
L2 :  [1, 3, 5, 7, 9]
L3 :  [0, 3, 6, 9, 12]
```

불리언 자료형 (bool)

◆ bool 자료형

- True 또는 False의 값을 가짐

◆ 불리언 자료형 연산자

논리 연산자	의미
x or y	if x is False then y else x
x and y	if x is False the x else y
not x	if x is False then True else False

◆ 불리언 자료형 연산 예

x	y	x and y	x or y	not x
True	True	True	True	False
True	False	False	True	False
False	True	False	True	False
False	False	False	False	True

불리언 원소와 list

◆ Boolean 원소의 list 구성

```
# Boolean operations wth lists of boolean elements

BL_1 = [False, False, True, True]
BL_2 = [False, True, False, True]

print("BL_1 = ", BL_1)
print("BL_2 = ", BL_2)

BL_and = []
BL_or = []
BL_not = []
for i in range(len(BL_1)):
    da = BL_1[i] and BL_2[i]
    BL_and.append(da)

    do = BL_1[i] or BL_2[i]
    BL_or.append(do)

    dn = not BL_1[i]
    BL_not.append(dn)

print("BL_1 and BL_2 = ", BL_and)
print("BL_1 or BL_2 = ", BL_or)
print("not BL_1 = ", BL_not)
```

```
BL_1 =  [False, False, True, True]
BL_2 =  [False, True, False, True]
BL_1 and BL_2 =  [False, False, False, True]
BL_1 or BL_2 =  [False, True, True, True]
not BL_1 =  [True, True, False, False]
```

숫자 자료형

◆ 파이썬 프로그램의 숫자 자료형

숫자 자료형	의미
int	<class 'int'> 소숫점 이하 값이 사용되지 않는 정수
float	<class 'float'> 소숫점 이하 값이 사용되는 실수
complex	<class 'complex'> 실수부와 허수부가 포함되는 복소수

◆ 숫자 데이터에 대한 기본 산술 연산

산술연산자	의미
c = a + b	덧셈
c = a - b	뺄셈
c = a * b	곱셈
c = a / b	실수 나눗셈
c = a // b	정수 나눗셈
c = a % b	모듈로, 나머지
c = a ** b	거듭제곱

숫자 자료형에 대한 비교 연산자

◆ 숫자 데이터에 대한 비교 연산자

비교 연산자	의미
a < b	less than
a <= b	less than or equal
a > b	greater than
a >= b	greater than or equal
a == b	equal
a != b	not equal
a is b	object identity (two objects are same)
a is not b	negated object identity (two objects are different)

숫자 자료형의 비트단위 연산자

◆ 비트단위 연산자

비트 연산자	의미
x & y	bit-wise and
x | y	bit-wise or
x ^ y	bit-wise exclusive or
~x	bit-wise not
x << n	bit-wise shift left by n bit positions
x >> n	bit-wise shift right by n bit positions

◆ 비트 단위 연산 공식

x	y	x & y	x | y	x ^ y	~x
0	0	0	0	0	1
0	1	0	1	1	1
1	0	0	1	1	0
1	1	1	1	0	0

숫자 자료형에 대한 비트단위 연산

◆ 비트단위 연산 예

변수 및 연산	연산 결과 (비트 단위 표시)	값 (16진수)
x	0000 0111	0x07
y	1010 0011	0xA3
x & y	0000 0011	0x03
x | y	1010 0111	0xA7
x ^ y	1010 0100	0xA4
~x	1111 1000	0xF8
x << 2	0001 1100	0x1C
x >> 2	0000 0001	0x01
y << 2	1000 1100	0x8C
y >> 2	1110 1000 (MSB가 sign 비트일 때)	0xE8

숫자 자료형 관련 파이썬 내장 함수

◆ 숫자 데이터 관련 파이썬 내장 함수

내장 함수	의미
abs(x)	숫자 x의 절대값을 반환
int(n_str)	정수형 문자열 n-str을 정수 데이터로 변환
float(f_str)	실수형 문자열 f_str을 실수 데이터로 변환
complex(c_str)	복소수형 문자열 c_str을 복소수 데이터로 변환
bin(x)	정수 x를 2진 binary digit 문자열로 변환하여 반환
hex(x)	정수 x를 16진 bytes 문자열로 변환하여 반환
oct(x)	수 x를 8진 bytes 문자열로 변환하여 반환
divmod(a, b)	a를 b로 나눈 후 몫과 나머지를 tuple로 반환 a = q * b + a % b 정수: (a//b, a%b) 실수: (q = math.floor(a, b), a%b)
pow(x, y[, z])	z가 없으면 x의 y 거듭제곱 결과를 반환; z가 있으면 x의 y 거듭제곱 결과를 z로 나눈 나머지 (pow(x, y) % z)를 반환
round(number[, ndigits])	실수 number를 소숫점 이하 ndigits에서 반올림;

숫자 데이터관련 내장 함수

◆ abs(), divmode(), pow(), round()

