



# 인공지능 활용을 위한 파이썬 기초

## [2차시]



# 목 차

- 01. 반복문의 이해
- 02. 반복문 for 활용
- 03. 반복문 while 활용
- 04. 다중 for 활용
- 05. turtle 이해하기
- 06. turtle 활용하기
- 07. 예외처리의 이해와 활용
- 08. 문자열의 이해
- 09. 리스트의 이해
- 10. 문자열과 리스트의 활용

# 01. 반복문의 이해

## [2차시]

# 학습목표

---

- 반복문에서 조건절 의미 이해하기
- while() 기본 이해하기
- for() 기본 이해하기

# 반복문

- 반복되는 일들을 처리하는 프로그램 필요
  - 동일하거나, 규칙적으로 변화하는 작업
  - 조건에 맞는동안 반복
- 반복문 종류
  - while
  - for

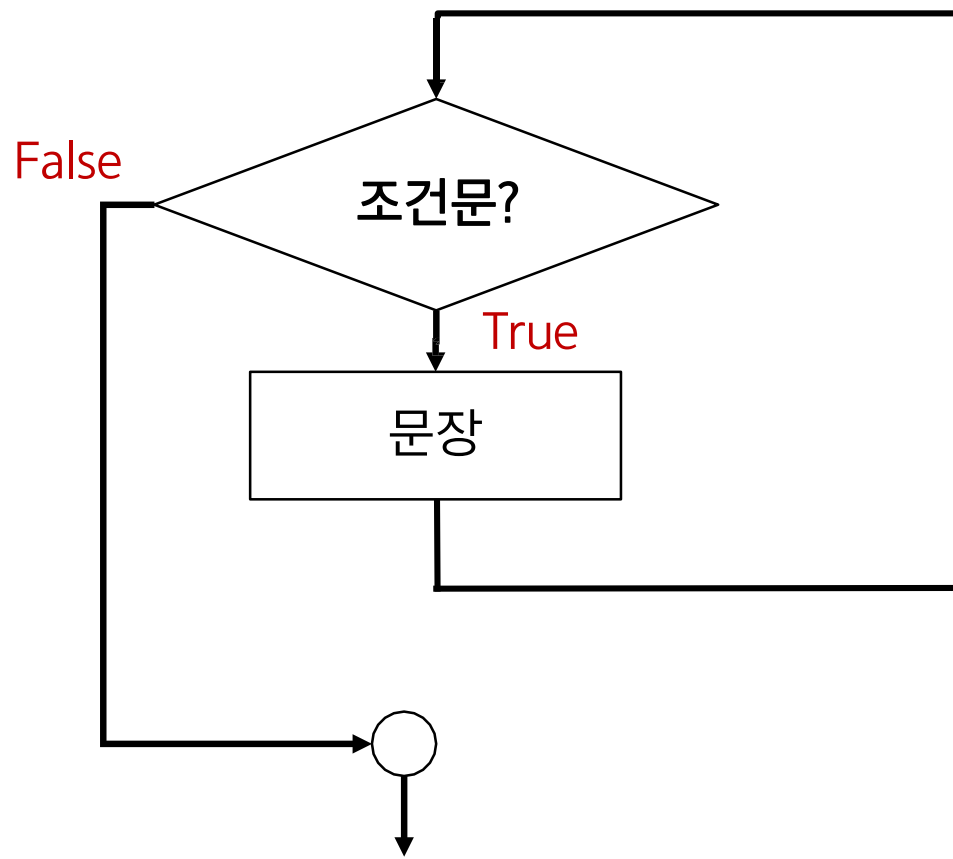
```
# print 0,1,2,...9
```

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

```
# print 0,1,2,...9
```

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

# 반복문 흐름



# while문

```
# print 0,1,2,...9
i = 0
while i < 10 :
    print(i)
    i = i + 1

print('exit value: ', i)
```

- 조건절을 평가, True또는 False를 확인
- 만약 조건절의 결과가 False(0)이면
  - while문에서 빠져 나와 다음 명령문을 실행
- 만약 조건절의 결과가 True(1)이면
  - while문 안의 몸체(body)를 실행
  - 몸체를 실행한 이후 다시 1단계부터 시작함

# while문 연습하기 1

# 숫자 출력

```
value = 50
```

```
while value < 100 :  
    value = value + 10  
    print(value)
```

```
print("last value= ", value)
```

```
60  
70  
80  
90  
100  
last value= 100
```



# while문 연습하기 2

# 숫자 출력

```
value = 50
```

```
while value > 0:  
    value = value - 5  
    print(value)
```

```
print("last value= ", value)
```

```
↳ 45  
   40  
   35  
   30  
   25  
   20  
   15  
   10  
   5  
   0  
   last value=  0
```

# while문 연습하기 3

# 숫자 출력

```
i = 1
while i <= 2**10:
    print(i)
    i = i * 2

print("last i= ", i)
```

```
1
2
4
8
16
32
64
128
256
512
1024
last i= 2048
```

# while문 연습하기 4

# 숫자 출력

```
i = 2**10
while i > 1:
    print(i)
    i = i // 2

print("last i= ", i)
```

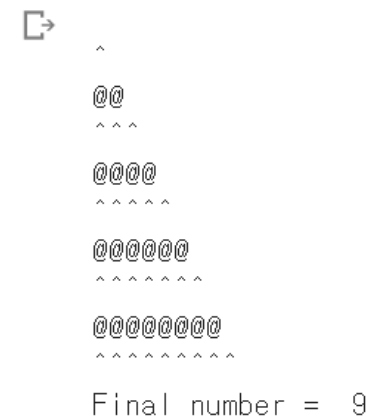
```
1024
512
256
128
64
32
16
8
4
2
last i= 1
```

# for문

- for 루프(loop)는 반복을 기반으로 실행
  - 반복 가능한 문자열, 리스트 활용
  - range 함수 사용

```
for <variable> in <sequence>:  
    <statements>
```

```
for i in range(10):  
    if i%2 == 0:  
        print("@" * i)  
    else:  
        print("^" * i)  
  
print("Final number = ", i)
```

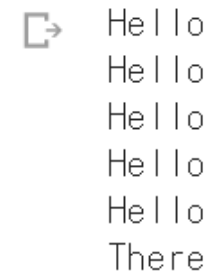


```
^  
@@  
^^^  
@@@@  
^^^^  
@@@@@@  
^^^^^^  
@@@@@@@@  
^^^^^^^^  
Final number = 9
```

# for문 연습하기 1

# 'Hello'를 5번 , 'There'를 한번 출력

```
for i in range(5):  
    print ("Hello")  
  
print ("There")
```




```
↳ Hello  
Hello  
Hello  
Hello  
Hello  
There
```

# for문 연습하기 2

# 'Hello','There'를 둘다 5번 출력

```
for i in range(5):  
    print("Hello")  
    print("There")
```



```
↳ Hello  
    There  
    Hello  
    There  
    Hello  
    There  
    Hello  
    There  
    Hello  
    There
```

# for문 연습하기 3

# 0에서 9까지 숫자를 출력

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

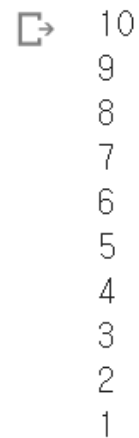


```
0  
1  
2  
3  
4  
5  
6  
7  
8  
9
```

# for문 연습하기 4

# 10에서 1까지 숫자를 감소하며 출력

```
for i in range(10,0,-1):  
    print(i)
```



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



# 연습문제 1.1

---

문법적으로 틀린 부분을 찾아보세요.

```
# 1,4,7 제곱 계산
```

```
for i in range(1,10,3)  
    print(i, i**2)
```

# 연습문제 1.1-답안

---

문법적으로 틀린 부분을 찾아보세요.

```
# 1,4,7 제곱 계산
```

```
for i in range(1,10,3) :  
    print(i, i**2)
```

## 연습문제 1.2

---

문법적으로 틀린 부분을 찾아보세요.

```
# 1~10까지 출력하기
```

```
i = 1
```

```
while i < 10 :
```

```
    print(i)
```

```
    i = i + 1
```

```
print("last i = ", i)
```

## 연습문제 1.2-답안

문법적으로 틀린 부분을 찾아보세요.

```
# 1~10까지 출력하기
```

```
i = 1
while i <= 10 :
    print(i)
    i = i + 1

print("last i = ", i)
```

```
1
2
3
4
5
6
7
8
9
last i = 10
```

## 연습문제 1.3

---

문법적으로 틀린 부분을 찾아보세요.

```
# 10~1까지 역순으로 출력하기
```

```
i = 10
```

```
while i > 1 :
```

```
    print(i)
```

```
    i = i - 1
```

```
print("last i = ", i)
```

## 연습문제 1.3-답안

문법적으로 틀린 부분을 찾아보세요.

```
# 10~1까지 역순으로 출력하기
```

```
i = 10
while i >= 1 :
    print(i)
    i = i - 1

print("last i = ", i)
```

```
10
9
8
7
6
5
4
3
2
last i = 1
```

## 연습문제 2.1

---

실행 결과를 예상해서 적어보세요.

```
i = 0
j = 10
n = 0

while i < j:
    i = i + 1
    n = n + 2

    print("i= ", i)
    print("n= ", n, "***")
```

## 연습문제 2.1-답안

실행 결과를 예상해서 적어보세요.

```
i = 0
j = 10
n = 0

while i < j:
    i = i + 1
    n = n + 2

    print("i= ", i)
    print("n= ", n, "***")
```

```
➞ i = 1
   n = 2 ***
   i = 2
   n = 4 ***
   i = 3
   n = 6 ***
   i = 4
   n = 8 ***
   i = 5
   n = 10 ***
   i = 6
   n = 12 ***
   i = 7
   n = 14 ***
   i = 8
   n = 16 ***
   i = 9
   n = 18 ***
   i = 10
   n = 20 ***
```



## 연습문제 2.2

---

실행 결과를 예상해서 적어보세요.

```
for i in [1, 3, 6, 9]:  
    print("i = ", i)  
    print("i ** 2 = ", i**2)
```

## 연습문제 2.2-답안

실행 결과를 예상해서 적어보세요.

```
for i in [1, 3, 6, 9]:  
    print("i = ", i)  
    print("i ** 2 = ", i**2)
```

```
➞ i = 1  
   i ** 2 = 1  
   i = 3  
   i ** 2 = 9  
   i = 6  
   i ** 2 = 36  
   i = 9  
   i ** 2 = 81
```

## 연습문제 2.3

---

실행 결과를 예상해서 적어보세요.

```
num = 0

while num < 20 :
    print(num)
    num = num + 2

print("last num: ", num)
```

## 연습문제 2.3-답안

실행 결과를 예상해서 적어보세요.

```
num = 0

while num < 20 :
    print(num)
    num = num + 2

print("last num: ", num)
```

```
0
2
4
6
8
10
12
14
16
18
last num: 20
```

# 강의 요약

---

- **반복문에서 조건절 의미 이해하기**
  - 조건절의 값에 따라 반복 또는 정지가 결정됨
- **while() 기본 이해하기**
  - 조건절의 결과가 False이면 while문에서 빠져나옴
  - 조건절의 결과가 True이면 while문 내부를 실행
- **for() 기본 이해하기**
  - 조건절에 반복 가능한 문자열, 리스트, range() 함수 사용

# 퀴즈 1

다음 코드의 실행 결과는?

```
value = 50
while value < 100 :
    print(value)
    value = value + 15
print("last value= ", value)
```

①

50  
65  
80  
95  
110  
last value= 110

②

50  
65  
80  
95  
last value= 110

③

50  
65  
80  
95  
110  
last value= 125

④

50  
65  
80  
95  
last value= 95

## 퀴즈 1-답안

다음 코드의 실행 결과는?

```
value = 50
while value < 100 :
    print(value)
    value = value + 15
print("last value= ", value)
```

①

50  
65  
80  
95  
110  
last value= 110

②

50  
65  
80  
95  
last value= 110

③

50  
65  
80  
95  
110  
last value= 125

④

50  
65  
80  
95  
last value= 95

## 퀴즈 2

다음과 같은 결과가 나타나는 코드는?

```
Hello  
There  
Hello  
There  
Hello  
There
```

①

```
for i in range(6) :  
    print("Hello")  
    print("There")
```

②

```
for i in range(3) :  
    print("Hello")  
    print("There")
```

③

```
for i in range(3) :  
    print("Hello")  
    print("There")
```

④

```
for i in range(6) :  
    print("Hello")  
    print("There")
```



## 퀴즈 2-답안

다음과 같은 결과가 나타나는 코드는?

```
Hello  
There  
Hello  
There  
Hello  
There
```

①

```
for i in range(6) :  
    print("Hello")  
    print("There")
```

②

```
for i in range(3) :  
    print("Hello")  
    print("There")
```

③

```
for i in range(3) :  
    print("Hello")  
    print("There")
```

④

```
for i in range(6) :  
    print("Hello")  
    print("There")
```

## 02. 반복문 for 활용

### [2차시]

# 학습목표

---

- for 다양하게 활용하기
- range, in 이해하기
- range, in 다양하게 활용하기

# for, while 표현하기 1

- 특정한 구간에서 홀수만 출력하기
  - 규칙적인 변화가 있는 경우, for가 코드가 짧고
  - 가독성이 좋음

```
# for()
n = int(input("양의 정수 입력 : "))

for i in range(1, n, 2):
    print(i)
```

```
☞ 양의 정수 입력 : 8
1
3
5
7
```

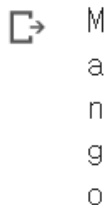
```
# while()
n = int(input("양의 정수 입력 : "))

i = 1
while i < n:
    print(i)
    i = i + 2
```

## for, while 표현하기 2

- 문자열을 한 글자씩 출력하기
  - 간략하게 표현된 것이 for
  - 문자열, 리스트를 활용 시 for가 간단함

```
# for()
for letter in "Mango":
    print(letter)
```



```
↳ M
   a
   n
   g
   o
```

```
# while()
i = 0
fr = "Mango"

while i < len(fr):
    letter = fr[i]
    print(letter)
    i = i + 1
```

## 연습문제 1

---

2에서 10까지 짝수를 출력한다

- for, in, range 사용한다

```
2  
4  
6  
8  
10
```

## 연습문제 1-코드

# 2에서 10까지 짝수를 출력하는 두 가지 방법

# 방법1

```
for i in range(2,12,2):  
    print(i)
```

# 방법2

```
for i in range(5):  
    print((i+1)*2)
```

```
2  
4  
6  
8  
10
```

## 연습문제 2

1에서 50 사이의 7의 배수와 그 수의 제곱 값을 출력 하시오

- for, range, in 사용한다

```
➞ 7 = 49  
   14 = 196  
   21 = 441  
   28 = 784  
   35 = 1225  
   42 = 1764  
   49 = 2401
```



## 연습문제 2-코드

# 2에서 10까지 짝수를 출력하는 두 가지 방법

# 방법1

```
for i in range(1, 50, 1):  
    if i % 2 == 0:  
        print(i, "=", i**2)
```

# 방법2

```
for i in range(2, 50, 2):  
    print(i, "=", i**2)
```

```
7 = 49  
14 = 196  
21 = 441  
28 = 784  
35 = 1225  
42 = 1764  
49 = 2401
```

# range 함수

- 내장 함수 **range**
  - 숫자들을 나열하여 반복을 수행하기에 적합
  - 산술 계산을 수행하는 반복의 기반을 생성

<code>list(range(10))</code>	<code>[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]</code>
<code>list(range(4,20))</code>	<code>[4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]</code>
<code>list(range(1,20,3))</code>	<code>[1, 4, 7, 10, 13, 16, 19]</code>
<code>list(range(100,20,-5))</code>	<code>[100, 95, 90, 85, 80, 75, 70, 65, 60, 55, 50, 45, 40, 35, 30, 25]</code>
<code>list(range(1,15,2))</code>	<code>[1, 3, 5, 7, 9, 11, 13]</code>

# in 함수

- 내장함수 in
  - range 와 같이 사용

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

```
name = "홍길동"  
for n in name:  
    print(n*5)
```

```
0  
1  
2  
3  
4  
5  
6  
7  
8  
9  
홍홍홍홍홍  
길길길길길  
동동동동동
```

## 연습문제 2

다음 표의 for문을 보고, 몇 번 실행되는지 쓰시오

for문	실행 횟수
for i in range(1, 11)	
for i in range(10)	
for i in range(10, 0, -1)	
for i in range(-10, 3)	
for i in range(10, 0)	
for i in range(-10, 11, 3)	

## 연습문제 2-답안

다음 표의 for문을 보고, 몇 번 실행되는지 쓰시오

for문	실행 횟수
for i in range(1, 11)	10번, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
for i in range(10)	10번, [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
for i in range(10, 0, -1)	10번, [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
for i in range(-10, 3)	13번, [-10, -9, -8, -7, -6, -5, -4, -3, -2, -1, 0, 1, 2]
for i in range(10, 0)	0번, []
for i in range(-10, 11, 3)	7번, [-10, -7, -4, -1, 2, 5, 8]

# for문 활용 1

# 문자열 처리

```
name = "HORSTMANN"  
count = 0
```

```
for char in name :  
    print(char)  
    count = count + 1  
  
print("글자수 = ", count)
```

```
↳ H  
   O  
   R  
   S  
   T  
   M  
   A  
   N  
   N  
   글자수 = 9
```

## for문 활용 2

```
# 리스트 처리  
fruit = ["apple", "banana", "lemon", "tomato"]  
count = 0  
  
for fr in fruit :  
    print(fr)  
    count = count + 1  
  
print("아이템 수 = ", count)
```

```
➞ apple  
   banana  
   lemon  
   tomato  
   아이템 수 = 4
```

## for문 활용 3

```
# 리스트 처리
fruit = ["apple", "banana", "lemon", "tomato"]
for fr in fruit :
    count = 0
    for f in fr :
        print(f)
        count = count + 1
    print("글자 수 = ", count)
```

```
↳ a
   p
   p
   l
   e
  글자 수 = 5
  b
  a
  n
  a
  n
  a
  글자 수 = 6
  l
  e
  m
  o
  n
  글자 수 = 5
  t
  o
  m
  a
  t
  o
  글자 수 = 6
```



## 연습문제 3

---

- for문을 사용하여,  
#1, 문자 "\*"를 첫 줄에는 한 개, 두번째 줄에는 두개, 반복하여 열 번째 줄에는 열 개를 출력 하시오  
  
#2, 문자 "\*"를 첫 줄에 열 개, 둘째 줄에 아홉 개..이어서 열 번째 줄에 한 개를 출력 하시오

## 연습문제 3-코드

```
# 1
# "*" 순차적으로 점점 많이 출력

i = 0

for i in range(1, 11):
    print('*' * i)
```

```

*
**
***
****
*****
*****
*****
*****
*****
*****
*****
```

```
# 2
# "*" 순차적으로 점점 적게 출력

i = 0

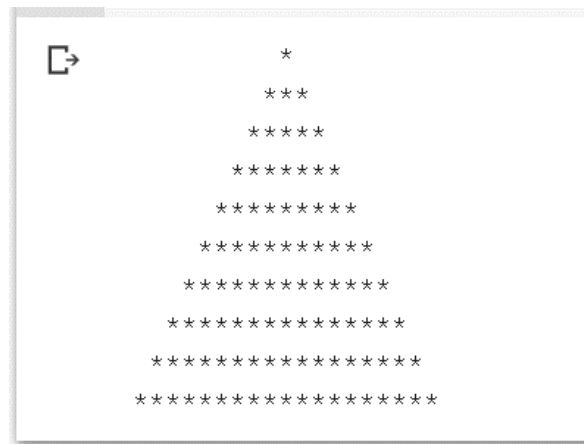
for i in range(10, 0, -1):
    print('*' * i)
```

```

*****
*****
*****
*****
*****
*****
*****
***
**
*
```

## 연습문제 4

- for문을 사용하여,
  - 문자 " ", "\*" 로 다음과 같이 좌우 대칭의 삼각형 모양을 출력해본다



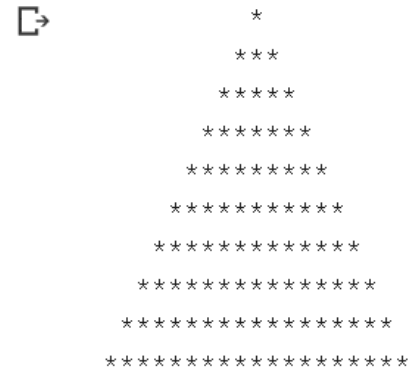
```

      *
     **
    ***
   ****
  *****
 *****
*****
*****
*****
*****
*****
*****
*****
*****

```

## 연습문제 4-코드

```
i=0  
  
for i in range(1, 11):  
    print(' '* (10-i), '*'*(i*2-1), ' '* (10-i))
```



```
      *  
     ***  
    *****  
   *********  
  ***********  
 *****  
*****  
*****  
*****  
*****  
*****  
*****  
*****  
*****  
*****
```

## 연습문제 5

---

- 자신의 이름(성 포함)을 문자열 변수 name에 저장하고, for문을 이용하여 이름을 한 글자씩 출력되도록 한다
- 글자수가 모두 몇 글자인지 출력한다
  - 이 때 space는 제외한다

## 연습문제 5-코드

```
# 성과 이름 한 글자씩 출력

name = "Peter Rabbit"
count = 0

for char in name :
    print(char)
    if char != ' ':
        count = count + 1

print( '총 글자 수는 = ', count)
```

```
☞ P
  e
  t
  e
  r

  R
  a
  b
  b
  i
  t

총 글자 수는 = 11
```

## 강의 요약

---

- **for 다양하게 활용하기**
  - 주로 규칙적인 변화가 있는 경우 사용
  - 문자열, 리스트 활용시 사용
- **range, in 이해하기**
  - `for i in range([start], stop[, step])`
    - start 숫자에서 시작, stop 숫자 이전까지 진행
    - step 숫자만큼 변화(default: 1)

## 퀴즈 1

---

다음 코드의 for문은 몇 번 실행 되는가?

```
for i in range(7, 50, 7) :  
    print(i)
```

- ① 6번
- ② 7번
- ③ 43번
- ④ 50번



## 퀴즈 1-답안

다음 코드의 for문은 몇 번 실행 되는가?

```
for i in range(7, 50, 7) :  
    print(i)
```

- ① 6번
- ② 7번
- ③ 43번
- ④ 50번

```
7  
14  
21  
28  
35  
42  
49
```

## 퀴즈 2

---

다음 문장의 기술이 적절하지 않은 것은?

- ① `range(1,5)`는 `[1, 2, 3, 4, 5]` 값이 나타난다
- ② `range(1,5)`는 `[1, 2, 3, 4]` 값이 나타난다
- ③ `range(1,15,4)`는 `[1, 5, 9, 13]` 값이 나타난다
- ④ `range(-1,6,2)`는 `[-1, 1, 3, 5]` 값이 나타난다

## 퀴즈 2-답안

---

다음 문장의 기술이 적절하지 않은 것은?

- ① `range(1,5)`는 `[1, 2, 3, 4, 5]` 값이 나타난다
- ② `range(1,5)`는 `[1, 2, 3, 4]` 값이 나타난다
- ③ `range(1,15,4)`는 `[1, 5, 9, 13]` 값이 나타난다
- ④ `range(-1,6,2)`는 `[-1, 1, 3, 5]` 값이 나타난다

# 03. 반복문 while 활용

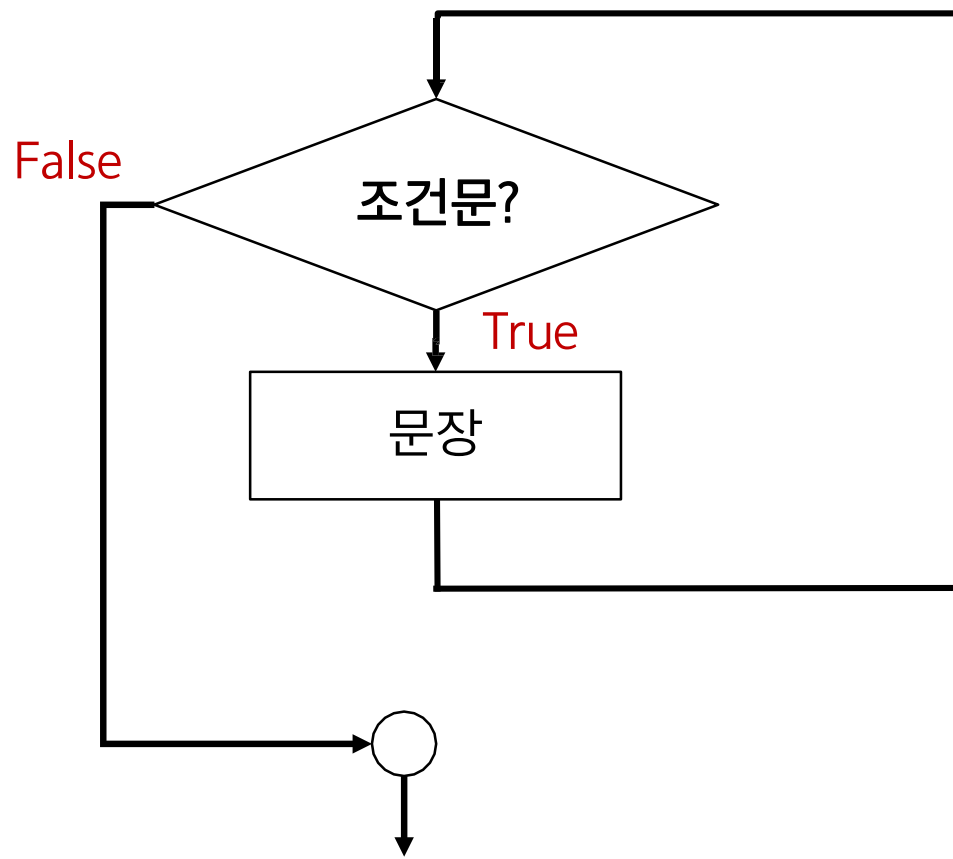
## [2차시]

## 학습목표

---

- while 다양하게 활용하기
- 원하는 과정을 반복하기
- 원하는 값을 입력 받을 때까지 반복하기

# 반복문의 흐름



## 연습문제 1

---

while문을 사용하여 1~100사이의 5의 배수를  
출력하시오

## 연습문제 1-코드

# 1~100사이의 5의 배수

```
i = 1
n = 1
while n < 100:
    n = i * 5
    i = i + 1
    print(n)
```

```
5
10
15
20
25
30
35
40
45
50
55
60
65
70
75
80
85
90
95
100
```



## 연습문제 2

---

while문을 사용하여,

- 입력 받은 값이 99999가 아닌 경우에 while안의 블록을 실행한다
- 블록 내에서는 입력 받은 수 만큼 반복 출력한다

## 연습문제 2-코드

```
# 조건에 따른 while문 실행
num = int(input("숫자를 입력하시오: "))

if num != 99999 :
    i = 0

    while i < num :
        print(num)
        i = i+1

print("num =", num)
```

```
☞ 숫자를 입력하시오: 99999
num = 99999
```

```
☞ 숫자를 입력하시오: 3
3
3
3
num = 3
```

# while문을 사용-원하는 값 입력 받기 1

```
# 사용자가 종료를 요청할 때("y" 입력) 까지 반복하기  
  
quit = "n"  
  
while quit != "y":  
    quit = input("Do you want to quit? ")
```

```
➞ Do you want to quit? n  
Do you want to quit? r  
Do you want to quit? q  
Do you want to quit? y
```

## while문을 사용-원하는 값 입력 받기 2

```
# 맞는 패스워드를 입력 할 때 까지 반복하기
```

```
pwd = "????"
```

```
while pwd != "1234" :
```

```
    pwd = input ("비밀번호를 입력하세요 : ")
```

```
➡ 비밀번호를 입력하세요 : 0000  
비밀번호를 입력하세요 : 0123  
비밀번호를 입력하세요 : 4789  
비밀번호를 입력하세요 : 1234
```

## 비밀번호와 최대 입력 횟수 확인하기

```
# 맞는 패스워드를 입력 할 때 까지 반복하기, 3번 까지 비밀번호 틀리면 반복 입력 기회는 없다
pwd = "????"
count = 1

while pwd != "1234" and count <= 3:
    pwd = input("비밀번호를 입력하세요: ")
    count = count + 1

if pwd == "1234" :
    print("비밀번호가 정확합니다!")
elif count > 3 :
    print("비밀번호 입력 오류가 3번 발생하여, 처리할 수 없습니다")
```

```
➡ 비밀번호를 입력하세요: 0000
비밀번호를 입력하세요: 1200
비밀번호를 입력하세요: 2222
비밀번호 입력 오류가 3번 발생하여, 처리할 수 없습니다
```

## 조건 쓰기

---

- 비밀번호가 맞거나, 입력 횟수가 3보다 크면 종료
  - `pwd == "1234" or count > 3 :`
  - 위의 조건의 반대 조건을 쓰면, 반복절의 조건절
  - `pwd != "1234" and count <= 3 :`
- 드모르간 법칙 사용
  - $(A \text{ and } B)^c \rightarrow A^c \text{ or } B^c$
  - $(A \text{ or } B)^c \rightarrow A^c \text{ and } B^c$

## 연습문제 3.1

# 출력 결과는?

```
s = 1
```

```
n = 1
```

```
while s < 10 :
```

```
    s = s + n
```

```
    print("s =", s)
```

```
➞ s = 2  
s = 3  
s = 4  
s = 5  
s = 6  
s = 7  
s = 8  
s = 9  
s = 10
```

## 연습문제 3.2

# 출력 결과는?

```
x = 1.0
```

```
y = 1.0
```

```
i = 0
```

```
while y <= 1.5 :
```

```
    x = x / 2
```

```
    y = x + y
```

```
    i = i + 1
```

```
    print("x =", x)
```

```
    print("y =", y)
```

```
print("i =", i)
```

```
➡ x = 0.5  
   y = 1.5  
   x = 0.25  
   y = 1.75  
   i = 2
```



## 연습문제 4

---

- 사용자에게 입력 받을 정수의 개수를 입력 받는다  
이 때 음수가 입력되면 양수가 입력 될 때까지 반복 한다
- 만약 5가 입력 되었다면, 5번 정수를 입력 받아서 입력 받은 값의 평균을 출력한다

## 연습문제 4-답안

```
numofnum = 0
count = 1
sum = 0

while numofnum <= 0 :
    numofnum = int(input("정수를 몇개 입력 받을까요? "))

while count <= numofnum :
    num = float(input("숫자를 입력하세요 ; "))
    count = count + 1
    sum = sum + num

print("입력받은 수의 평균은 =", sum / numofnum)
```

☞ 정수를 몇개 입력 받을까요? 4  
숫자를 입력하세요 : 23  
숫자를 입력하세요 : 12  
숫자를 입력하세요 : 58  
숫자를 입력하세요 : 33  
입력받은 수의 평균은 = 31.5

☞ 정수를 몇개 입력 받을까요? -2  
정수를 몇개 입력 받을까요? 1  
숫자를 입력하세요 : 5  
입력받은 수의 평균은 = 5.0

## 연습문제 5

---

- 구매할 물건의 금액을 입력 받는다
- 금액이 만원 이하일 경우에만 반복문을 실행한다
- 반복문이 종료되면 구매할 물건의 개수와 모든 금액의 합계를 출력한다

## 연습문제 5-답안

```
num = 0
count = 0
sum = 0

while num <= 10000 :
    num = float(input("금액을 입력하세요 : "))
    if num <= 10000 :
        sum = sum + num
        count = count +1

print(count, "개 물품의 모든 금액의 합은 = ", sum, "원 입니다")
```

```
➡ 금액을 입력하세요 : 2000
   금액을 입력하세요 : 1000
   금액을 입력하세요 : 5200
   금액을 입력하세요 : 12000
   3 개 물품의 모든 금액의 합은 = 8200.0 원 입니다
```

## 연습문제 6

---

- 이름을 입력 받아서 성이 '김', '최', '이' 이면 통과 아니면 다시 입력 받는다
- 입력 받는 횟수가 5회 초과이면 더 이상 입력 받지 않고 종료한다

## 연습문제 6-답안

```
name = "???"  
count = 1  
  
while name[0] not in ("김", "최", "이") and count <= 5 :  
    name = input ("이름을 입력하세요 : ")  
    count = count +1  
  
if count > 5 :  
    print("5회 까지 입력 가능합니다")  
else :  
    print(" 성이 '김', '최', '이' 중에 있습니다.")
```

➡ 이름을 입력하세요 : 최길동  
성이 '김', '최', '이' 중에 있습니다.

➡ 이름을 입력하세요 : 홍길동  
이름을 입력하세요 : 정길동  
이름을 입력하세요 : 강길동  
이름을 입력하세요 : 구길동  
이름을 입력하세요 : 황길동  
5회 까지 입력 가능합니다

## 강의 요약

---

- while() 다양하게 활용하기
- 원하는 과정을 반복하기
  - ==, !=, <, >, <=, >= 등 활용
- 원하는 값을 입력 받을 때까지 반복하기
  - (입력 값) != (원하는 값)

## 퀴즈 1

---

다음 코드의 while()문은 몇 번 실행되는가?

```
s = 1  
n = 2  
while s < 10 :  
    s = s + n  
    print("s = ", s)
```

- ① 4번
- ② 5번
- ③ 8번
- ④ 9번



## 퀴즈 1-답안

다음 코드의 while()문은 몇 번 실행되는가?

```
s = 1  
n = 2  
while s < 10 :  
    s = s + n  
    print("s = ", s)
```

- ① 4번
- ② 5번
- ③ 8번
- ④ 9번

```
↳ s = 3  
   s = 5  
   s = 7  
   s = 9  
   s = 11
```

## 퀴즈 2

다음 코드의 while문은 몇 번 실행 되는가?

```
pwd = "????"  
while pwd != "1234" :  
    pwd = input ("비밀번호를 입력하세요 : ")
```

- ① 비밀번호 "1234"가 입력 될 때까지
- ② 3번
- ③ 무한반복
- ④ "????"가 입력 될 때까지

## 퀴즈 2-답안

- 다음 코드의 while문은 몇 번 실행 되는가?

```
pwd = "????"  
while pwd != "1234" :  
    pwd = input ("비밀번호를 입력하세요 : ")
```

- ① 비밀번호 "1234"가 입력 될 때까지
- ② 3번
- ③ 무한반복
- ④ "????"가 입력 될 때까지

```
➤ 비밀번호를 입력하세요 : 2312  
비밀번호를 입력하세요 : 231  
비밀번호를 입력하세요 : 1234
```

# 04. 다중 for 활용

## [2차시]

## 학습목표

---

- 다중 for문 이해하기
- break 이해하기
- 다중 for문 활용 연습하기

## 다중 for문

- for문 안에 for문이 사용되는 것
  - 아래 예제와 같이 활용

```
# 리스트 처리  
fruit = ["apple", "banana", "lemon", "tomato"]  
  
for fr in fruit :  
    count = 0  
    for f in fr :  
        print(f)  
        count = count + 1  
  
print("아이템 수 =", count)
```

```
↳ a  
p  
p  
l  
e  
아이템 수 = 5  
b  
a  
n  
a  
n  
a  
아이템 수 = 6  
l  
e  
m  
o  
n  
아이템 수 = 5  
t  
o  
m  
a  
t  
o  
아이템 수 = 6
```

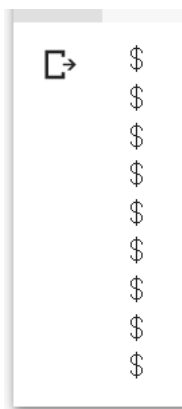
## 다중 for문 실행 과정

변수 fr		변수 f		변수 fr		변수 f	
1번 for	"apple"			1번 for	"lemon"		
		2번 for	"a"			2번 for	"l"
			"p"				"e"
			"p"				"m"
			"l"				"o"
			"e"				"n"
1번 for	"banana"			1번 for	"tomato"		
		2번 for	"b"			2번 for	"t"
			"a"				"o"
			"n"				"m"
			"a"				"a"
			"n"				"t"
			"a"				"o"

## 중첩 for문 (nested for)

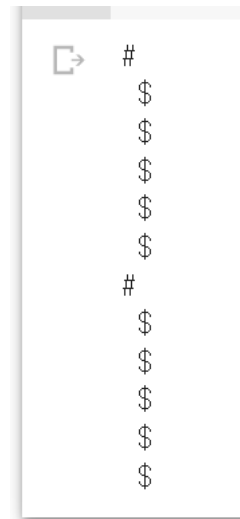
- for문 여러 개 나열

```
for i in range(3):  
    for j in range(3):  
        print ("  $")
```



A terminal window showing the output of the first nested loop. It contains a copy icon at the top left and a vertical column of 9 dollar signs (\$).

```
for i in range(2):  
    print("#")  
    for j in range(5):  
        print ("  $")
```



A terminal window showing the output of the second nested loop. It contains a copy icon at the top left. The output consists of two groups of 5 dollar signs (\$), each preceded by a hash symbol (#) on the same line.