```
# embedded functions for number data
x = -123
print("x      = {:4d}".format(x))
print("abs(x) = {:4d}".format(abs(x)))

a, b = 7, 2
print("a = ", a)
print("b = ", b)

q, r = divmod(a, b)
print("q, r = divmod(a, b)")
print("q = ", q)
print("r = ", r)

a_b = pow(a, b)
print("pow({}, {}) = {}".format(a, b, a_b))

c = 3.1415926535
c_r = round(c, 4)
print("c = ", c)
print("c_r = ", c_r)
```

```
x      = -123
abs(x) =  123
a =  7
b =  2
q, r = divmod(a, b)
q =  3
r =  1
pow(7, 2) = 49
c =  3.1415926535
c_r =  3.1416
```

숫자 데이터관련 내장 함수

◆ bin(), hex(), oct()

```
# embedded functions for number data
x = 255
bin_x = bin(x)
print("bin(x) = {}".format(bin_x))
hex_x = hex(x)
print("hex(x) = {}".format(hex_x))
oct_x = oct(x)
print("oct(x) = {}".format(oct_x))
```

```
bin(x) = 0b11111111
hex(x) = 0xff
oct(x) = 0o377
```

숫자 자료형 클래스의 속성과 메소드

◆ 숫자 자료형 클래스의 주요 속성과 메소드

분류	내장 함수	의미
정수 자료형	int.bit_length(x)	정수 x를 이진수로 표현하는데 필요한 비트수를 반환 (부호와 leading 0은 제외)
	int.to_bytes(length, byteorder, *, signed=False)	정수를 length로 길이가 지정되는 바이트로 표현하는 bytes를 반환. byteorder는 "big" 또는 "little" 중 하나의 값을 가짐. signed=True이면 음수를 위해 2의 보수 (2's complement) 표현을 사용
	int.from_bytes(bytes, byteorder, *, signed=False)	bytes를 byteorder (big 또는 little)를 고려하여 정수로 변환
실수 자료형	float.hex()	실수의 16진수 표현 문자열로 변환하여 반환
	float.fromhex(s)	16진수 문자열 실수 표현 s로부터 실수를 계산하여 반환
	float.as_integer_ratio()	실수에 대한 비율을 갖는 분자와 분모를 tuple로 반환
	float.is_integer()	실수에 대응하는 오차 없는 정수가 있는지 판단
복소수 자료형	.real	복소수의 실수부
	.imag	복소수의 허수부
	complex.conjugate()	켤레복소수를 반환

숫자 자료형 클래스 – pow(), bit_length()

◆ pow(), bit_length()

```
# methods of number data types classes

d = 10
for e in range(0, 10, 1):
    de = pow(d, e)
    bit_len = int.bit_length(de)
    print("bit_length for {:12} = {:3}"\
          .format(de, bit_len))
```

```
bit_length for           1 =   1
bit_length for          10 =   4
bit_length for         100 =   7
bit_length for        1000 =  10
bit_length for       10000 =  14
bit_length for      100000 =  17
bit_length for     1000000 =  20
bit_length for    10000000 =  24
bit_length for   100000000 =  27
bit_length for  1000000000 =  30
```

복소수 (complex)

◆ class complex([real[, imag]])

```
# complex, conjugate

c1 = complex(3, 4)
print("c1 = ", c1)
print("type(c1) = ", type(c1))
print("c1.real = ", c1.real)
print("c1.imag = ", c1.imag)

cj = complex.conjugate(c1)
print("conjugate(c1) = ", cj)
print("cj.real = ", cj.real)
print("cj.imag = ", cj.imag)

c2 = complex(5, 6)
print("c2 = ", c2)
print("c1 + c2 = ", c1 + c2)
print("c1 - c2 = ", c1 - c2)
print("c1 * c2 = ", c1 * c2)
print("c1 / c2 = ", c1 / c2)
```

```
c1 =  (3+4j)
type(c1) =  <class 'complex'>
c1.real =  3.0
c1.imag =  4.0
conjugate(c1) =  (3-4j)
cj.real =  3.0
cj.imag =  -4.0
c2 =  (5+6j)
c1 + c2 =  (8+10j)
c1 - c2 =  (-2-2j)
c1 * c2 =  (-9+38j)
c1 / c2 =  (0.6393442622950819+0.03278688524590165j)
```

복소수 (complex) 입력, Conjugate

```
# complex, input, conjugate, addition, subtraction,
#   multiplication, division, conjugate

cmplx_str = input("input complex number (a+bj) : ")
c1 = complex(cmplx_str)
print("c1 = ", c1)
print("type(c1) = ", type(c1))
print("c1.real = ", c1.real)
print("c1.imag = ", c1.imag)

c2 = complex.conjugate(c1)
print("conjugate(c1) = ", c2)
print("c2.real = ", c2.real)
print("c2.imag = ", c2.imag)

print("c1 + c2 = ", c1 + c2)
print("c1 - c2 = ", c1 - c2)
print("c1 * c2 = ", c1 * c2)
print("c1 / c2 = ", c1 / c2)
```