## 중첩 for문 예제

```
for i in range(2,7,2) :
    print ("1st level = ", i)

    for j in range(1,5,1) :
        print(" 2nd level =", j)
        print("   i * j =", i*j)
```

```
↳ 1st level = 2
   2nd level = 1
     i * j = 2
   2nd level = 2
     i * j = 4
   2nd level = 3
     i * j = 6
   2nd level = 4
     i * j = 8
1st level = 4
   2nd level = 1
     i * j = 4
   2nd level = 2
     i * j = 8
   2nd level = 3
     i * j = 12
   2nd level = 4
     i * j = 16
1st level = 6
   2nd level = 1
     i * j = 6
   2nd level = 2
     i * j = 12
   2nd level = 3
     i * j = 18
   2nd level = 4
     i * j = 24
```

# times table : 7, 8

```
for i in range(7, 9) :
    print (i, "th table")

    for j in range(1,10,1) :
        print( "   ", i, "*", j, "=", i*j)
```

```
↳ 7 th table
   7 * 1 = 7
   7 * 2 = 14
   7 * 3 = 21
   7 * 4 = 28
   7 * 5 = 35
   7 * 6 = 42
   7 * 7 = 49
   7 * 8 = 56
   7 * 9 = 63
8 th table
   8 * 1 = 8
   8 * 2 = 16
   8 * 3 = 24
   8 * 4 = 32
   8 * 5 = 40
   8 * 6 = 48
   8 * 7 = 56
   8 * 8 = 64
   8 * 9 = 72
```

## 무엇이 출력 될까?

---

```
# a의 값은 무엇인가?
```

```
a=0
```

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

```
    a=a+1
```

```
for j in range(10) :
```

```
    a=a+1
```

```
print(a)
```

## 무엇이 출력 될까?-답안

```
# a의 값은 무엇인가?
```

```
a=0
```

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

```
    a=a+1
```

```
for j in range(10) :
```

```
    a=a+1
```

```
print(a)
```

20

## 무엇이 출력 될까?

---

```
# a의 값은 무엇인가?  
a=0  
for i in range(10) :  
    a=a+1  
    for j in range(10) :  
        a=a+1  
  
print(a)
```

## 무엇이 출력 될까?-답안

```
# a의 값은 무엇인가?  
a=0  
for i in range(10) :  
    a=a+1  
    for j in range(10) :  
        a=a+1  
  
print(a)
```

바깥쪽 for문 10회 + 안쪽 for문 100회 = 110회

110

## 무엇이 출력 될까?

---

# 다음은 무엇이 출력되는가?

```
fruit = "banana"
count = 0
for char in fruit :
    if char == 'a' :
        count = count + 1

print(count)
```

## 무엇이 출력 될까?-답안

# 다음은 무엇이 출력되는가?

```
fruit = "banana"
count = 0
for char in fruit :
    if char == 'a' :
        count = count + 1

print(count)
```

문자 'a'를 카운트

3

# 버블 정렬

- 숫자나 문자를 일정한 순서대로 재 정리 하는 것
- 제일 간단한 정렬 방식

[https://www.youtube.com/watch?v=MtcrEhrt\\_K0](https://www.youtube.com/watch?v=MtcrEhrt_K0)

Original List	23	78	45	8	32	56
After pass 1	23	45	8	32	56	78
After pass 2	23	8	32	45	56	78
After pass 3	8	23	32	45	56	78
After pass 4	8	23	32	45	56	78
After pass 5	8	23	32	45	56	78
After pass 6	8	23	32	45	56	78



## 버블 정렬

```
numbers = [ 5, 4, 1, 3, 9, 2]
for i in range(0, 6):
    for j in range(0, 7-i):
        if numbers[j] > numbers[j+1]:
            temp = numbers[j]
            numbers[j] = numbers[j+1]
            numbers[j+1] = temp
```



```
numbers = [ 5, 4, 1, 3, 9, 2]
for i in range(0, len(numbers)):
    for j in range(0, len(numbers)-(i+1)):
        if numbers[j] > numbers[j+1]:
            temp = numbers[j]
            numbers[j] = numbers[j+1]
            numbers[j+1] = temp
    print(numbers)
```

```
[1, 2, 3, 4, 5, 9]
```

## 버블 정렬 예제

```
number01 = int(input("첫번째 수 입력: "))
number02 = int(input("두번째 수 입력: "))
number03 = int(input("세번째 수 입력: "))
number04 = int(input("네번째 수 입력: "))
number05 = int(input("다섯번째 수 입력: "))

numbers = [number01, number02, number03, number04, number05]

for i in range(0, len(numbers)):
    for j in range(0, len(numbers)-(i+1)):
        if numbers[j] > numbers[j+1]:
            temp = numbers[j]
            numbers[j] = numbers[j+1]
            numbers[j+1] = temp

print(numbers)
```

☞ 첫번째 수 입력: 20  
두번째 수 입력: 59  
세번째 수 입력: 30  
네번째 수 입력: 409  
다섯번째 수 입력: 9  
[9, 20, 30, 59, 409]

## 버블 정렬 연습문제

---

- 앞의 예제와 같이 Bubble sort 사용한다
- 이번에는 큰 수부터 나열되도록 한다
- 나열 후 가장 큰 숫자와 가장 작은 숫자의 차를 출력한다

## 버블 정렬 연습문제

```
number01 = int(input("첫번째 수 입력: "))
number02 = int(input("두번째 수 입력: "))
number03 = int(input("세번째 수 입력: "))
number04 = int(input("네번째 수 입력: "))
number05 = int(input("다섯번째 수 입력: "))
```

```
numbers = [number01, number02, number03, number04, number05]
```

```
for i in range(0, len(numbers)):
    for j in range(0, len(numbers)-(i+1)):
        if numbers[j] < numbers[j+1]:
            numbers[j], numbers[j+1] = numbers[j+1], numbers[j]
```

```
print(numbers)
print("큰 숫자와 작은 숫자의 차는 ", int(numbers[0]) - int(numbers[len(numbers)-1]))
```

```
[59, 50, 42, 30, 9]
큰 숫자와 작은 숫자의 차는 50
```

# break문

- 반복문에서 벗어나도록 중단시킴
  - 실행 흐름이 루프 이후, 첫번째 명령문으로 넘어감

```
fruit = "banana"
count = 0

for char in fruit :
    if char == 'a':
        count = count + 1
    elif char == 'n':
        break
    else :
        print(char)

print(count)
```

```
↳ b
  1
```

## break문 예제

- Nested for문에서 사용

#2개 수를 더한 결과가 5의 배수 이면 반복문 종료

```
count = 0

for i in range(5) :
    for j in range(3) :
        if (i + j) % 5 != 0 :
            count = count + 1
            print("i + j =", i+j, "and count =", count)
        else :
            break

print("i + j = ", i+j, "last count: ", count)
```

```
➞ i + j = 1 and count = 1
   i + j = 2 and count = 2
   i + j = 3 and count = 3
   i + j = 2 and count = 4
   i + j = 3 and count = 5
   i + j = 4 and count = 6
   i + j = 3 and count = 7
   i + j = 4 and count = 8
   i + j = 4 and count = 9
   i + j = 5 last count: 9
```

## break문 연습문제

---

- 현재 가지고 있는 금액으로 물건을 몇 개까지 살 수 있는지 계산한다
- 현재 가지고 있는 금액, 사려는 물건의 가격, 원하는 개수를 입력 받는다
- 구매할 때마다 잔액이 출력되게 한다
- 잔액이 부족하면 break를 사용하여 반복을 종료한다

## break문 연습문제-코드

```
balance = int(input("현재 가지고 있는 금액은: "))
cost = int(input("사려고 하는 물건의 가격은: "))
num = int(input("원하는 갯수는: "))

for i in range(1, num+1):
    if balance - cost >= 0 :
        balance = balance - cost
        print("물건을", i, "개 구매합니다. 현재 잔액은", balance, "원 입니다")
    else :
        print("잔액이 부족합니다. 현재 잔액은", balance, "원 입니다")
        break
```

☞ 현재 가지고 있는 금액은: 4000  
사려고 하는 물건의 가격은: 1400  
원하는 갯수는: 3  
물건을 1 개 구매합니다. 현재 잔액은 2600 원 입니다  
물건을 2 개 구매합니다. 현재 잔액은 1200 원 입니다  
잔액이 부족합니다. 현재 잔액은 1200 원 입니다



## 소수 확인 과정

---

- 소수의 정의
  - 양의 정수 중 1과 자기 자신으로만 나누어 지는 수
- 확인 과정
  - 2부터 시작해서 입력 받은 수를 나누기 시작하여,
  - (입력 받은 수 - 1) 까지 나누는 것을 반복한다
  - 그 과정에서 나누어지는 경우가 발생하면, 소수가 아 니고
  - 끝까지 나누어지지 않고 반복문이 끝나면 소수이다
  - 나누어지면, 소수를 확인하는 flag 변수의 값을 False로 처리한다

## 소수 확인

```
# prime number 여부 확인

num = int(input("input a positive integer : "))

prime_yes = True

for i in range(2, num) :
    if num % i == 0:
        prime_yes = False
        break

if prime_yes == True :
    print(num, "is a prime number")
else :
    print(num, "is not a prime number")
```

```
input a positive integer : 12
12 is not a prime number
```

```
input a positive integer : 11
11 is a prime number
```

## 연습문제 1

---

- 영어단어를 입력한다
- 위의 단어에 들어가는 글자 중 하나를 입력한다
- 몇 번째 위치에 그 글자가 있는지 찾는다
- 존재하지 않는 글자인 경우, 없다고 출력한다

## 연습문제 1-코드와 결과

```
word = input("단어를 입력하세요: ")
letter = input("위치를 찾을 한 글자를 입력하세요: ")

count = 0

for char in word :
    count = count + 1
    if char == letter:
        break

if letter not in word:
    print( letter, "는", word, "내에 없습니다")
else :
    print(letter, "는", word, "의", count, "번째 위치합니다")
```

☞ 단어를 입력하세요: Python  
위치를 찾을 한 글자를 입력하세요: o  
o 는 Python 의 5 번째 위치합니다

☞ 단어를 입력하세요: Python  
위치를 찾을 한 글자를 입력하세요: s  
s 는 Python 내에 없습니다

## 연습문제 2

---

- 사용자에게 메시지를 입력 받는다
- 메시지 내에서 'a' 를 개수를 확인하여 출력한다

## 연습문제 2-코드와 결과

문자열에서 한 글자씩 읽어서 그 글자가 'a' 인지를 확인하여 센다

```
s = input("Input message : ")
count = 0

for c in s :
    if c == "a" :
        count += 1

print("count of 'a' : ", count)
```

```
Input message : apple and banana
count of 'a' : 5
```

## 연습문제 3

아래와 같은 구구단을 출력하는 코드 작성

```

===== 구구단 =====
      2단      3단      4단      5단      6단      7단      8단      9단
2 * 1 = 2  3 * 1 = 3  4 * 1 = 4  5 * 1 = 5  6 * 1 = 6  7 * 1 = 7  8 * 1 = 8  9 * 1 = 9
2 * 2 = 4  3 * 2 = 6  4 * 2 = 8  5 * 2 = 10 6 * 2 = 12 7 * 2 = 14 8 * 2 = 16 9 * 2 = 18
2 * 3 = 6  3 * 3 = 9  4 * 3 = 12 5 * 3 = 15 6 * 3 = 18 7 * 3 = 21 8 * 3 = 24 9 * 3 = 27
2 * 4 = 8  3 * 4 = 12 4 * 4 = 16 5 * 4 = 20 6 * 4 = 24 7 * 4 = 28 8 * 4 = 32 9 * 4 = 36
2 * 5 = 10 3 * 5 = 15 4 * 5 = 20 5 * 5 = 25 6 * 5 = 30 7 * 5 = 35 8 * 5 = 40 9 * 5 = 45
2 * 6 = 12 3 * 6 = 18 4 * 6 = 24 5 * 6 = 30 6 * 6 = 36 7 * 6 = 42 8 * 6 = 48 9 * 6 = 54
2 * 7 = 14 3 * 7 = 21 4 * 7 = 28 5 * 7 = 35 6 * 7 = 42 7 * 7 = 49 8 * 7 = 56 9 * 7 = 63
2 * 8 = 16 3 * 8 = 24 4 * 8 = 32 5 * 8 = 40 6 * 8 = 48 7 * 8 = 56 8 * 8 = 64 9 * 8 = 72
2 * 9 = 18 3 * 9 = 27 4 * 9 = 36 5 * 9 = 45 6 * 9 = 54 7 * 9 = 63 8 * 9 = 72 9 * 9 = 81
=====
  
```

## 연습문제 3-코드와 결과

```
print("="*43, "구구단", "="*43)
for i in range(0,10):
    for j in range(2,10):
        if i==0:
            print(f" {j:d}단 ", end=" ")
        else:
            print(f"{j:d} * {i:d} = {i*j:2d}", end=" ")
    print("")
print("="*(43+8+43))
```

```

===== 구구단 =====
      2단      3단      4단      5단      6단      7단      8단      9단
2 * 1 = 2  3 * 1 = 3  4 * 1 = 4  5 * 1 = 5  6 * 1 = 6  7 * 1 = 7  8 * 1 = 8  9 * 1 = 9
2 * 2 = 4  3 * 2 = 6  4 * 2 = 8  5 * 2 = 10 6 * 2 = 12 7 * 2 = 14 8 * 2 = 16 9 * 2 = 18
2 * 3 = 6  3 * 3 = 9  4 * 3 = 12 5 * 3 = 15 6 * 3 = 18 7 * 3 = 21 8 * 3 = 24 9 * 3 = 27
2 * 4 = 8  3 * 4 = 12 4 * 4 = 16 5 * 4 = 20 6 * 4 = 24 7 * 4 = 28 8 * 4 = 32 9 * 4 = 36
2 * 5 = 10 3 * 5 = 15 4 * 5 = 20 5 * 5 = 25 6 * 5 = 30 7 * 5 = 35 8 * 5 = 40 9 * 5 = 45
2 * 6 = 12 3 * 6 = 18 4 * 6 = 24 5 * 6 = 30 6 * 6 = 36 7 * 6 = 42 8 * 6 = 48 9 * 6 = 54
2 * 7 = 14 3 * 7 = 21 4 * 7 = 28 5 * 7 = 35 6 * 7 = 42 7 * 7 = 49 8 * 7 = 56 9 * 7 = 63
2 * 8 = 16 3 * 8 = 24 4 * 8 = 32 5 * 8 = 40 6 * 8 = 48 7 * 8 = 56 8 * 8 = 64 9 * 8 = 72
2 * 9 = 18 3 * 9 = 27 4 * 9 = 36 5 * 9 = 45 6 * 9 = 54 7 * 9 = 63 8 * 9 = 72 9 * 9 = 81
=====
```



## 연습문제 4

---

- 별의 개수를 입력받아
- 뒤집어진 삼각형 모양으로 출력하기

## 연습문제 4-코드와 결과

```
num = int(input("별의 개수를 입력하세요: "))  
  
for i in range(0, num):  
    for j in range(num - i):  
        print("*", end="") # 다음줄로 출력되지 않게 함  
    print() # 다음줄로 출력
```

```
➞ 별의 개수를 입력하세요: 12  
*****  
*****  
*****  
*****  
*****  
*****  
*****  
*****  
*****  
*****  
*****  
*****
```

## 강의 요약

---

- 다중 for문 이해하기
  - for문 안에 for문이 사용되는 것
- break 이해하기
  - 반복문에서 벗어나도록 중단시킴
- 다중 for문 활용 연습하기

## 퀴즈 1

---

다음 for문 안의 print문은 몇 번 출력되는가?

```
for i in range(2) :  
    for j in range(7) :  
        print ("  $")
```

- ① 2번
- ② 7번
- ③ 9번
- ④ 14번

## 퀴즈 1-답안

다음 for문 안의 print문은 몇 번 출력되는가?

```
for i in range(2) :  
    for j in range(7) :  
        print ("  $")
```

- ① 2번
- ② 7번
- ③ 9번
- ④ 14번



## 퀴즈 2

---

다음 코드의 결과 값은?

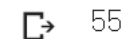
```
a=0
for i in range(5) :
    a=a+1
    for j in range(5) :
        a=a+2
print(a)
```

- ① 50
- ② 55
- ③ 60
- ④ 65

## 퀴즈 2-답안

다음 코드의 결과 값은?

```
a=0
for i in range(5) :
    a=a+1
    for j in range(5) :
        a=a+2
print(a)
```

55

- ① 50
- ② 55
- ③ 60
- ④ 65

{바깥쪽 for문 5회}\*1 + {안쪽 for문 25회}\*2 = 55

# 05. turtle 이해하기

## [2차시]



# 학습목표

---

- Module turtle 사용 용도 이해하기
- Module 사용하기
- Turtle 명령어 활용하기

# turtle 모듈

---

- turtle 모듈은
  - 객체 지향적 그리고 절차 지향적 방법으로 그래픽스를 표현할 수 있도록 지원함
  - 간단한 그래픽 처리 가능
  - 다양한 그래픽 활용은
    - 'Tkinter'를 사용
- turtle 그래픽스는 프로그래밍을 시작할 때 적합
  - 처음 프로그램을 하는 사람에게 흥미 유발

# turtle 모듈

- colab에서 모듈을 사용하려면
  - 모듈(ColabTurtlePlus)을 설치해 줘야함

```
!pip install ColabTurtlePlus
```

```
↳ Collecting ColabTurtlePlus
  Downloading ColabTurtlePlus-2.0.1-py3-none-any.whl (31 kB)
  Installing collected packages: ColabTurtlePlus
  Successfully installed ColabTurtlePlus-2.0.1
```

- 설치된 모듈을 import해서 사용

```
import ColabTurtlePlus.Turtle as t
t.clearscreen()
또는
from ColabTurtlePlus.Turtle import *
clearscreen()
```

# turtle 예제 1

```
t.clearscreen()  
t.forward(150)  
t.pos()
```

C+



(150.0, 0.0)

```
t.clearscreen()  
  
t.forward(150)  
t.left(90)  
t.forward(200)  
t.pos()
```

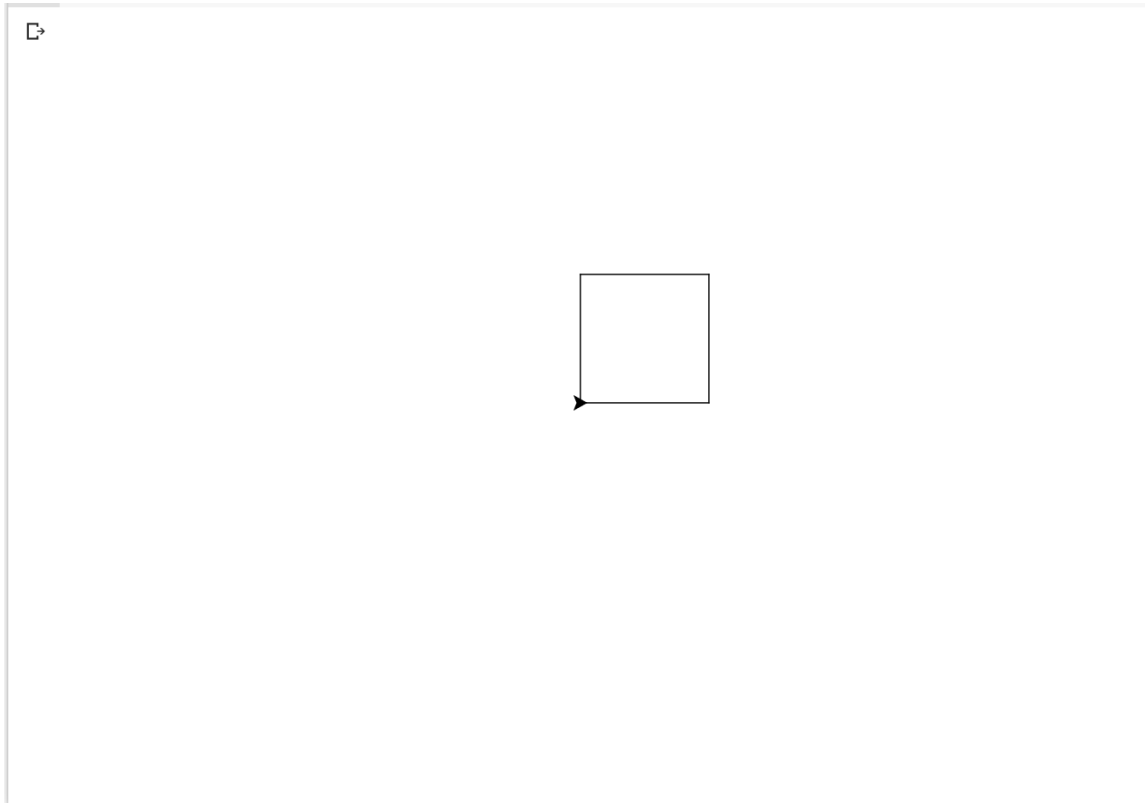
C+



(150.0, 200.0)

## turtle 예제 2

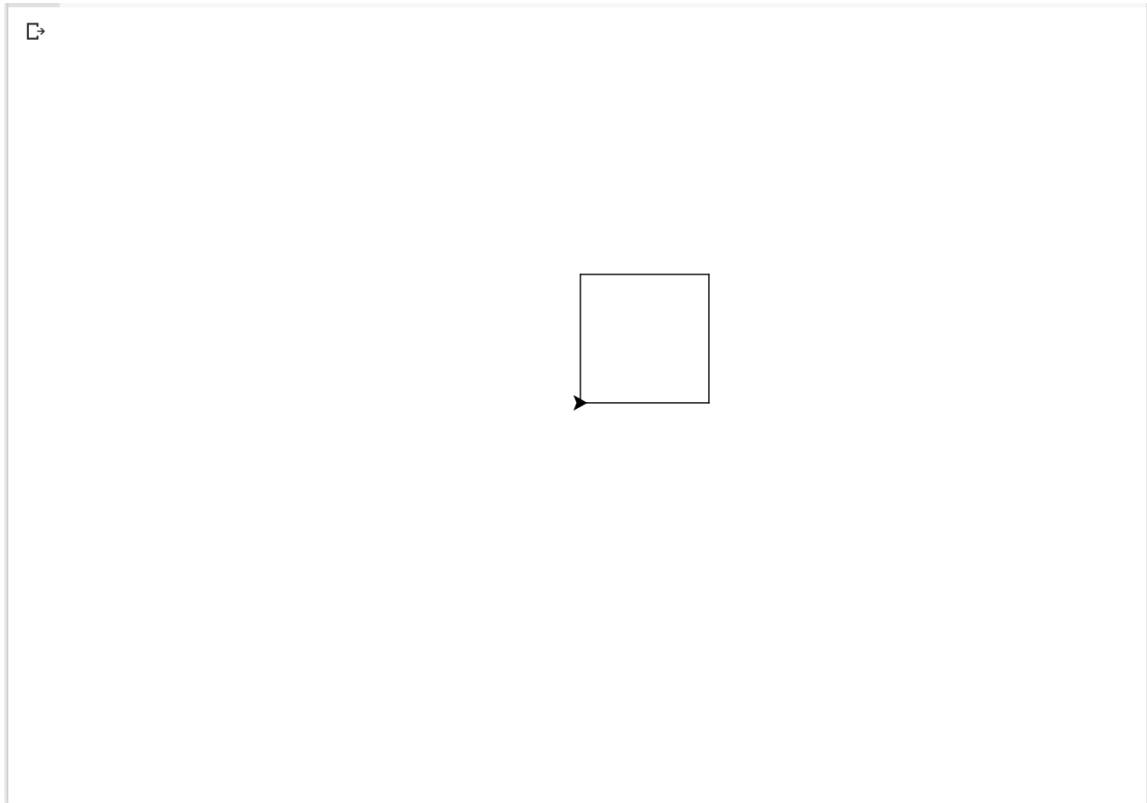
```
t.clearscreen()  
  
t.forward(100)  
t.left(90)  
t.forward(100)  
t.left(90)  
t.forward(100)  
t.left(90)  
t.forward(100)  
t.left(90)
```



## turtle 예제 3

```
t.clearscreen()
```

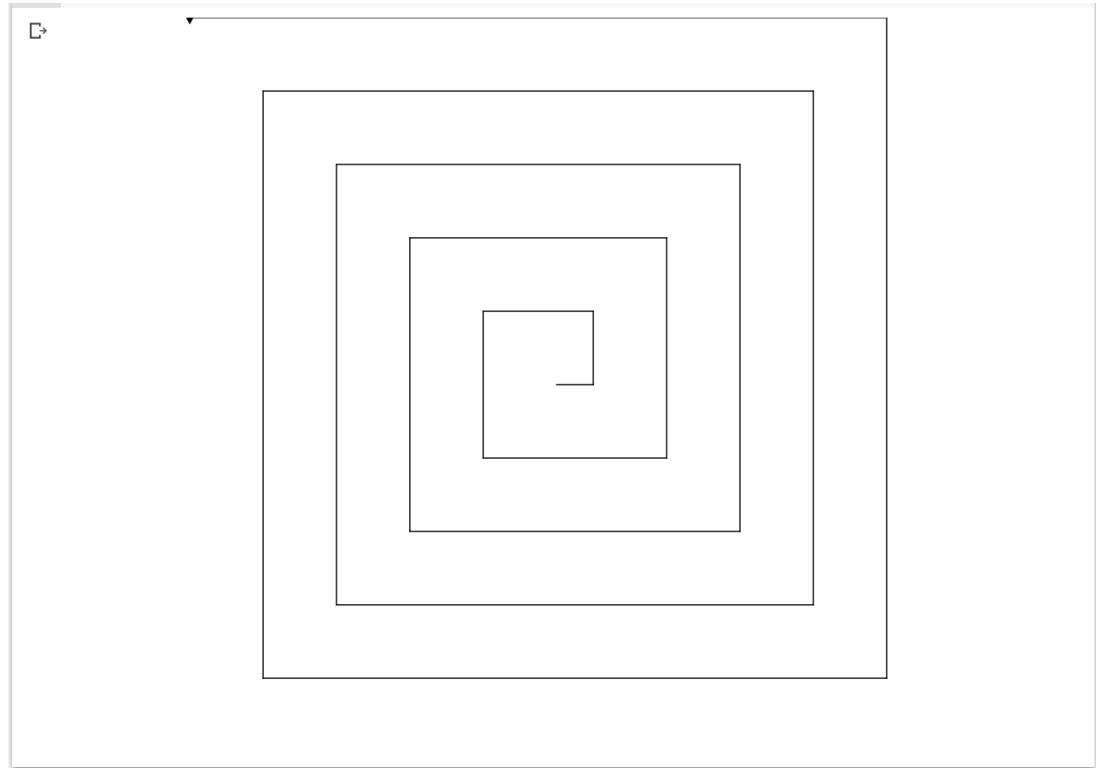
```
for i in range(4) :  
    t.forward(100)  
    t.left(90)
```



## turtle 예제 4

```
t.clearscreen()
```

```
for i in range(30, 600, 30) :  
    t.forward(i)  
    t.left(90)
```



# turtle 메소드

forward(n)	Turtle을 앞으로 n 만큼 이동시킴	begin_fill() ...	도형을 현재 색으로 채움
left(n)	Turtle을 왼쪽으로 n 도 회전시킴	xcor(), ycor()	현재 x와 y좌표 값을 반환
right(n)	Turtle을 오른쪽으로 n 도 회전시킴	setx(a), sety(b)	X와 y 좌표로 각각 값을 설정
penup()	pen up (그리기를 멈춤)	goto(x,y)	주어진 (x,y) 좌표로 이동
pendown()	pen down (그리기를 시작)	write( ... )	write("Hello",False,align="center", font=("Times ",20,"bold"))
color(c)	현재 색을 설정함	circle(r)	반지름이 r인 원을 그림 circle(50,180) # 호를 그릴때도 circle을 활용
width(n)	펜의 굵기를 결정	ht()	Turtle을 숨김
speed(s)	움직임의 속도를 지정(0~13 사이) 최저 1, 최고 13이며 0으로 지정하면 애니메이션 없음.	done()	이미지를 새로 그림. 속도가 0일 경우 이미지 표시를 위해 지정이 필요함

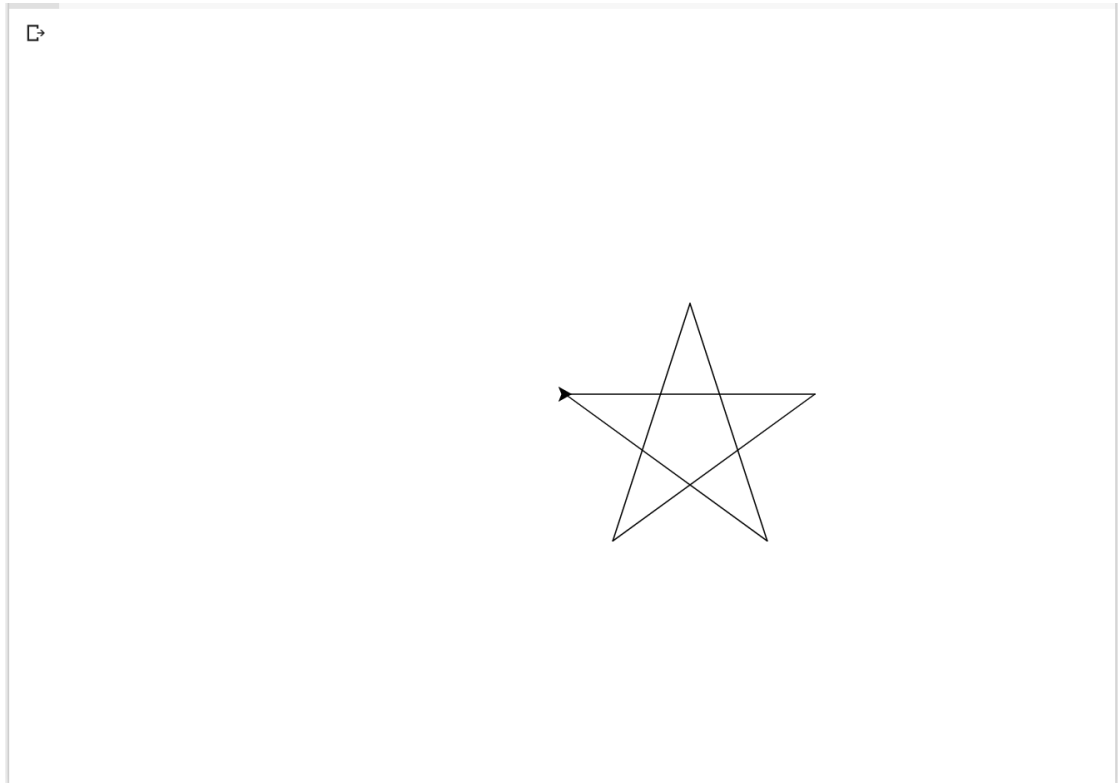
참고 : <https://github.com/mathriddle/ColabTurtlePlus>



# 선으로 별 모양 그리기

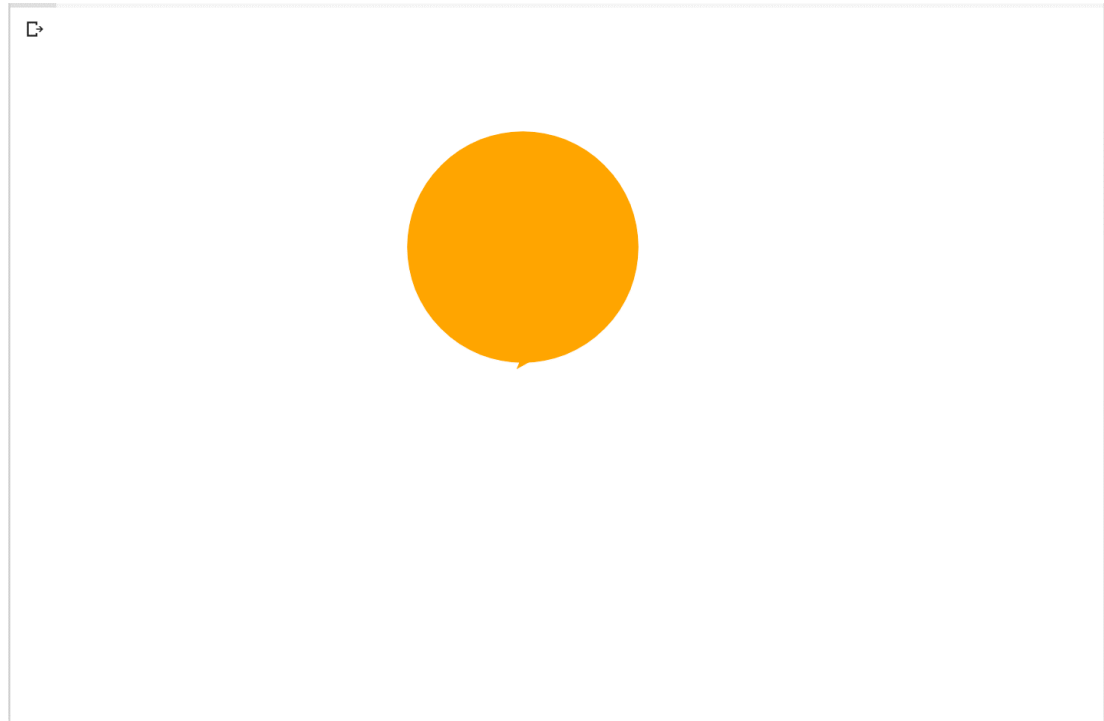
```
t.clearscreen()

for i in range(5) :
    t.forward(200)
    t.left(144)
```

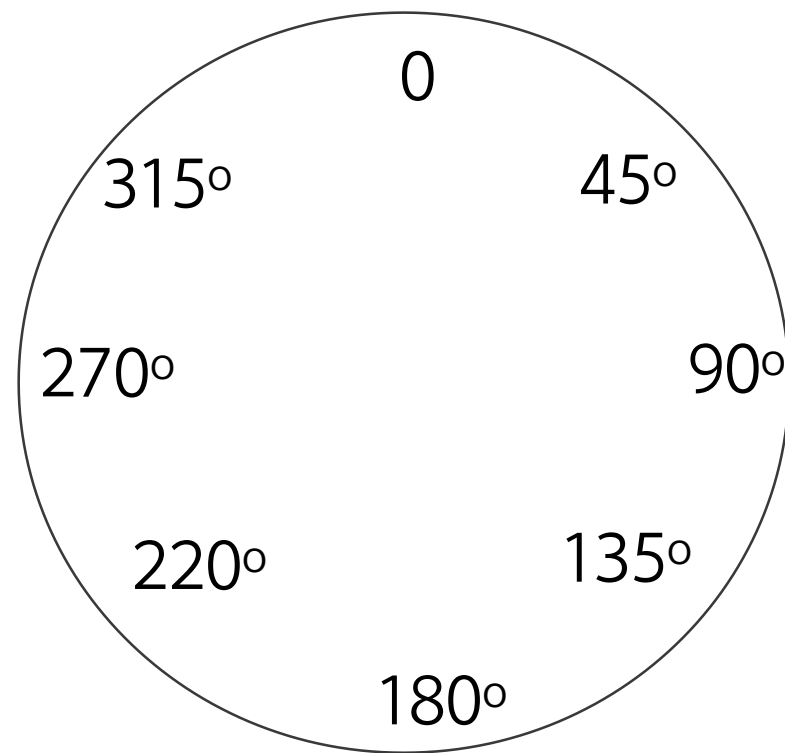
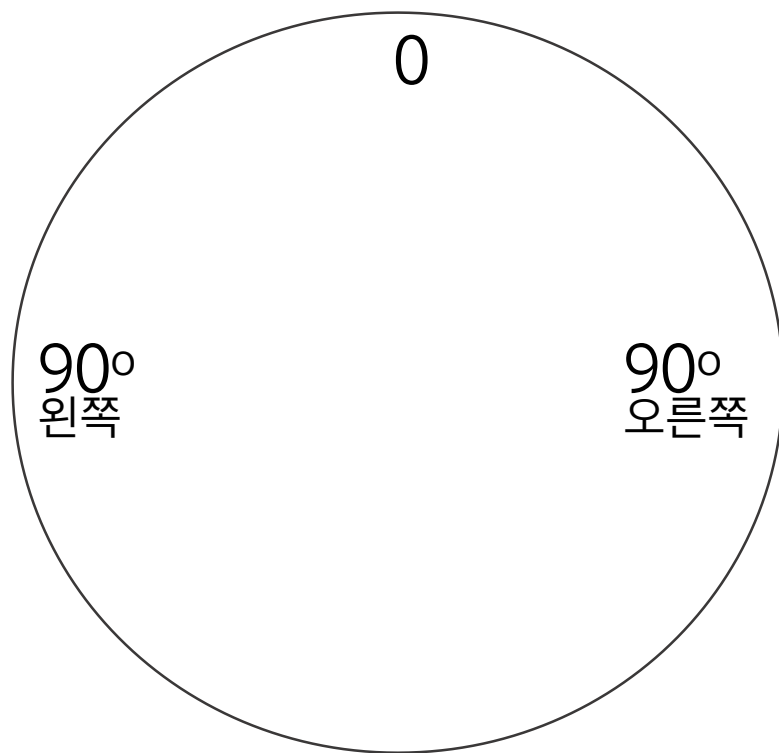