```
input complex number (a+bj) : 5+7j
c1 =  (5+7j)
type(c1) =  <class 'complex'>
c1.real =  5.0
c1.imag =  7.0
conjugate(c1) =  (5-7j)
c2.real =  5.0
c2.imag =  -7.0
c1 + c2 =  (10+0j)
c1 - c2 =  14j
c1 * c2 =  (74+0j)
c1 / c2 =  (-0.3243243243243243+0.9459459459459459j)
```

숫자 자료형 데이터의 통계 분석

◆ while-loop, find min and max

```
# while-loop, find min and max of input data list

TARGET_NUM_DATA = 10
L = [] # empty list
num_data = 0
print("Input {} integer data."\
      .format(TARGET_NUM_DATA))
while num_data < TARGET_NUM_DATA:
    data = int(input("data = "))
    L.append(data)
    num_data = num_data + 1

L_min = min(L)
L_max = max(L)
L_sum = sum(L)
L_len = len(L)

print("L (num_data = {}) = {}"\
      .format(L_len, L))
print("Min = {}, Max = {}, Avg = {}"\
      .format(L_min, L_max, L_sum/L_len))
```

```
Input 10 integer data.
data = 3
data = 7
data = 1
data = 0
data = 2
data = 8
data = 9
data = 5
data = 4
data = 6
L (num_data = 10) = [3, 7, 1, 0, 2, 8, 9, 5, 4, 6]
Min = 0, Max = 9, Avg = 4.5
```

숫자 자료형에 대한 비교 연산

◆ find min, max

```
# while-loop, find min and max of input data list
TARGET_NUM_DATA = 10
L = [] # empty list
L_len = 0
L_sum = 0
print("Input {} integer data.".format(TARGET_NUM_DATA))
while L_len < TARGET_NUM_DATA:
    data = int(input("data = "))
    L.append(data)
    L_sum = L_sum + data
    L_len = L_len + 1
    if L_len == 1:
        L_min = L_max = data
        continue
    if data > L_max:
        L_max = data
    if data < L_min:
        L_min = data

print("L (num_data = {}) = {}"\
      .format(L_len, L))
print("Min = {}, Max = {}, Avg = {}"\
      .format(L_min, L_max, L_sum/L_len))
```

```
Input 10 integer data.
data = 3
data = 7
data = 1
data = 0
data = 2
data = 8
data = 9
data = 5
data = 4
data = 6
L (num_data = 10) = [3, 7, 1, 0, 2, 8, 9, 5, 4, 6]
Min = 0, Max = 9, Avg = 4.5
```

첨가산술대입 연산 (augmented assignment)

◆ +=, -=, *=, /=, %=, **=, >>=, <<=, &=, ^=, !=

첨가 산술 대입 연산자	예	의미
+=	v1 += v2	v1 = v1 + v2
-=	v1 -= v2	v1 = v1 - v2
*=	v1 *= v2	v1 = v1 * v2
/=	a /= b	a = a / b
//=	a //= b	a = a // b
%=	n %= m	n = n % m
**=	x **= n	x = x ** n
<<=	b <<= n	b = b << n
>>=	b >>= n	b = b >> n
&=	a &= b	a = a & b
^=	a ^= b	a = a^b
|=	a |= b	a = a | b

```
>>> x = 1
>>> x += 10
>>> x
11
>>> x -= 5
>>> x
6
>>> x *= 10
>>> x
60
>>> x %= 7
>>> x
4
>>>
```

파이썬 시퀀스 (Sequence) 자료형 개요
- range, list

시퀀스 자료형 관련 연산과 함수

◆ 시퀀스 자료형에 사용할 수 있는 연산과 함수

연산, 함수	기능	사용 예
len()	시퀀스 자료형 객체의 길이	len([0, 1, 2, 3, 4])
+	2개의 시퀀스를 연결 (concatenation)	[1, 2, 3, 4] + [5, 6, 7, 8, 9]
*	지정된 횟수만큼 시퀀스를 반복 (repetition)	['ABC'] * 3
in	시퀀스에 포함되어 있는지 확인	strList['Mon', 'Tue', Wed'] 'Mon' in strList
not in	시퀀스에 포함되어 있지 않는지 확인	5 not in [0, 1, 2, 3, 4]
[ ]	인덱싱	strList[0]
min()	시퀀스에서 제일 작은 요소	min([3, 1, 5, 7])
max()	시퀀스에서 제일 큰 요소	max([3, 1, 5, 7])
sum()	시퀀스에 포함된 원소들의 합	sum([3, 1, 5, 7])
sorted()	시퀀스에 포함된 원소들을 정렬한 결과를 반환	L = [3, 1, 5, 7, 0, 4, 2, 6] L_sorted = sorted(L)
for 반복문	반복문 구성	for n in [3, 1, 5, 7, 0, 4, 2, 6] : print(n)

파이썬 리스트 (list) 자료형

◆ 리스트 (list) 기본 함수

구분	list 자료형의 기본 함수	설명
list 생성	L = [] L = list()	포함된 항목이 없는 빈 (empty) 리스트 자료형 객체 생성
	L = [1, 2, 3, 4]	콤마로 나누어진 원소들을 차례대로 포함하는 리스트 객체 생성
	L = list(range(10))	range() 함수가 생성하는 정수 데이터를 원소로 하는 리스트 객체 생성
list 기본 연산	len(L)	리스트의 원소 개수 (길이)를 반환
	L.append(7)	리스트 L의 맨 뒤에 7을 새로운 원소로 추가
	L.expand([5, 6, 7])	리스트 L 뒤에 인수로 주어진 리스트 [5, 6, 7]의 원소들을 차례로 추가
인덱싱	L[i]	리스트에 포함된 원소를 인덱스 i로 접근하며 그 원소 값을 읽거나 변경
	L[i][j]	2차원 리스트에 포함된 원소를 인덱스 i, j로 접근하며 그 원소 값을 읽거나 변경
	del L[i]	리스트 L의 i 번째 원소를 삭제
슬라이싱	L[i:j] = [4, 5, 6, 7]	리스트 L의 i ~ j-1번째 원소를 대입식 오른쪽에서 제공하는 리스트의 원소들로 변경
	L2 = L1[i:j]	리스트 L1 i ~ j-1번째 원소를 읽어 새로운 리스트 L2를 생성
	L[i:j] = []	리스트 L1 i ~ j-1번째 원소를 삭제
	del L[i:j]	리스트 L1 i ~ j-1번째 원소를 삭제

시퀀스 자료형에 대한 기본 연산

◆ 시퀀스 자료형 list에 대한 기본 연산