# 색상있는 원 그리기

```
t.clearscreen()  
  
t.color("orange")  
t.begin_fill()  
t.circle(100)  
t.end_fill()
```



## 위치, 각도 사용시



# 선 그리기

## ■ 배경색 및 펜 굵기

```
t.clearscreen()
```

```
t.bgcolor("lightgreen")
```

```
t.color("hotpink")  
t.pensize(3)
```

```
t.forward(200)  
t.right(120)  
t.forward(200)
```



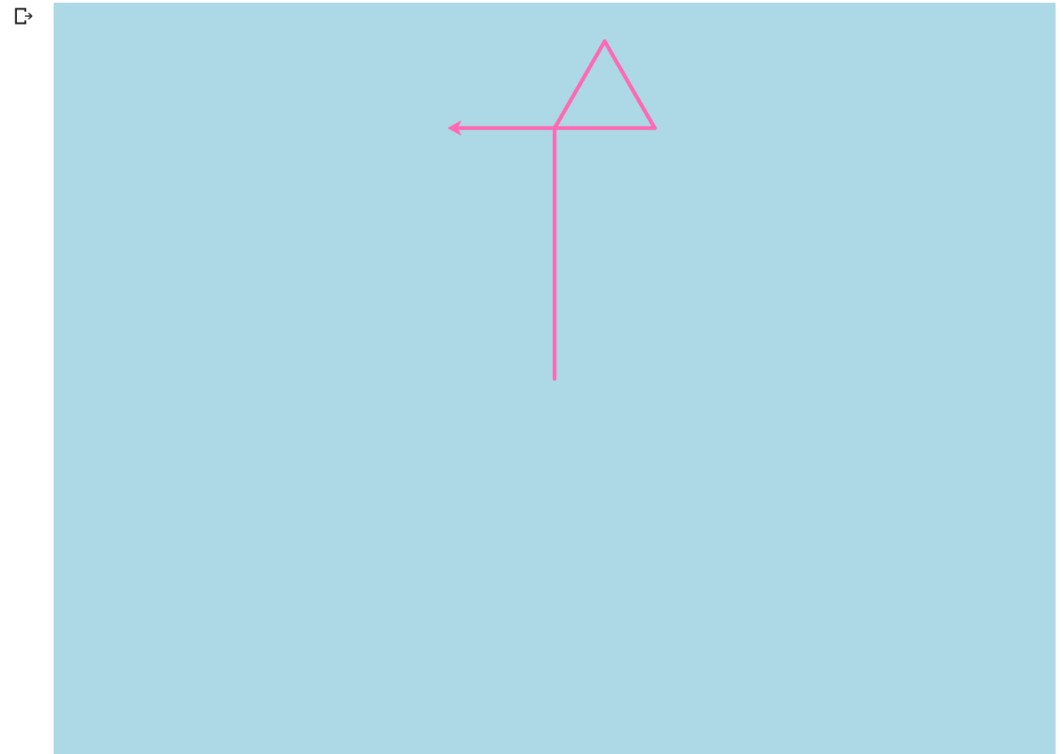
# 삼각형 그리기

```
t.clearscreen()  
t.bgcolor("lightblue")
```

```
t.color("hotpink")  
t.pensize(3)
```

```
t.goto(0, 200)  
for i in range(3):  
    t.forward(80)  
    t.left(120)
```

```
t.right(180)  
t.forward(80)
```



# 삼각형 그리기, 이동선 제거

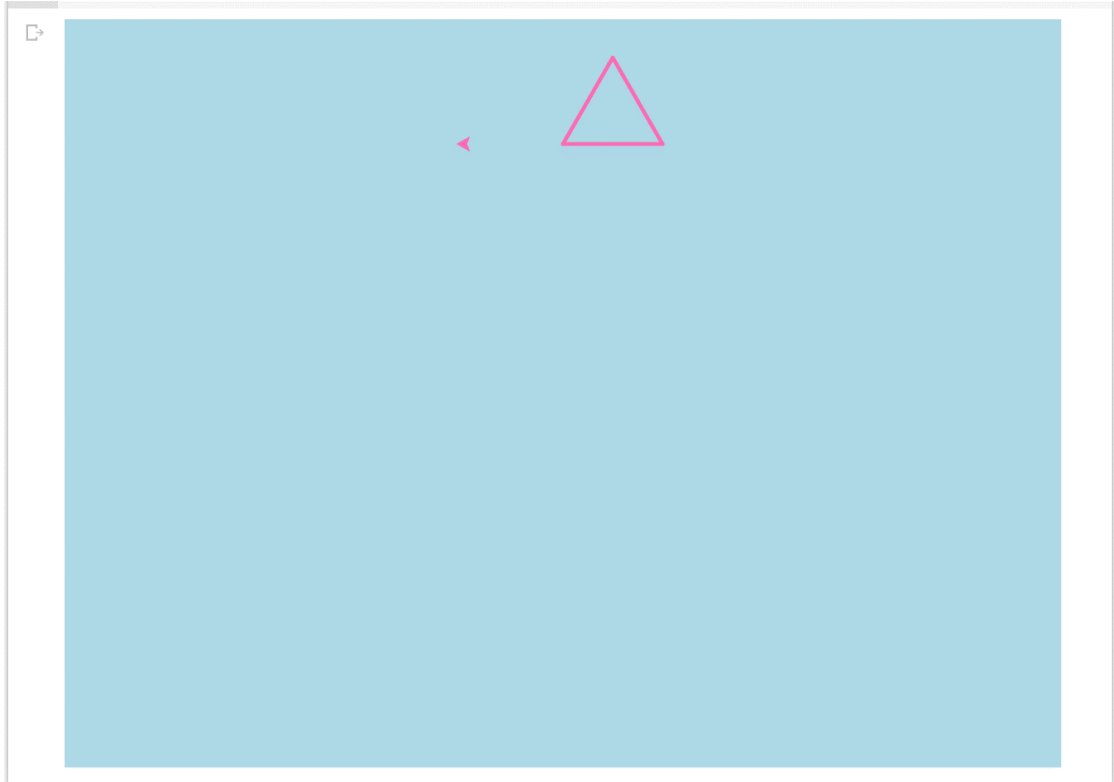
```
t.clearscreen()  
t.bgcolor("lightblue")
```

```
t.color("hotpink")  
t.pensize(3)
```

```
t.penup()  
t.goto(0, 200)  
t.pendown()
```

```
for i in range(3):  
    t.forward(80)  
    t.left(120)
```

```
t.penup()  
t.right(180)  
t.forward(80)
```



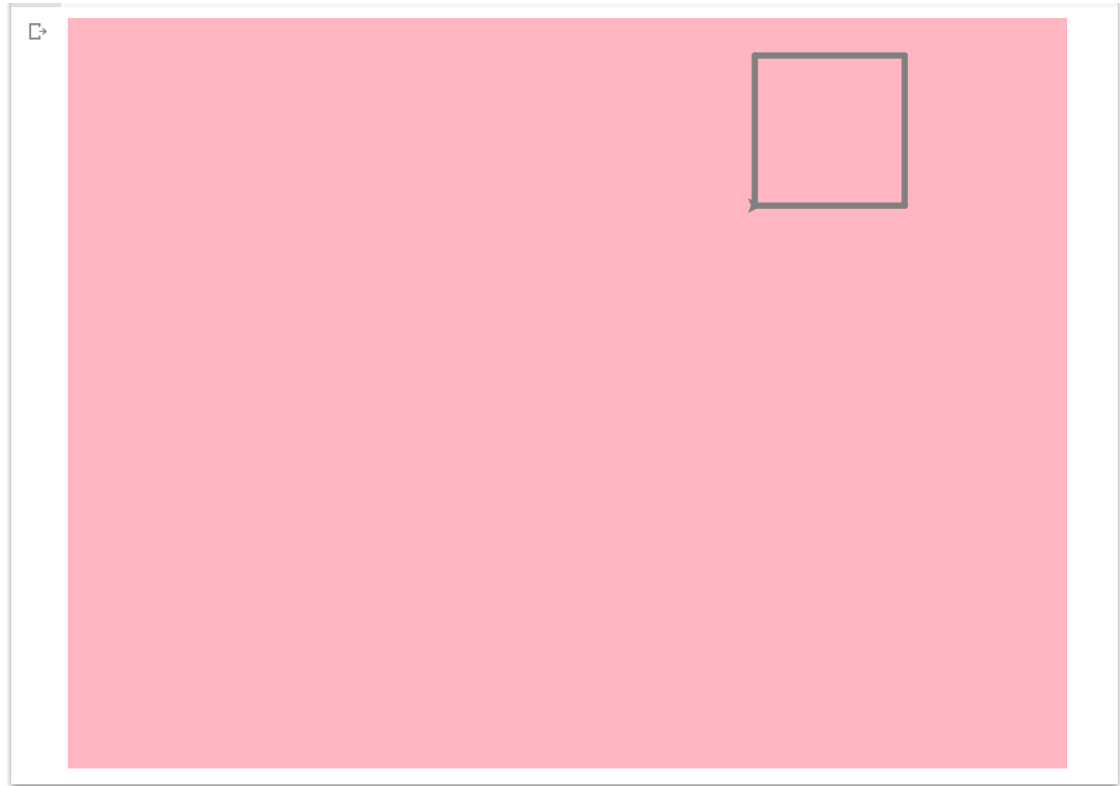
# 사각형 그리기

```
t.clearscreen()  
t.bgcolor("lightpink")
```

```
t.color("grey")  
t.pensize(5)
```

```
t.penup()  
t.goto(150, 150)  
t.pendown()
```

```
for i in range(4):  
    t.forward(120)  
    t.left(90)
```



## 강의 요약

---

- Module turtle 사용 용도
  - 객체 지향적, 절차 지향적으로 그래픽스 표현 지원
  - 간단한 그래픽 처리 가능
- Module 사용하기
  - `import turtle`
- Turtle 명령어 활용하기
  - `.forward(n)`, `.left(n)`, `.right(n)`, `.color(n)`, `.penup()`, ...



# 06. turtle 활용하기

## [2차시]

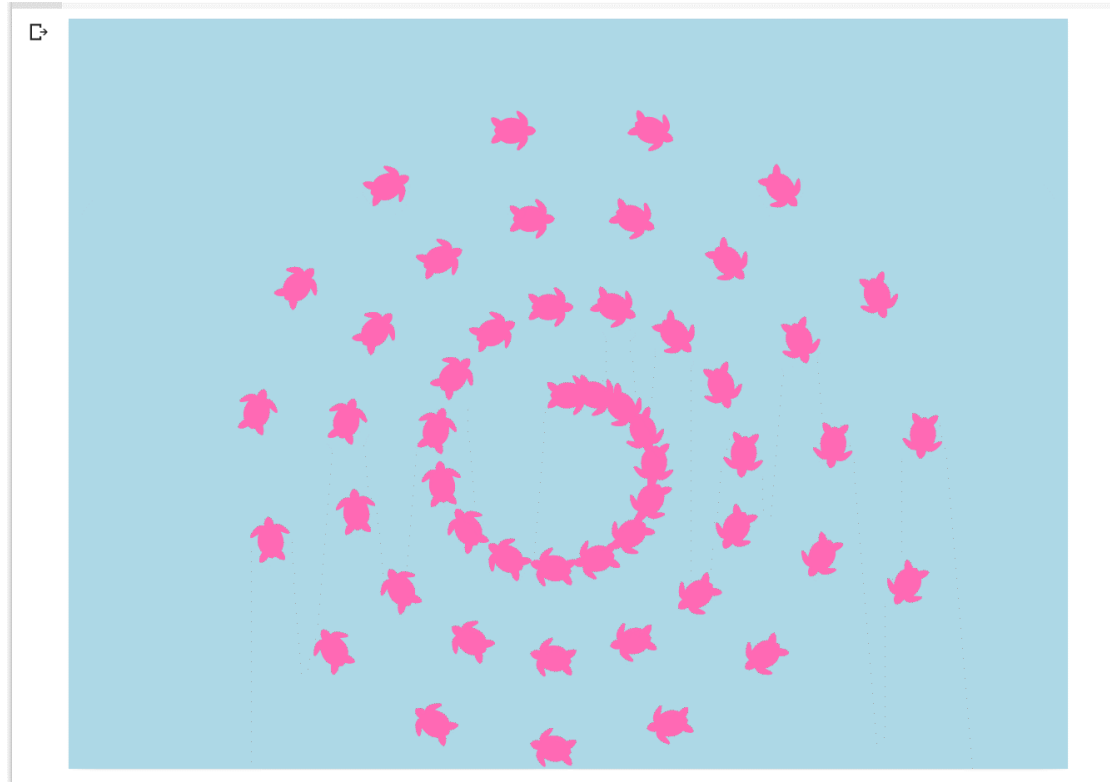
# 미로그리기

```
t.clearscreen()

t.speed(13)
t.bgcolor("lightblue")
t.shape("turtle")
t.color("hotpink")

t.penup()
size = 20

for i in range(50):
    t.stamp()
    size = size + 2
    t.forward(size)
    t.right(24)
```



## 정사각형 36개 배치

```
t.clearscreen()
```

```
t.speed(13)
```

```
t.color("blue")
```

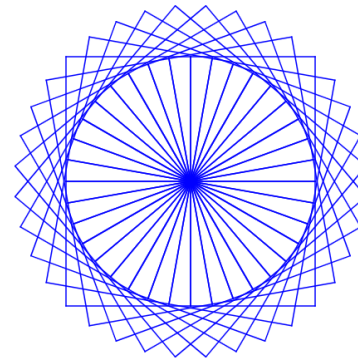
```
for i in range(36):
```

```
    t.left(10)
```

```
    for j in range(4):
```

```
        t.forward(100)
```

```
        t.right(90)
```

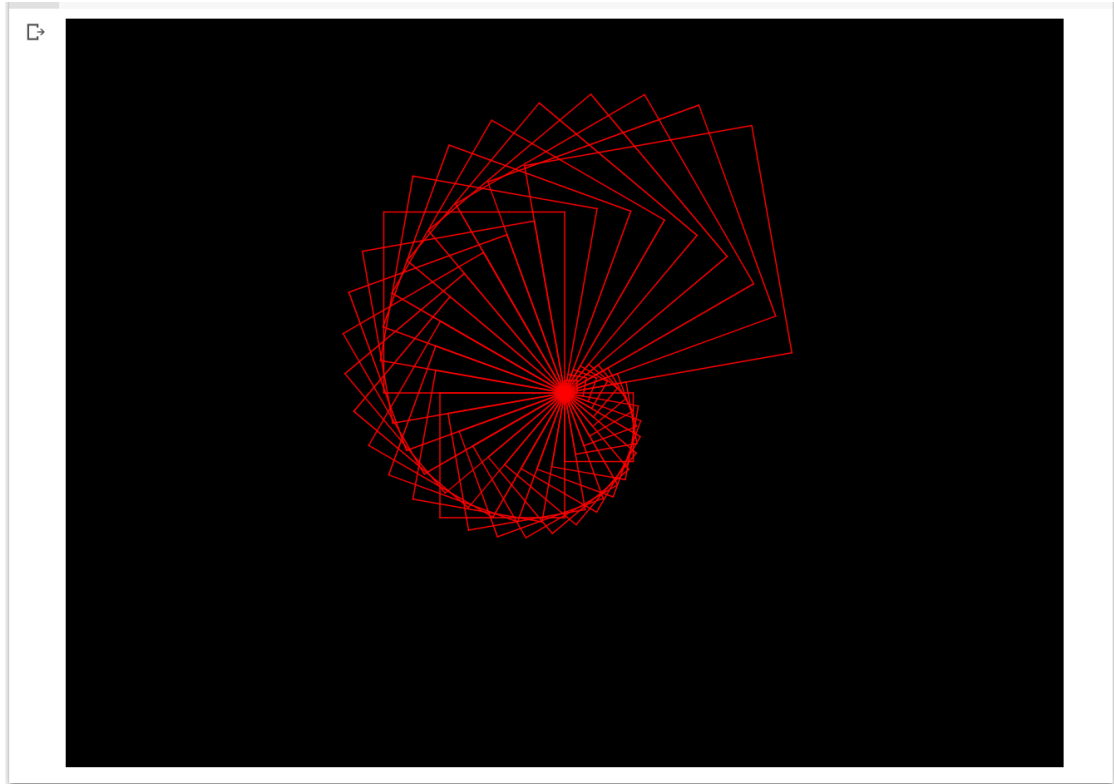


## 점점 커지는 정사각형 36개

```
t.clearscreen()

t.speed(13)
t.bgcolor("black")
t.color("red")

for i in range(36):
    t.forward(10+i*5)
    t.left(90)
    t.forward(10+i*5)
    t.left(90)
    t.forward(10+i*5)
    t.left(90)
    t.forward(10+i*5)
    t.left(80)
```

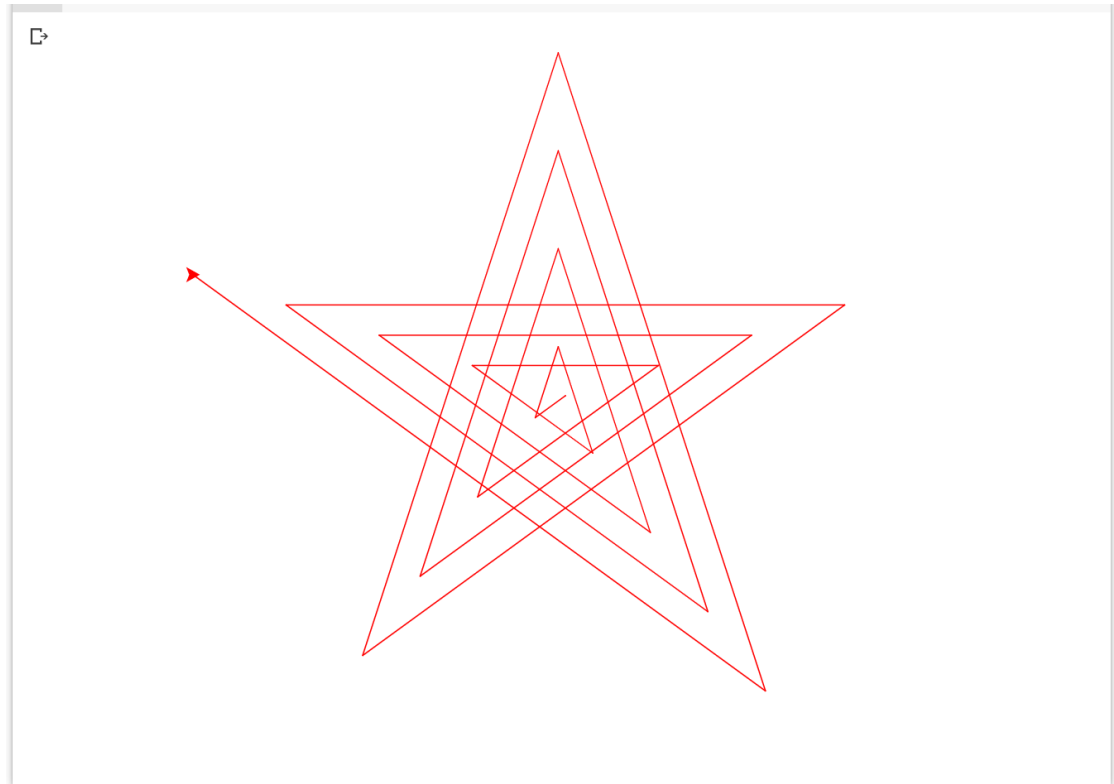


# 점점 커지는 별

```
t.clearscreen()

t.speed(13)
t.color("red")

for i in range(20):
    t.forward(i*30)
    t.right(144)
```

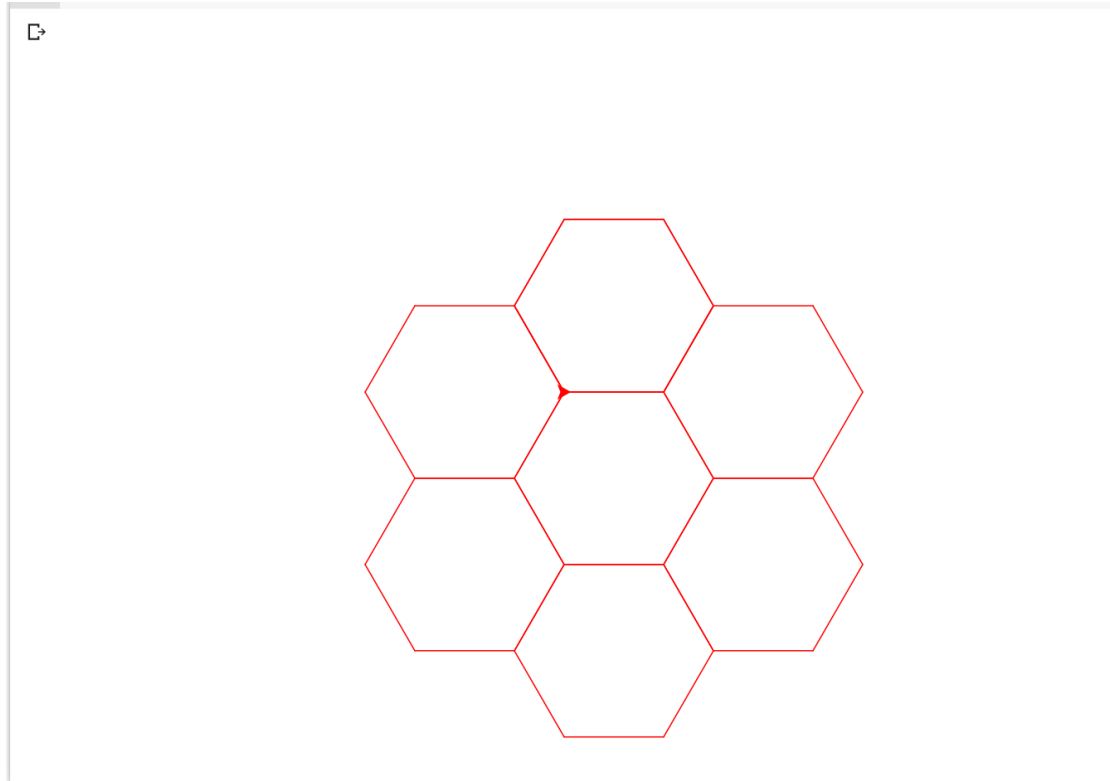


# 벌집 그리기, 함수

```
def hexagon(t):  
    for i in range(6):  
        t.forward(80)  
        t.left(60)
```

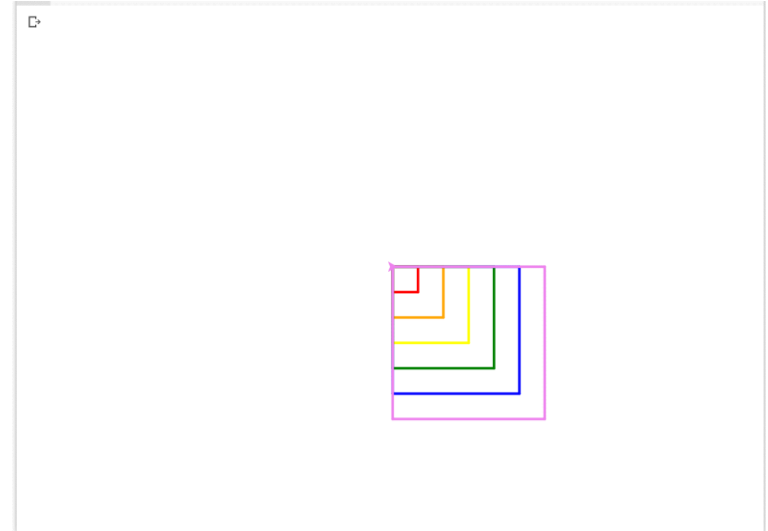
```
t.clearscreen()  
t.speed(13)  
t.color('red')  
hexagon(t)
```

```
for i in range(6):  
    hexagon(t)  
    t.forward(80)  
    t.right(60)
```



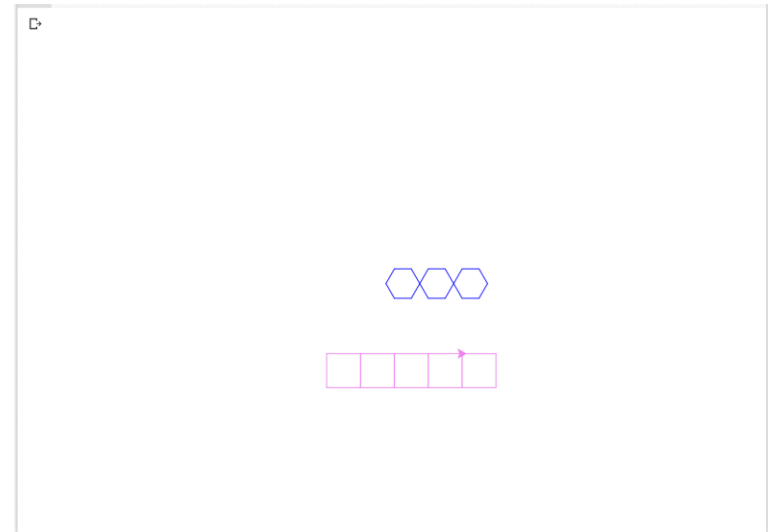
## 여러가지 색 정사각형 그리기, 함수

```
def square(t, size, color):  
    t.color(color)  
    for i in range(4):  
        t.forward(size)  
        t.right(90)  
  
t.clearscreen()  
t.speed(13)  
t.pensize(3)  
colors = ['red', 'orange', 'yellow', 'green',  
          'blue', 'violet']  
  
i = 30  
for color in colors:  
    square(t, i, color)  
    i = i + 30
```



## 다각형 그리기, 함수

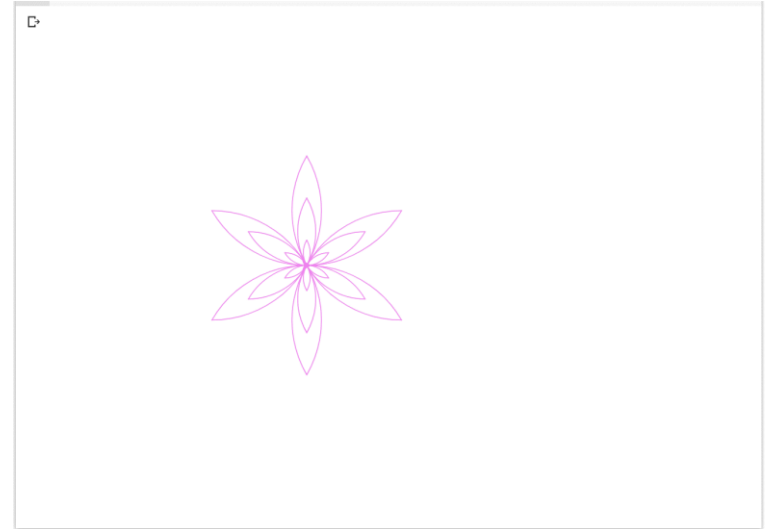
```
def drawPolygon(t, sideLength, numSides, color):  
    t.color(color)  
    turnAngle= 360 / numSides  
    for i in range(numSides):  
        t.pendown()  
        t.forward(sideLength)  
        t.right(turnAngle)  
  
t.clearscreen()  
t.speed(13)  
for i in range(3):  
    t.penup()  
    t.setposition(40*i, 0)  
    drawPolygon(t, 20, 6, "blue")  
  
for i in range(5):  
    t.penup()  
    t.setposition(40*(i-2), -100)  
    drawPolygon(t, 40, 4, "violet")
```





## 꽃 그리기, 함수

```
def flower(t, n, r):  
    angle = 360/n  
    for i in range(n):  
        for i in range(2):  
            t.circle(r,angle)  
            t.left(180-angle)  
        t.left(angle)  
  
def move(t, length):  
    t.penup()  
    t.forward(length)  
    t.pendown()  
  
t.clearscreen()  
t.speed(13)  
t.color("violet")  
move(t, -100)  
  
for i in range(3):  
    flower(t, 6, 30+(50*i))
```

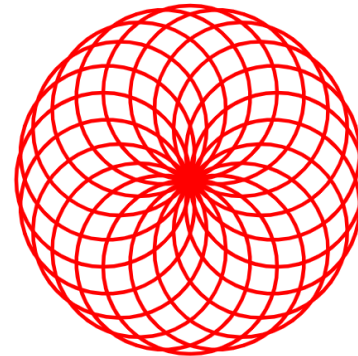


## 여러 개 원 출력, 함수

```
def n_circle(t, n, size):  
    for i in range(n):  
        t.circle(size)  
        t.left(360.0/n)
```

```
t.clearscreen()  
t.speed(13)  
t.color('red')  
t.pensize(3)
```

```
n_circle(t, 20, 70)
```



# 연습문제 1

---

정사각형 5개를 규칙적으로 위치를 바꾸어서 그리기

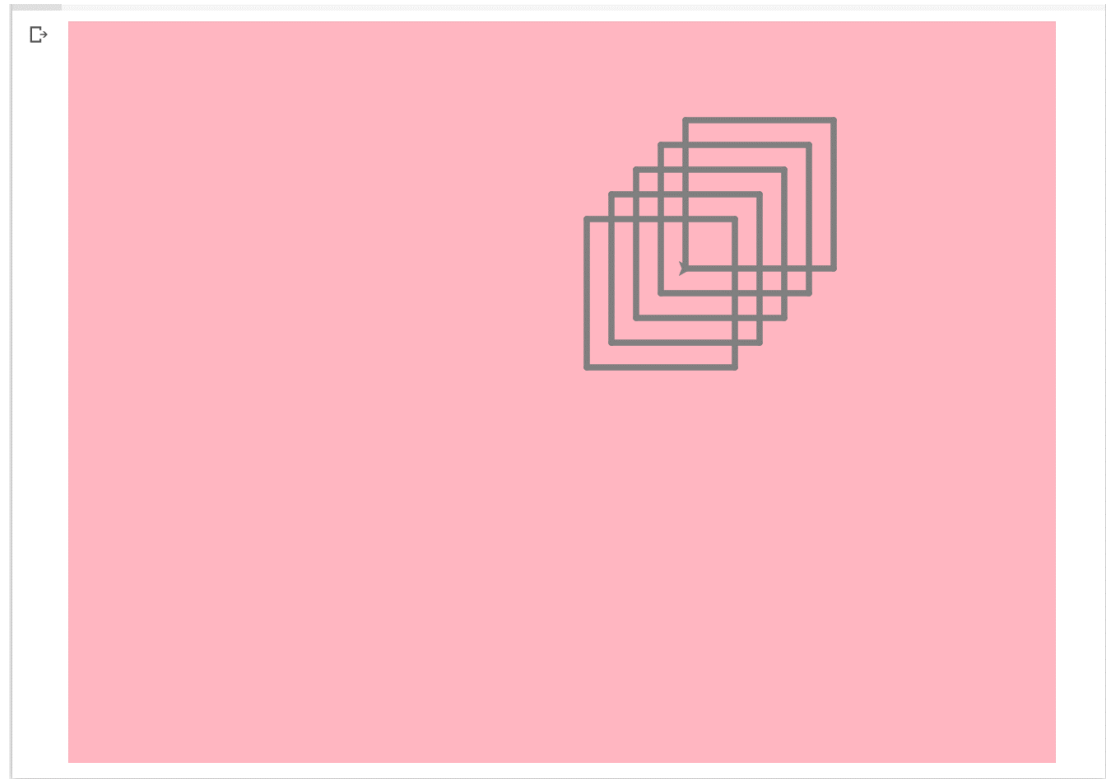
## 연습문제 1-코드와 결과

```
t.clearscreen()

t.speed(13)
t.bgcolor("lightpink")

t.color("grey")
t.pensize(5)

for i in range(20, 110, 20) :
    t.penup()
    t.goto(i,i)
    t.pendown()
    for j in range(4):
        t.forward(120)
        t.left(90)
```



## 연습문제 2

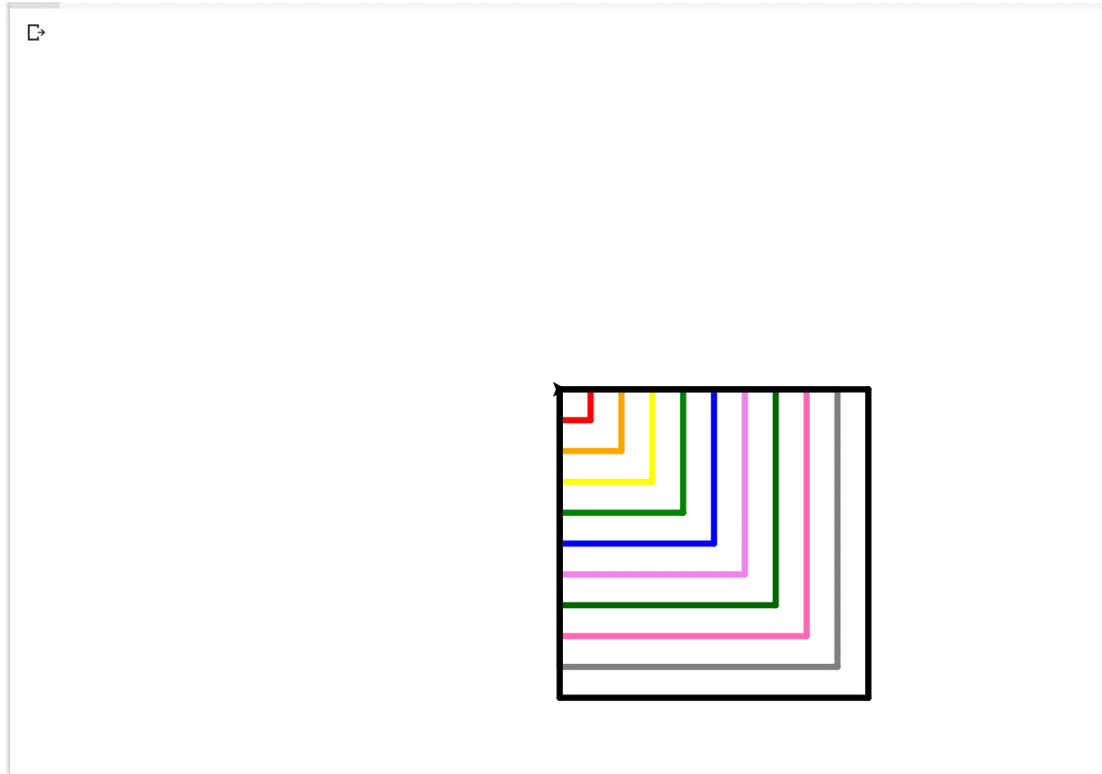
---

여러가지 색 정사각형 그리기 예제 수정

- 색상을 10가지 지정
- 사각형 크기가 25씩 커지게 하기

## 연습문제 2-코드와 결과

```
def square(t, size, color):  
    t.color(color)  
    for i in range(4):  
        t.forward(size)  
        t.right(90)  
  
t.clearscreen()  
t.speed(13)  
t.pensize(5)  
colors = ['red', 'orange', 'yellow',  
          'green', 'blue', 'violet',  
          'darkgreen', 'hotpink', 'grey',  
          'black']  
  
i = 25  
for color in colors:  
    square(t, i, color)  
    i = i + 25
```