```
# Application of sequence type

L = [4, 5, 6, 1, 3, 8, 9, 2, 0, 7]
sum_L = sum(L)
max_L = max(L)
min_L = min(L)
len_L = len(L)
avg_L = sum_L / len_L
print("L (size: {}) : {}".format(len_L, L))
print("Statistics of L : Max ({}), Min ({}), Avg({:7.2f})"\
      .format(max_L, min_L, avg_L))

L_sorted = sorted(L)
print("Sorted L : ", L_sorted)
print("Original L : ", L)
```

```
L (size: 10) : [4, 5, 6, 1, 3, 8, 9, 2, 0, 7]
Statistics of L : Max (9), Min (0), Avg(   4.50)
Sorted L :  [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
Original L :  [4, 5, 6, 1, 3, 8, 9, 2, 0, 7]
```

시퀀스 자료형의 접합 (concatenation), 반복 (repetition)

◆ 시퀀스 자료형의 concatenation, repetition

```
# Applications of sequence types - concatenation, repeat

Str_A = "ABCD_" # str
Str_B = "1234"
print("Str_A = ", Str_A)
print("Str_B = ", Str_B)
print("Str_A + Str_B = ", Str_A + Str_B)
print("Str_A * 3 = ", Str_A * 3)

L_A = [1, 3, 5, 7, 9] # list
L_B = [0, 2, 4, 6, 8]
print("L_A = ", L_A)
print("L_B = ", L_B)
print("L_A + L_B = ", L_A + L_B)
print("L_A * 3 = ", L_A * 3)

T_A = (1, 3, 5, 7, 9) # tuple
T_B = (0, 2, 4, 6, 8)
print("T_A = ", T_A)
print("T_B = ", T_B)
print("T_A + T_B = ", T_A + T_B)
print("T_A * 3 = ", T_A * 3)
```

```
Str_A =  ABCD_
Str_B =  1234
Str_A + Str_B =  ABCD_1234
Str_A * 3 =  ABCD_ABCD_ABCD_
L_A =  [1, 3, 5, 7, 9]
L_B =  [0, 2, 4, 6, 8]
L_A + L_B =  [1, 3, 5, 7, 9, 0, 2, 4, 6, 8]
L_A * 3 =  [1, 3, 5, 7, 9, 1, 3, 5, 7, 9, 1, 3, 5, 7, 9]
T_A =  (1, 3, 5, 7, 9)
T_B =  (0, 2, 4, 6, 8)
T_A + T_B =  (1, 3, 5, 7, 9, 0, 2, 4, 6, 8)
T_A * 3 =  (1, 3, 5, 7, 9, 1, 3, 5, 7, 9, 1, 3, 5, 7, 9)
```

시퀀스 자료형의 in, not in

◆ Sequence data type with in, not in

```
# sequence type - in, not in

L = ["black", "white", "yellow", "red", "green",\
     "blue", "orange", "purple"]
Color_to_Search = ["red", "blue", "cyan", "violet"]

print("L = ", L)
print("Color_to_search = ", Color_to_Search)

for color in Color_to_Search:
    if color in L:
        print("{} is in L".format(color))
    elif color not in L:
        print("{} is not in L".format(color))
    else:
        print("{} is not determined in L".format(color))
```

```
L =  ['black', 'white', 'yellow', 'red', 'green', 'blue', 'orange', 'purple']
Color_to_search =  ['red', 'blue', 'cyan', 'violet']
red is in L
blue is in L
cyan is not in L
violet is not in L
```

리스트 (List) 생성

◆ class list([iterable])

```
# class list

A = []
print("A = ", A)
print("type(A) = ", type(A))
A = [1, 2, 3]
print("A = ", A)

B = [x for x in range(5)]
print("B = ", B)

C = [x for x in 'abcd']
print("C = ", C)