## 연습문제 3

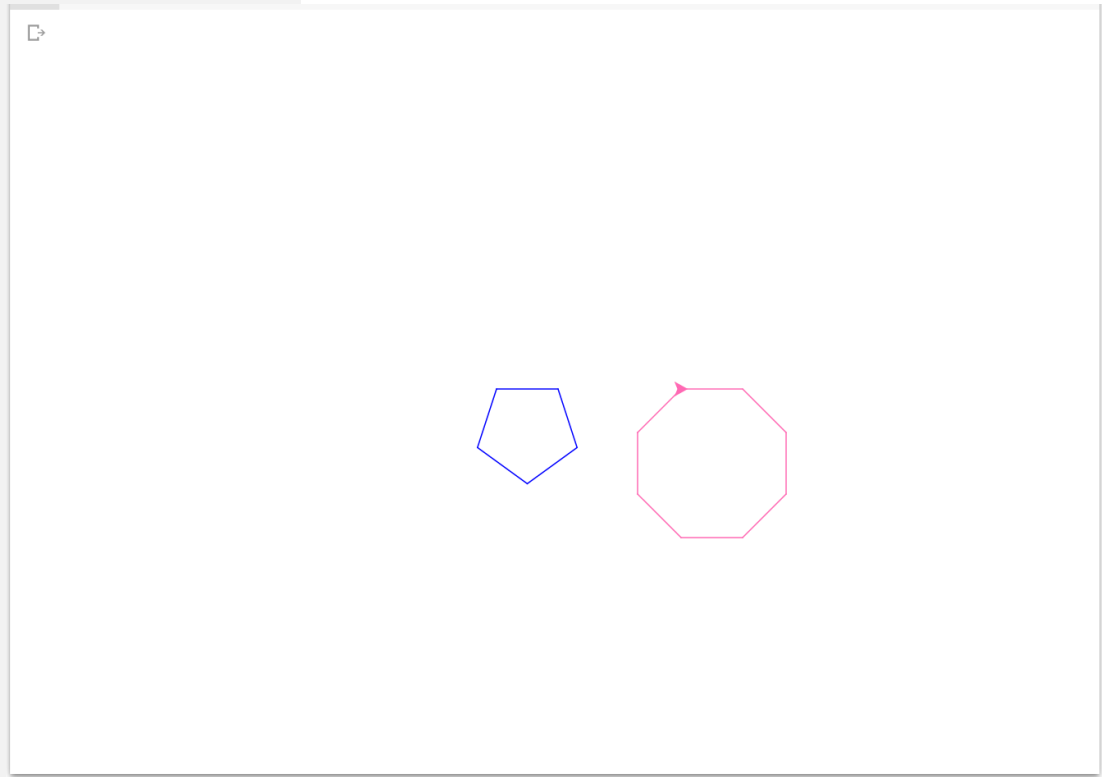
---

다각형을 그리는 다음 함수를 사용하여 5각형과 8각형을 그리시오

```
def drawPolygon(sideLength, numSides, color):  
    t.color(color)  
    turnAngle= 360 / numSides  
    for i in range(numSides):  
        t.pendown()  
        t.forward(sideLength)  
        t.right(turnAngle)
```

## 연습문제 3-코드와 결과

```
def drawPolygon(t, sideLength, numSides, color):  
    t.color(color)  
    turnAngle= 360 / numSides  
    for i in range(numSides):  
        t.pendown()  
        t.forward(sideLength)  
        t.right(turnAngle)  
  
t.clearscreen()  
t.speed(13)  
t.penup()  
t.setposition(-50, 0)  
drawPolygon(t, 50, 5, 'blue')  
  
t.penup()  
t.setposition(100, 0)  
drawPolygon(t, 50, 8, 'hotpink')
```





## 요약

---

- Turtle활용하여 다양한 모양 그려보기
- 반복문 활용하기
- 함수 활용하기

# 07. 예외처리의 이해와 활용

## [2차시]

## 학습목표

---

- 예외처리 개념 이해하기
- 입력하는 값의 범위를 지정하여 처리하기
- 없는 파일을 열었을 때 어떻게 처리하는지 알아보기
- ZeroDivisionerror 를 이해하고 활용해 보기

## 예외 처리

---

- 프로그래머의 의도와 동떨어진 상황이 발생하는 것을 예외(exception)라고 함
- 예상되는 에러를 처리하도록 코딩을 추가하는 과정
- 코드를 실행하다가 문제가 발생할 수 있는 예외적인 상황을 처리하는 코드를 추가
- 예상 가능한 문제를 구체적으로 표시하여 처리

# 사용 예제

---

```
#check File IO

import sys

try:
    inf = open('myfile.txt', 'r')
    s = f.readline()

except IOError as err:
    print("I/O error: {0}".format(err))
```

```
I/O error: [Errno 2] No such file or directory: 'myfile.txt'
```

## 사용 예제

```
# check zero division

import sys

def divide(x, y):
    try:
        result = x / y

    except ZeroDivisionError:
        print("division by zero!")

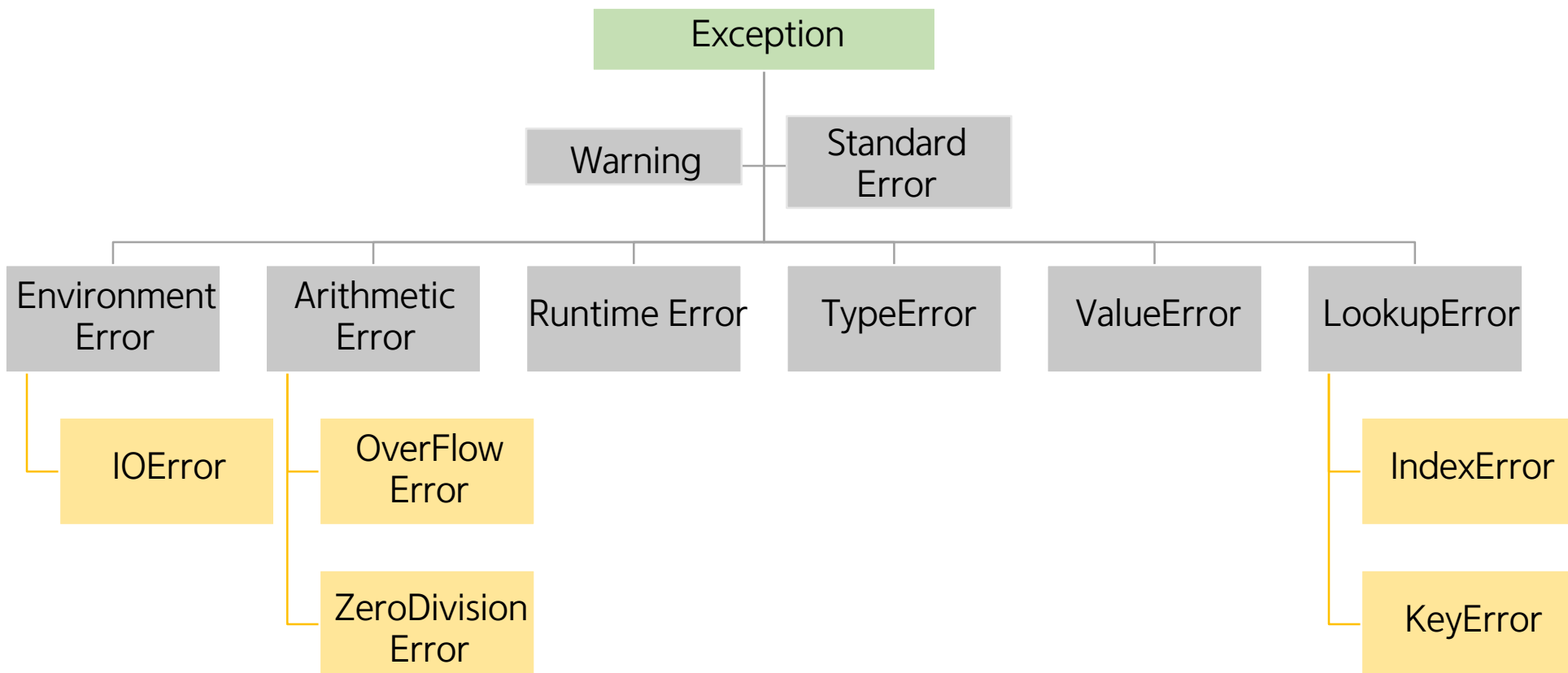
    else:
        print("result is", result)

    finally:
        print("executing finally clause")

divide(10, 0)
```

➡ division by zero!  
executing finally clause

# 예외처리 구성



# Try: Exception 1

```
import sys

while True:
    try:
        x = int(input("Please enter an integer: "))
        break

    except ValueError:
        print("Oops! That was no valid number. Try again...")
```

```
➞ Please enter an integer: as
    Oops! That was no valid number. Try again...
    Please enter an integer: v
    Oops! That was no valid number. Try again...
    Please enter an integer: 5.4
    Oops! That was no valid number. Try again...
    Please enter an integer: 9
```



## Try: Exception 2

---

```
import sys

try:
    x = 1/0

except ZeroDivisionError as err:
    print('Handling run-time error:', err)
```

 Handling run-time error: division by zero

# Except with Multiple Exceptions

```
import sys

try:
    f = open('myfile.txt', 'r')
    s = f.readline()
    i = int(s.strip())
except IOError as err:
    print("I/O error: {0}".format(err))
except ValueError:
    print("Could not convert data to an integer.")
except:
    print("Unexpected error:", sys.exc_info()[0])
```

 I/O error: [Errno 2] No such file or directory: 'myfile.txt'

# Try-Finally 사용

```
try:
    fh = open("t.txt", "w")
    try:
        fh.write("This is my test file for exception handling!!\n")
        fh.write("=" * 35)
    finally:
        print("Going to close the file")
        fh.close()
        fh = open("t.txt", "r")
        print(fh.readlines())
        fh.close()

except IOError:
    print("Error: can't find file or read data")
```



Going to close the file

```
['This is my test file for exception handling!!\n', '=====']
```

## Assertion (1)

---

- 원하는 범위의 값을 입력 받고 싶을 때, `assert()`문장을 넣으면 그 범위의 값이 들어오지 않았을 때 에러 메시지를 띄우고 프로그램이 중단 됨
- 에러 처리를 해서 진행하지 않고, 에러 처리가 힘들거나 중요한 값이라서 중단해야 할 때 사용 함

# Assertion (2)

```
# assert
```

```
def KToFahrenheit(Temperature):  
    assert (Temperature >= 0), print("Colder than absolute zero!")  
    return ((Temperature-273)*1.8)+32
```

```
print(KToFahrenheit(273))  
print(KToFahrenheit(505.78))  
print(KToFahrenheit(-5))
```

```
32.0  
451.00399999999996  
Colder than absolute zero!  
-----  
AssertionError                                Traceback (most recent call last)  
<ipython-input-11-13750d01cc8f> in <module>()  
      7 print(KToFahrenheit(273))  
      8 print(KToFahrenheit(505.78))  
----> 9 print(KToFahrenheit(-5))  
  
<ipython-input-11-13750d01cc8f> in KToFahrenheit(Temperature)  
      2  
      3 def KToFahrenheit(Temperature):  
----> 4     assert (Temperature >= 0), print("Colder than absolute zero!")  
      5     return ((Temperature-273)*1.8)+32  
      6  
  
AssertionError: None
```

SEARCH STACK OVERFLOW

# Assertion (3)

```
# assert
```

```
num = int(input('Enter a positive number: '))  
assert(num > 0), print('Only positive numbers are allowed!')
```

```
print("number is ", num )
```

```
Enter a positive number: 7  
number is 7
```

```
Enter a positive number: -3  
Only positive numbers are allowed!
```

```
-----  
AssertionError                                Traceback (most recent call last)  
<ipython-input-13-da7b6bbbc30f> in <module>()  
      1 # Assertion (3)  
      2 num = int(input('Enter a positive number: '))  
----> 3 assert(num > 0), print('Only positive numbers are allowed!')  
      4  
      5 print("number is ", num )
```

```
AssertionError: None
```

SEARCH STACK OVERFLOW

## 연습문제 1

---

- 나이를 입력 받는다
- 입력 받은 나이가 1보다 작고, 100보다 크면 오류라고 알려준다
- assert문을 사용한다

## 연습문제 1-코드와 결과

```
# assert
```

```
age = int(input('Enter your age: '))  
assert(age > 1 and age <= 100), print("나이는 1세에서 100세까지 만 입력 가능!")  
  
print("age is ", age )
```

```
Enter your age: 5  
age is 5
```

```
Enter your age: 102  
나이는 1세에서 100세까지 만 입력 가능!  
-----  
AssertionError                                Traceback (most recent call last)  
<ipython-input-15-6e302b9a14af> in <module>()  
      2  
      3 age = int(input('Enter your age: '))  
----> 4 assert(age > 1 and age <= 100), print("나이는 1세에서 100세까지 만 입력 가능!")  
      5  
      6 print("age is ", age )
```

AssertionError: None

SEARCH STACK OVERFLOW



## 연습문제 2

---

- 나이를 입력 받는다
- 입력 받은 나이가 1보다 작고, 100보다 크면 다시 입력 받도록 코딩한다
- while문 사용한다

## 연습문제 2-코드와 결과

```
age = int(input('Enter your age: '))

while age < 1 or age > 100 :
    print("처리할 수 없는 숫자가 입력 되었습니다, 다시 입력해 주세요")
    age = int(input('Enter your age: '))

print("입력된 나이는 ", age, "세 입니다")
```

```
Enter your age: 12
입력된 나이는 12 세 입니다
```

```
Enter your age: 102
처리할 수 없는 숫자가 입력 되었습니다, 다시 입력해 주세요
Enter your age: 99
입력된 나이는 99 세 입니다
```

## 연습문제 3

---

- 나이를 입력 받는다
- 숫자가 아니고, 이십오세등의 문자로 입력되면 다시 입력 받도록 코딩한다
- Try문을 사용한다

## 연습문제 3-코드와 결과

```
import sys

while True:
    try:
        age = int(input('Enter your age: '))
        break

    except ValueError:
        print("처리할 수 없는 문자가 입력되었습니다! 정수로 입력하세요")
```

Enter your age: 이십오세  
처리할 수 없는 문자가 입력되었습니다! 정수로 입력하세요  
Enter your age: 25

## 연습문제 4

---

- 더치페이를 계산해주는 프로그램을 작성한다
- 금액과 인원을 입력받아 각자 내야 하는 금액을 출력 한다
- 인원이 0일때, 정수가 아닌 인원이 입력되면 오류라고 알려준다

## 연습문제 4-코드

```
import sys

try:
    money = int(input("금액을 입력하세요: "))
    people = int(input("인원을 입력하세요: "))

    pay = money/people

    print("각자 ", str(pay), " 원 씩 나누어 내세요")

except ZeroDivisionError as err:
    print(err, ': 인원은 1명 이상이어야 합니다 ')

except ValueError as err:
    print(err, ': 정수만 입력 가능합니다 ')
```

## 연습문제 4-결과

☞ 금액을 입력하세요 : 20000  
인원을 입력하세요 : 4  
각자 5000.0 원 씩 나누어 내세요

☞ 금액을 입력하세요 : 50000  
인원을 입력하세요 : 0  
division by zero : 인원은 1명 이상이어야 합니다

☞ 금액을 입력하세요 : 40000  
인원을 입력하세요 : 네명  
invalid literal for int() with base 10: '네명' : 정수만 입력 가능합니다

## 강의 요약

---

- 예외처리 개념 이해하기
  - 의도되지 않은 상황(예외)을 처리하는 코드를 추가하는 것
  - try, except, finally 구문 활용
- 입력하는 값의 범위를 지정하여 처리하기
  - assert(원하는 범위 지정)
- 없는 파일을 열었을 때 어떻게 처리하는지 알아보기
  - IOError
- ZeroDivisionerror 를 이해하고 활용해 보기



## 퀴즈 1

---

예외처리 구문에 속하는 명령어를 모두 표시하시오

- ① try
- ② traceback
- ③ except
- ④ finally
- ⑤ assert
- ⑥ if

## 퀴즈 1-답안

---

예외처리 구문에 속하는 명령어를 모두 표시하시오

- ① try
- ② traceback
- ③ except
- ④ finally
- ⑤ assert
- ⑥ if

## 퀴즈 2

---

다음 assert()문에 대해 설명하시오

```
age = input('Enter your age: ')\nage = int(age)\nassert(age > 1 and age <= 100), print("나이는 1세에서 100세까지!")
```

## 퀴즈 2-답안

### 다음 assert()문에 대해 설명하시오

```
age = input('Enter your age: ')
age = int(age)
assert(age > 1 and age <= 100), print("나이는 1세에서 100세까지!")
```

age > 1 and age <= 100 값인 경우에 허용하고,  
그렇지 않은 값이 입력되면 "나이는 1세에서 100세까지"  
라고 출력한다

```
Enter your age: 105
나이는 1세에서 100세까지!

-----
AssertionError                                Traceback (most recent call last)
<ipython-input-22-fc294e55589f> in <module>()
      2 age = input('Enter your age: ')
      3 age = int(age)
----> 4 assert(age > 1 and age <= 100), print("나이는 1세에서 100세까지!")

AssertionError: None
```

SEARCH STACK OVERFLOW

# 08. 문자열의 이해

## [2차시]

# 학습목표

---

- 문자열 이해하기
- 문자열에서 사용하는 연산자를 이해하기
- 문자열 지원하는 메소드를 알고 활용하기

## 문자열 (string)

---

- 문자열은 글자들(characters)의 나열(sequence)
- 문자열은 구성되는 각 글자를 첨자 표현으로 따로 나눠서 활용 가능

```
>>> name = 'apple'
```

```
>>> print(name[0])
```

```
'a'
```

```
>>> school = 'DB하이텍'
```

```
>>> address = '경기도 부천시 수도로 90'
```

# 문자열의 구성

변수명	저장된 값		
str	"apple"	"lemon"	"DB하이텍"
str[0]	"a"	"l"	"D"
str[1]	"p"	"e"	"B"
str[2]	"p"	"m"	"하"
str[3]	"l"	"o"	"이"
str[4]	"e"	"n"	"텍"



## 문자열에서 사용하는 연산자 (1)

- 산술연산자 : +, \*

```
>>> s1 = 'Cutty'
>>> s2 = 'Cat'
>>> s3 = s1 + s2
>>> s3
'CuttyCat'
>>> s1 * 3
'CuttyCuttyCutty'
>>> '@' * 10
'@@@@@@@@@@@'
```