D = [(x, x+2) for x in range(5)]
print("D = ", D)
```

```
A =  []
type(A) =  <class 'list'>
A =  [1, 2, 3]
B =  [0, 1, 2, 3, 4]
C =  ['a', 'b', 'c', 'd']
D =  [(0, 2), (1, 3), (2, 4), (3, 5), (4, 6)]
```

2차원 리스트 생성

◆ 2차원 리스트

```
# 2-D list
ROWS = 2
COLS = 3
A = [[i*3+j for j in range(COLS)] for i in range(ROWS)] # 2D array, matrix
print("A = ", A)
print("A = ")
for i in range(ROWS):  # i: 0, 1
    for j in range(COLS):  # j: 0, 1, 2
        print(A[i][j], end = " ")
    print()

B = list() # B = []
print("B = ", B)
B = list('abc')
print("B = ", B)

C = list(range(5))
print("C = ", C)

D = list((1, 2, 3))
print("D = ", D)
```

```
A =  [[0, 1, 2], [3, 4, 5]]
A =
0 1 2
3 4 5
B =  []
B =  ['a', 'b', 'c']
C =  [0, 1, 2, 3, 4]
D =  [1, 2, 3]
```

Creation of List with range(start, stop, step)

◆ Creation of List with range()

```
# sequency type - list and range()

print("range(10)        : ", end='')
for i in range(10):
    print("{:2}".format(i), end=' ')
print()

print("range(0, 10)     : ", end='')
for i in range(0, 10):
    print("{:2}".format(i), end=' ')
print()

print("range(0, 10, 2)  : ", end='')
for i in range(0, 10, 2):
    print("{:2}".format(i), end=' ')
print()

print("range(1, 10, 2)  : ", end='')
for i in range(1, 10, 2):
    print("{:2}".format(i), end=' ')
print()

print("range(10, -1, -1): ", end='')
for i in range(10, -1, -1):
    print("{:2}".format(i), end=' ')
print()
```

```
range(10)        :  0  1  2  3  4  5  6  7  8  9
range(0, 10)     :  0  1  2  3  4  5  6  7  8  9
range(0, 10, 2)  :  0  2  4  6  8
range(1, 10, 2)  :  1  3  5  7  9
range(10, -1, -1): 10  9  8  7  6  5  4  3  2  1  0
```

range(start, stop, stride) and List

◆ range() and list

```
# list and range
L = range(11, 20)
print("L = ", L)
print("len(L) = ", len(L))
print("min(L) = ", min(L))
print("max(L) = ", max(L))
print("11 in L = ", 11 in L)
print("20 in L = ", 20 in L)
print("L.start = ", L.start)
print("L.stop = ", L.stop)
print("L.step = ", L.step)
print("L.count(11) = ", L.count(11))
print("L.index(11) = ", L.index(11))
print("L[0] = ", L[0])
print("L[-1] = ", L[-1])
```

```
L =  range(11, 20)
len(L) =  9
min(L) =  11
max(L) =  19
11 in L =  True
20 in L =  False
L.start =  11
L.stop =  20
L.step =  1
L.count(11) =  1
L.index(11) =  0
L[0] =  11
L[-1] =  19
```

다양한 자료형의 원소를 포함하는 리스트

◆ list with different types, different length

```
# List with heterogeneous elements

A = [0, 1.0, '1', b'bcd']
print("A = ", A)

B = [[1, 2], [3, 4, 5], [6, 7, 8, 9, 10]]
print("B = ", B)

C = [1, 2, 3]
print("C = ", C)

D = [1, 2, 3]
print("D = ", D)
print("C == D : ", C==D)

C[0] = 10
print("C = ", C)
print("C == D : ", C==D)

E = D
print("E = ", E)
print("E is D : ", E is D)

print("id(D), id(E) = ", id(D), id(E))
```

```
A =  [0, 1.0, '1', b'bcd']
B =  [[1, 2], [3, 4, 5], [6, 7, 8, 9, 10]]
C =  [1, 2, 3]
D =  [1, 2, 3]
C == D :  True
C =  [10, 2, 3]
C == D :  False
E =  [1, 2, 3]
E is D :  True
id(D), id(E) =  19345936 19345936
```

리스트 연산자 – in, not in, +, 인덱싱, 슬라이싱

◆ sequence-type operations for list

```
# List with heterogeneous elements

A = [10, 20, 30]
print("A = ", A)
print("10 in A : ", 10 in A)
print("10 not in A : ", 10 not in A)

B = [1, 2, 3] * 3
print("B = ", B)

C = [1, 2, 3] + [4, 5, 6]
print("C = ", C)

print("C[0] = {}, C[-1] = {}"\
      .format(C[0], C[-1])) # indexing
print("C[1:4] = ", C[1:4]) # slicing
print("C[::2] = ", C[::2])
print("len(C) = {}, min(C) = {}, max(C) = {}"\
      .format(len(C), min(C), max(C)))
print("C.index(4) = ", C.index(4))
print("C.count(5) = ", C.count(5))
```

```
A =  [10, 20, 30]
10 in A :  True
10 not in A :  False
B =  [1, 2, 3, 1, 2, 3, 1, 2, 3]
C =  [1, 2, 3, 4, 5, 6]
C[0] = 1, C[-1] = 6
C[1:4] =  [2, 3, 4]
C[::2] =  [1, 3, 5]
len(C) = 6, min(C) = 1, max(C) = 6
C.index(4) =  3
C.count(5) =  1
```

리스트 슬라이싱, append(), clear()

◆ sequence-type operations for list

```
# List with heterogeneous elements

A = [1, 2, 3, 4, 5, 6]
print("A = ", A)
A[0] = 10 # indexing
print("A = ", A)

A[1:4] = [15, 25, 35] # slicing
print("A = ", A)
A[1:4] = [] # delete, del A[1:4]
print("A = ", A)

A.append(7)
print("A = ", A)

A.append([8, 9, 10])
print("A = ", A)

A.clear()
print("A = ", A)
```

```
A =  [1, 2, 3, 4, 5, 6]
A =  [10, 2, 3, 4, 5, 6]
A =  [10, 15, 25, 35, 5, 6]
A =  [10, 5, 6]
A =  [10, 5, 6, 7]
A =  [10, 5, 6, 7, [8, 9, 10]]
A =  []
```

리스트 복사 - copy

```
# Testing copy of List
import copy

A = [0, [1, 2]]
print("A (id = {}) = {}".format(id(A), A))

B = A
print("B = ", B)
B[0] = 100
print("B (id = {}) = {}".format(id(B), B))
print("A (id = {}) = {}".format(id(A), A))

C = copy.copy(A) # shallow copy on mutable item
print("\nC = copy.copy(A) = {} (id = {}) "\
    .format(C, id(C)))
print("C[0] is A[0] : ", C[0] is A[0])
C[0] = 300
print("After C[0] = 300 :")
print("C (id = {}) = {}".format(id(C), C))
print("A (id = {}) = {}".format(id(A), A))
print("C[1] is A[1] : ", C[1] is A[1])
C[1][0] = 5
print("After C[1][0] = 5")
print("C = ", C)
print("A = ", A)

D = copy.deepcopy(A) # deep copy
print("\nD = copy.deepcopy(A) = {} (id = {})".format(D, id(D)))
print("D[1] is A[1] : ", D[1] is A[1])
A[1][0] = 500
print("A (id = {}) = {}".format(id(A), A))
print("D (id = {}) = {}".format(id(D), D))
```

```
A (id = 66223560) = [0, [1, 2]]
B =  [0, [1, 2]]
B (id = 66223560) = [100, [1, 2]]
A (id = 66223560) = [100, [1, 2]]

C = copy.copy(A) = [100, [1, 2]] (id = 59714184)
C[0] is A[0] :  True
After C[0] = 300 :
C (id = 59714184) = [300, [1, 2]]
A (id = 66223560) = [100, [1, 2]]
C[1] is A[1] :  True
After C[1][0] = 5
C =  [300, [5, 2]]
A =  [100, [5, 2]]

D = copy.deepcopy(A) = [100, [5, 2]] (id = 66223112)
D[1] is A[1] :  False
A (id = 66223560) = [100, [500, 2]]
D (id = 66223112) = [100, [5, 2]]
```

리스트 – append(), extend()

◆ append() vs extend()

- L1, L2: list
- L1.append(L2) inserts the whole L2 as an element in L1
- L1.extend(L2) inserts each element in L2 as elements in L1

```
# List with append() and extend()

A = [0, 1, 2, 3, 4]
print("A = ", A)
B = [5, 6, 7, 8, 9]
print("B = ", B)
C = [10, 11, 12, 13, 14]
print("C = ", C)

A.append(C)
print("A.append(C) = ", A)
B.extend(C)
print("B.extend(C) = ", B)
```

```
A =  [0, 1, 2, 3, 4]
B =  [5, 6, 7, 8, 9]
C =  [10, 11, 12, 13, 14]
A.append(C) =  [0, 1, 2, 3, 4, [10, 11, 12, 13, 14]]
B.extend(C) =  [5, 6, 7, 8, 9, 10, 11, 12, 13, 14]
```

리스트 연산 - insert(), pop(), remove(), reverse()

◆ insert(), pop(), remove(), reverse()

```
# List with insert(), pop(), remove(), reverse()

A = [0, 1, 2]
print("A = ", A)
A.extend([3, 4, 5])
print("After A.extend([3, 4, 5]) : A = ", A)
A.insert(1, 10)
print("After A.insert(1, 10): A = ", A)
print("A.pop() : ", A.pop()) # return the last item and delete it
print("A = ", A)
A.pop(1)
print("After A.pop(1) : A = ", A)

A.remove(3)
print("After A.remove(3): A = ", A)
A.reverse()
print("After A.reverse() : A = ", A)
```

```
A =  [0, 1, 2]
After A.extend([3, 4, 5]) : A =  [0, 1, 2, 3, 4, 5]
After A.insert(1, 10): A =  [0, 10, 1, 2, 3, 4, 5]
A.pop() :  5
A =  [0, 10, 1, 2, 3, 4]
After A.pop(1) : A =  [0, 1, 2, 3, 4]
After A.remove(3): A =  [0, 1, 2, 4]
After A.reverse() : A =  [4, 2, 1, 0]
```

리스트와 all() 및 any()

◆ all(), any()

함수	의미
all(iterable)	반복 가능한 자료형 (iterable)의 모든 항목 (item) 들이 True이면 함수의 결과는 True; 어떤 항목 하나라도 False이면 결과는 False
any(iterable)	반복 가능한 자료형 (iterable)의 어떤 항목 (item) 이 True이면 함수의 결과는 True; 모든 항목이 False이면 결과는 False

```
# Application of sequence type - all, any

L_A = [True, True, False, True] # list of boolean
L_B = [False, False, True, True]
print("L_A = ", L_A)
print("L_B = ", L_B)

print("any(L_A) = ", any(L_A))
print("all(L_B) = ", all(L_B))

L_C = []
for i in range(len(L_A)):
    L_C.append(L_A[i] and L_B[i])
print("L_A and L_B = ", L_C)

L_D = []
for i in range(len(L_A)):
    L_D.append(L_A[i] or L_B[i])
print("L_A or L_B = ", L_D)
```

```
L_A =  [True, True, False, True]
L_B =  [False, False, True, True]
any(L_A) =  True
all(L_B) =  False
L_A and L_B =  [False, False, False, True]
L_A or L_B =  [True, True, True, True]
```

list와 sorted()

◆ sorted()

```
# list and sorted()

A = [2, 3, 5, 7, 1, 4]
print("A = ", A)
A_sorted = sorted(A)
print("A_sorted = ", A_sorted)

Colors = ['blue', 'green', 'orange', 'red', 'yellow', 'purple']
Colors_sorted = sorted(Colors)
print("Colors = ", Colors)
print("Colors_sorted = ", Colors_sorted)
```

```
A =  [2, 3, 5, 7, 1, 4]
A_sorted =  [1, 2, 3, 4, 5, 7]
Colors =  ['blue', 'green', 'orange', 'red', 'yellow', 'purple']
Colors_sorted =  ['blue', 'green', 'orange', 'purple', 'red', 'yellow']
```

Homework 3

Homework 3

3.1 2중 for-loop을 사용하여 1×1 ~ 12 ×12의 곱셈표를 출력하는 파이썬 프로그램을 작성하라.

(실행 예)

```
1x 1 =  1,  1x 2 =  2,  1x 3 =  3,  1x 4 =  4,  1x 5 =  5,  1x 6 =  6,  1x 7 =  7,  1x 8 =  8,  1x 9 =  9,  1x10 = 10,  1x11 = 11,  1x12 = 12,
2x 1 =  2,  2x 2 =  4,  2x 3 =  6,  2x 4 =  8,  2x 5 = 10,  2x 6 = 12,  2x 7 = 14,  2x 8 = 16,  2x 9 = 18,  2x10 = 20,  2x11 = 22,  2x12 = 24,
3x 1 =  3,  3x 2 =  6,  3x 3 =  9,  3x 4 = 12,  3x 5 = 15,  3x 6 = 18,  3x 7 = 21,  3x 8 = 24,  3x 9 = 27,  3x10 = 30,  3x11 = 33,  3x12 = 36,
4x 1 =  4,  4x 2 =  8,  4x 3 = 12,  4x 4 = 16,  4x 5 = 20,  4x 6 = 24,  4x 7 = 28,  4x 8 = 32,  4x 9 = 36,  4x10 = 40,  4x11 = 44,  4x12 = 48,
5x 1 =  5,  5x 2 = 10,  5x 3 = 15,  5x 4 = 20,  5x 5 = 25,  5x 6 = 30,  5x 7 = 35,  5x 8 = 40,  5x 9 = 45,  5x10 = 50,  5x11 = 55,  5x12 = 60,
6x 1 =  6,  6x 2 = 12,  6x 3 = 18,  6x 4 = 24,  6x 5 = 30,  6x 6 = 36,  6x 7 = 42,  6x 8 = 48,  6x 9 = 54,  6x10 = 60,  6x11 = 66,  6x12 = 72,
7x 1 =  7,  7x 2 = 14,  7x 3 = 21,  7x 4 = 28,  7x 5 = 35,  7x 6 = 42,  7x 7 = 49,  7x 8 = 56,  7x 9 = 63,  7x10 = 70,  7x11 = 77,  7x12 = 84,
8x 1 =  8,  8x 2 = 16,  8x 3 = 24,  8x 4 = 32,  8x 5 = 40,  8x 6 = 48,  8x 7 = 56,  8x 8 = 64,  8x 9 = 72,  8x10 = 80,  8x11 = 88,  8x12 = 96,
9x 1 =  9,  9x 2 = 18,  9x 3 = 27,  9x 4 = 36,  9x 5 = 45,  9x 6 = 54,  9x 7 = 63,  9x 8 = 72,  9x 9 = 81,  9x10 = 90,  9x11 = 99,  9x12 =108,
10x 1 = 10, 10x 2 = 20, 10x 3 = 30, 10x 4 = 40, 10x 5 = 50, 10x 6 = 60, 10x 7 = 70, 10x 8 = 80, 10x 9 = 90, 10x10 =100, 10x11 =110, 10x12 =120,
11x 1 = 11, 11x 2 = 22, 11x 3 = 33, 11x 4 = 44, 11x 5 = 55, 11x 6 = 66, 11x 7 = 77, 11x 8 = 88, 11x 9 = 99, 11x10 =110, 11x11 =121, 11x12 =132,
12x 1 = 12, 12x 2 = 24, 12x 3 = 36, 12x 4 = 48, 12x 5 = 60, 12x 6 = 72, 12x 7 = 84, 12x 8 = 96, 12x 9 =108, 12x10 =120, 12x11 =132, 12x12 =144,
```

Homework 3

3.2 2개의 16진수 데이터 문자열을 한 줄로 입력 받고, 이를 정수로 변환하여 a와 b에 저장하라. 이 두 16진수 a, b의 bit-wise AND, bit-wise OR, bit-wise XOR 값을 계산하여 출력하는 파이썬 프로그램을 작성하라.

(실행 예)

```
input two hexadecimal numbers (예: 0xA3 0x3A) : 0xA3 0x3A
a = 0xa3 = 0b10100011
b = 0x3a = 0b00111010
a & b = 0x22 = 0b00100010
a | b = 0xbb = 0b10111011
a ^ b = 0x99 = 0b10011001
```

Homework 3

3.3 날짜를 나타내는 연(year), 월(month), 일(day)의 3개 정수를 입력 받고, 이 날이 서기 1년 1월 1일부터 몇 번째 날짜인지를 계산하며, 이 날이 무슨 요일인지 계산하여 출력하는 파이썬 프로그램을 작성하라. 참고로 서기 1년 1월 1일은 월요일이다. 프로그램은 0 0 0이 입력될 때 까지 반복하도록 할 것.
(실행 예)

```
input year month day : 1 1 1
Input yr_mn_dy_strings :  ['1', '1', '1']
Day (year(1), month(1), day(1)) : week_day (MON), elapsed 1 days from Jan01AD01
input year month day : 2021 1 1
Input yr_mn_dy_strings :  ['2021', '1', '1']
Day (year(2021), month(1), day(1)) : week_day (FRI), elapsed 737791 days from Jan01AD01
input year month day : 2021 3 1
Input yr_mn_dy_strings :  ['2021', '3', '1']
Day (year(2021), month(3), day(1)) : week_day (MON), elapsed 737850 days from Jan01AD01
input year month day : 2021 4 1
Input yr_mn_dy_strings :  ['2021', '4', '1']
Day (year(2021), month(4), day(1)) : week_day (THR), elapsed 737881 days from Jan01AD01
input year month day : 0 0 0
Input yr_mn_dy_strings :  ['0', '0', '0']
```

Homework 3

3.4 날짜를 나타내는 연(year), 월(month), 일(day)의 3개 정수를 입력 받고, 이 달의 달력을 출력하는 파이썬 프로그램을 작성하라. 달력의 첫 줄은 요일을 의미하는 영문 약자 (SUN, MON, TUE, WED, THR, FRI, SAT)를 출력하고, 이 달의 1일 부터 요일에 맞추어 출력되도록 할 것.
(실행 예)

```
input year month day : 2022 3 1

March of Year 2022
============================
 SUN MON TUE WED THR FRI SAT
----------------------------
                1   2   3   4   5
   6   7   8   9  10  11  12
  13  14  15  16  17  18  19
  20  21  22  23  24  25  26
  27  28  29  30  31
============================
```

```
input year month day : 2022 4 1

April of Year 2022
============================
 SUN MON TUE WED THR FRI SAT
----------------------------
                        1   2
   3   4   5   6   7   8   9
  10  11  12  13  14  15  16
  17  18  19  20  21  22  23
  24  25  26  27  28  29  30

============================
```

Homework 3

3.5 시간을 나타내는 시 (hour), 분(min), 초(sec)의 3개 정수를 한 줄로 입력 받고, 이 시간이 그날의 0시 0분 0초로 부터 몇 초가 경과되었는지 계산하고, 그날의 자정 (24:00:00)까지 몇 초가 남았는지 계산하여 출력하는 파이썬 프로그램을 작성하라. 시간의 출력 양식은 (00:00:00) ~ (23:59:59)로 표시할 것.
(실행 예)

```
input hour min sec : 0 0 0
Input time : (00:00:00)
Elapsed seconds from last midnight =  0
Remaining seconds to next-midnight =  86400
input hour min sec : 0 0 1
Input time : (00:00:01)
Elapsed seconds from last midnight =  1
Remaining seconds to next-midnight =  86399
input hour min sec : 1 0 0
Input time : (01:00:00)
Elapsed seconds from last midnight =  3600
Remaining seconds to next-midnight =  82800
input hour min sec : 23 59 59
Input time : (23:59:59)
Elapsed seconds from last midnight =  86399
Remaining seconds to next-midnight =  1
input hour min sec : 23 0 0
Input time : (23:00:00)
Elapsed seconds from last midnight =  82800
Remaining seconds to next-midnight =  3600
input hour min sec :
```