## 문자열에서 사용하는 연산자 (2)

- 관계연산자 : >, >=, <, <=, ==, !=

```
>>> s1 = 'cat'
>>> s2 = 'Cat'
>>> s1 == s2
False
>>> s1 == 'cat'
True
>>> s1 > 'bird'
True
```

## 문자열 길이

- `len()` 은
  - 내장된 함수로서 문자열을 구성하는 글자수를 반환


```
>>> fruit = 'banana'
>>> len(fruit)
6
>>> length = len(fruit)
>>> last = fruit[length]
IndexError: string index out of range

>>> last = fruit[length-1]
>>> last
a
```

## 반복문으로 문자열 처리

```
fruit="apple"

index = 0
while index < len(fruit) :
    letter = fruit[index]
    print(letter)
    index = index + 1
```



```
fruit="apple"

for ch in fruit :
    print(ch)
```

for문이 더단순함

## 문자열 자르기

- 문자열의 일부분을 조각(slice) 이라고 함
- 조각을 고르는 것은 글자(character)를 고르는 것과 유사

```
>>> s = 'Monty Python'
>>> s[0:5]          # index 0,1,2,3,4
Monty
>>> s[6:12]         # index 6,7,8,9,10,11
Python
>>> s[1:3]          # index 1,2
on
>>> fruit = 'banana'
>>> fruit[:3]        # index 0,1,2
'ban'
>>> fruit[3:]        # index from 3 to last
'ana'
```

## 연습문제 1

---

- 문자열을 입력 받는다
  - 만약 입력된 값이 'apple'이라고 하면..
- 반복문과 함수 len()사용한다
  - 입력한 문자열을 다음과 같이 출력되도록 코딩한다
    - 'a'
    - 'ap'
    - 'app'
    - 'appl'
    - 'apple'

## 연습문제 1-코드와 결과

```
input_str = input("문자열 입력 : ")
i = len(input_str)
index = 0

while index < i :
    print("s[0:", index+1, "]= ", input_str[0 : index+1])
    index = index + 1
```

```
☞ 문자열 입력 : strawberry
s[0: 1 ]= s
s[0: 2 ]= st
s[0: 3 ]= str
s[0: 4 ]= stra
s[0: 5 ]= straw
s[0: 6 ]= strawb
s[0: 7 ]= strawbe
s[0: 8 ]= strawber
s[0: 9 ]= strawberry
s[0: 10 ]= strawberry
```

## 문자열 대문자 변환

- 메소드(method)는 인수들을 받고 값을 반환
  - 이 점에서 함수(function)와 유사
  - 사용하는 문법(syntax)은 다름
  - 변수명을 쓰고 메소드를 사용
- 메소드 **.upper()** 를 사용
  - 모든 문자를 대문자로 변환

```
>>> word = 'banana'
>>> new_word = word.upper()
>>> new_word
BANANA
```



## 문자열 검색

- 메소드 `.find()`는 문자열에서 특정문자를 찾음
- 찾는 문자가 존재하는 위치(index)를 알려줌

```
>>> word = 'banana'
>>> index = word.find('a')
>>> index
1
>>> word.find('na')
2
>>> word.find('na', 3)    #두번째 숫자는 찾기 시작하는 index number
4
>>> name = 'bob'
>>> name.find('b', 1, 2) # 세번째 숫자는 어디까지 찾을지 지정하는 index number
-1
```

# 문자열 매서드

<code>.capitalize()</code>	첫 글자만 대문자로 바꾼다	<code>.isupper()</code>	모든 요소가 대문자이면 참, 아니면 거짓
<code>.count()</code>	지정 구간에서 글자수를 센다	<code>.join()</code>	문자열을 합쳐준다
<code>.find()</code>	찾는 문자 또는 문자열이 시작하는 첨자를 찾아준다	<code>.lower()</code>	소문자로 바꾼다
<code>.isalpha()</code>	모든 요소가 알파벳이면 참, 아니면 거짓	<code>.replace()</code>	<code>.replace(old, new [,count])</code> 다른 문자열로 바꿔준다
<code>.isdigit()</code>	모든 요소가 숫자이면 참, 아니면 거짓	<code>.split()</code>	<code>.split(seq=None, maxsplit=-1)</code> seq(구분자)를 기준으로 maxsplit(기본값 -1)만큼 문자를 나눈다.
<code>.islower()</code>	모든 요소가 소문자이면 참, 아니면 거짓	<code>.swapcase()</code>	대문자는 소문자로, 소문자는 대문자로 바꾼다
<code>.isspace()</code>	모든 요소가 공백이면 참, 아니면 거짓	<code>.upper()</code>	대문자로 바꾼다

# in 연산자

```
>>> 'a' in 'banana'
True
>>> 'seed' in 'banana'
False
>>> 'ana' in 'banana'
True
```

#함수 사용

```
def in_both(word1, word2):
    for letter in word1:
        if letter in word2:
            print(letter)

in_both('apples', 'oranges')
```

A small, light gray rectangular window with a thin border. Inside the window, there is a small icon of a square with an arrow pointing right, followed by the letters 'a', 'e', and 's' stacked vertically.

## 연습문제 2

---

- 문자열을 입력 받는다
- 대문자로 바꿀 문자 또는 문자열을 입력 받는다
- 입력 받은 문자를 대문자로 바꾼 후 전체 문자열을 출력한다
- 예시
  - 입력 문자열 "apple"
  - 바꿀 문자열 "le"
  - 출력결과 "appLE"

## 연습문제 2-코드와 결과

```
Input_Str = input("문자열 입력 : ")
Input_find = input("찾아 바꿀 문자 : ")
find_Length = len(Input_find)

Index = Input_Str.find(Input_find)

if Index == -1 :
    print(" 바꿀 문자가 없네요!")
else:
    beforeStr = Input_Str[0 : Index]
    changeStr = Input_Str[Index : Index+find_Length].upper()
    afterStr = Input_Str[Index+find_Length : ]
    result = beforeStr + changeStr + afterStr
    print(result)
```

☞ 문자열 입력 : Python  
찾아 바꿀 문자 : on  
PythON

☞ 문자열 입력 : Python  
찾아 바꿀 문자 : PY  
바꿀 문자가 없네요!

## 연습문제 3

---

- 문자열을 입력 받는다
- 대문자는 소문자로, 소문자는 대문자로 바꾸어서 저장하고 출력한다
- 바꾼 후에 대문자 개수, 소문자 개수, 대소문자가 아닌 문자 개수를 출력한다

## 연습문제 3-코드와 결과

```
str = input("문자열 입력: ")
uppercount = 0
lowercount = 0
etccount = 0

swapstr = str.swapcase()
print("대소문자를 바꾼 결과 = ", swapstr)

for ch in swapstr :
    if ch.isupper() :
        uppercount +=1
    elif ch.islower() :
        lowercount +=1
    else :
        etccount +=1

print("대문자 갯수는 ", uppercount)
print("소문자 갯수는 ", lowercount)
print("대소문자 아닌 문자 갯수는 ", etccount)
```

☞ 문자열 입력 : I love PYTHON!!  
대소문자를 바꾼 결과 = i LOVE python!!  
대문자 갯수는 4  
소문자 갯수는 7  
대소문자 아닌 문자 갯수는 4

## 연습문제 4

---

- 문자열을 입력 받는다
- 문자열의 첫 글자는 대문자, 나머지는 소문자로 바꿔서 저장하고 출력한다
- 문자열의 첫 글자와 나머지 글자를 나누어 출력한다



## 연습문제 4-코드와 결과

```
str = input("문자열 입력: ")  
  
strChange = str.lower()  
strChange = str.capitalize()  
  
length = len(strChange)  
  
print("변경된 문자열: ", strChange)  
print("첫글자: ", strChange[0])  
print("나머지 글자: ", strChange[1: length])
```

☞ 문자열 입력 : cheeseCake  
변경된 문자열 : Cheesecake  
첫글자 : C  
나머지 글자 : heesecake

## 강의 요약

---

- 문자열 이해하기
  - 글자들의 나열
  - 구성되는 각 글자를 첨자 표현으로 나누어 활용 가능
- 문자열에서 사용하는 연산자를 이해하기
  - 산술연산자(+, \*), 관계연산자(>, >=, <, <=, ==, !=)
- 문자열 지원하는 메소드를 알고, 활용하기
  - .upper(), .find()

## 퀴즈 1

---

다음 코드 실행 시 결과는?

```
name = 'Blueberry'  
print(name[0:1])
```

- ① B
- ② Bl
- ③ y
- ④ ry

## 퀴즈 1-답안

다음 코드 실행 시 결과는?

```
name = 'Blueberry'  
print(name[0:1])
```

- ① B
- ② Bl
- ③ y
- ④ ry

A small terminal window with a light gray border and a white background. It contains a small icon of a terminal window on the left and the text 'B' on the right.

## 퀴즈 2

---

다음 코드 실행 시 결과는?

```
w = 'Blueberry'  
print(w.find('e', 3))
```

- ① 3
- ② 4
- ③ 5
- ④ 6

## 퀴즈 2-답안

다음 코드 실행 시 결과는?

```
w = 'Blueberry'  
print(w.find('e', 3))
```

- ① 3
- ② 4
- ③ 5
- ④ 6

3

# 09. 리스트의 이해

## [2차시]

# 학습목표

---

- 리스트를 이해하기
- 리스트에서 연산자 활용하기
- 리스트에서 메소드 활용하기
- 2차원 리스트 이해하기



# List (목록 또는 배열)

- 리스트는 값들의 나열(sequence)이다
  - 리스트 안 구성요소를
    - 원소(elements) 혹은 항목(items)이라고 부른다
    - 다양한 종류의 데이터타입으로 구성 가능하다
  - 리스트의 내용은 변경 가능하다

```
>>> cheeses = ['Cheddar', 'Edam', 'Gouda']
>>> numbers = [17, 123]
>>> print(numbers)
[17, 123]
>>> cheeses[0]
Cheddar
>>> numbers
[17, 123]
```

## 리스트에서 in 연산자

```
>>> cheeses = ['Cheddar', 'Edam', 'Gouda']
>>> 'Edam' in cheeses
True
>>> 'Brie' in cheeses
False
>>> cheeses
['Cheddar', 'Edam', 'Gouda']
>>> for food in cheeses :
    print(food)
Cheddar
Edam
Gouda
```

## 리스트 다루기

```
cheeses = ['Cheddar', 'Edam', 'Gouda']  
numbers = [1, 3, 5, 7, 9, 11]
```

```
for cheese in cheeses :  
    print(cheese)
```

```
for i in range(len(numbers)) :  
    numbers[i] = numbers[i] * 2  
    print(numbers[i])
```

```
print(numbers)
```

```
➞ Cheddar  
   Edam  
   Gouda  
   2  
   6  
   10  
   14  
   18  
   22  
   [2, 6, 10, 14, 18, 22]
```

# 리스트, 연산자

## ■ Operator

- + : 두개의 리스트를  
합하여 새로운 리스트를  
생성
- \* : (리스트)\*(정수)형태로  
정수만큼 리스트의 내용  
이 배가 됨

```
# operator +
>>> a = [1, 2, 3]
>>> b = [4, 5, 6]
>>> c = a + b
>>> c
[1, 2, 3, 4, 5, 6]
```

```
# operator *
>>> ['a'] * 4
['a', 'a', 'a', 'a']
>>> a = [1, 2, 3]
>>> a * 3
[1, 2, 3, 1, 2, 3, 1, 2, 3]
```

# 리스트 분할

- Slice
  - 문자열 사용과 동일
  - index를 사용하여, 리스트 내의 아이템들을 일부 사용한다
  - list\_fruit[:3]
  - 문자열에서 사용하는 것과 동일

```
# list slice
>>> t = ['a', 'b', 'c', 'd', 'e', 'f']
>>> t[1:3]
['b', 'c']

>>> t[:4]
['a', 'b', 'c', 'd']

>>> t[3:]
['d', 'e', 'f']
```

## 연습문제 1

---

- 새로 생성하는 List의 item 수를 입력 받는다
- list를 생성한다
- item의 수만큼 반복해서 값을 입력받는다
- '+' 연산자를 사용하여 list에 값을 추가한다
- 전체 리스트를 출력한다

## 연습문제 1-코드와 결과

```
num = int(input("List element 개수 입력: "))
NewList = []
tempList = [0]

for i in range(num):
    print(i, "번째")
    t = input("추가할 element 입력: ")
    tempList = [t]
    NewList = NewList + tempList

print(NewList)
```

```
➡ List element 개수 입력: 4
0 번째
추가할 element 입력: 12
1 번째
추가할 element 입력: 34
2 번째
추가할 element 입력: 2
3 번째
추가할 element 입력: 9
['12', '34', '2', '9']
```

# 리스트 메서드

<code>.append()</code>	리스트 내에 새로운 아이템 한 개를 추가하여, 마지막에 위치한다
<code>.insert()</code>	리스트 내에 index 번호와 아이템 내용을 추가한다
<code>.extend()</code>	다른 이름의 리스트, 아이템 모두를 추가한다
<code>.sort()</code>	리스트의 아이템들을 순서대로 정렬, 순서는 ascii code 순이다
<code>.pop()</code>	리스트 내에 존재하는 아이템의 index 번호를 입력 받아 삭제한다
<code>.remove()</code>	리스트 내에 존재하는 아이템을 삭제, 동일한 아이템 있으면 첫번째 아이템을 삭제



# .append()

```
# method append  
  
>>> t1 = ['x', 'y', 'z']  
>>> t1.append('a')  
>>> t1  
['x', 'y', 'z', 'a']  
>>> t1.append('e')  
>>> t1  
['x', 'y', 'z', 'a', 'e']
```

# .insert()

```
# method insert
```

```
>>> t1 = ['x', 'y', 'z']
```

```
>>> t1 ['x', 'y', 'z']
```

```
>>> t1.insert(1, 'a')
```

```
>>> t1
```

```
['x', 'a', 'y', 'z']
```

```
>>> t1.insert(1, 'e')
```

```
>>> t1
```

```
['x', 'e', 'a', 'y', 'z']
```

# .extend()

```
# method extend
```

```
>>> t1 = ['x', 'y', 'z']
```

```
>>> t2 = ['d', 'e']
```

```
>>> t1.extend(t2)
```

```
>>> t1
```

```
['x', 'y', 'z', 'd', 'e']
```

```
>>> t2.extend(t1)
```

```
>>> t2
```

```
['d', 'e', 'x', 'y', 'z', 'd', 'e']
```

# .sort()

---

```
#method sort
```

```
>>> t = ['d', 'c', 'e', 'b', 'a']
```

```
>>> t.sort()
```

```
>>> t
```

```
['a', 'b', 'c', 'd', 'e']
```

## .pop()

---

```
#method pop(index)

>>> t = ['a', 'b', 'c', 'd', 'e']
>>> x = t.pop(0) + t.pop(1)
>>> t
['b', 'd', 'e']
>>> t.append('a')
>>> t
['c', 'd', 'e', 'a']
```

## .remove()

---

```
# method remove(value)
```

```
>>> t = ['a', 'b', 'c']
```

```
>>> t.remove('b')
```

```
>>> t
```

```
['a', 'c']
```

## 메서드 활용하기 1

```
f1=['apple', 'blueberry', 'melon', 'tomato']  
f2=['strawberry', 'lemon', 'banana']  
f3=f1+f2  
print('f1+f2= ', f3)  
  
f3.append('blackberry')  
f3.sort()  
print("after sorting = ", f3)
```

```
➡ f1+f2= ['apple', 'blueberry', 'melon', 'tomato', 'strawberry', 'lemon', 'banana']  
   after sorting = ['apple', 'banana', 'blackberry', 'blueberry', 'lemon', 'melon', 'strawberry', 'tomato']
```

## 메서드 활용하기 2

```
f1=['apple', 'blueberry', 'melon', 'tomato']  
f2=['strawberry', 'lemon', 'banana']  
f3=f1+f2  
print(f3)
```

```
# remove element with its first char 'b'  
index=len(f3)  
i=0
```

```
↳ ['apple', 'blueberry ', 'melon', 'tomato', 'strawberry', 'lemon', 'banana']  
remove all 'b' elements = ['apple', 'melon', 'tomato', 'strawberry', 'lemon']
```

```
while i < index:  
    if f3[i][0] == "b" :  
        f3.remove(f3[i])  
        i=i-1  
        index=index-1  
    i=i+1  
  
print("remove all 'b' elements = ", f3)
```



## 메서드 활용하기 2-설명

#이전 예제 중 일부

```
while i < index:
    if f3[i][0] == "b" :
        f3.remove(f3[i])
        i=i-1
        index=index-1
    i=i+1
```

- 'b'로 시작하는 문자열을 찾아서 삭제하고 나면
  - 리스트 f3에 있는 전체 아이템 개수는 하나 줄어든다 (**index=index-1**)
  - 그 다음 문자열을 확인하기 위해서는 **i=i-1** 이 필요하다
  - 조건에 맞아서 삭제된 아이템이 가지는 첨자를 그 뒤에 있는 아이템이 가지게 된다

## 연습문제 1

---

- 자신의 이름을 입력 받아서 리스트로 만들어서 출력 한다
- 이름은 문자 중 삭제할 문자를 입력 받아 그 문자를 생성된 리스트에서 삭제한다
- 삭제 후 리스트의 내용을 출력한다

## 연습문제 1-코드

```
name = input("이름을 입력하세요: ")
nameList = []
```

```
for ch in name :
    nameList.append(ch)
print(nameList)
```

```
chDel = input("제거할 문자 입력: ")
count = 0
```

```
for ch in nameList :
    if ch == chDel :
        nameList.remove(chDel)
```

```
print(nameList)
```



이름을 입력하세요: hong gil dong

['h', 'o', 'n', 'g', ' ', 'g', 'i', 'l', ' ', 'd', 'o', 'n', 'g']

제거할 문자 입력: n

['h', 'o', 'g', ' ', 'g', 'i', 'l', ' ', 'd', 'o', 'g']

## 연습문제 2

---

- 원하는 단어를 입력받는다
- 입력받은 단어를 구성하는 알파벳을 순서대로 정렬한다
- 모든 알파벳은 소문자로 바꾸어 정렬한다
- 공백이 있는 경우 삭제한다

## 연습문제 2-코드와 결과

```
word = input("단어를 입력하세요: ")
wordList = []
```

```
word = word.lower()
for ch in word:
    wordList.append(ch)
print(wordList)
```

```
wordList.sort()
```

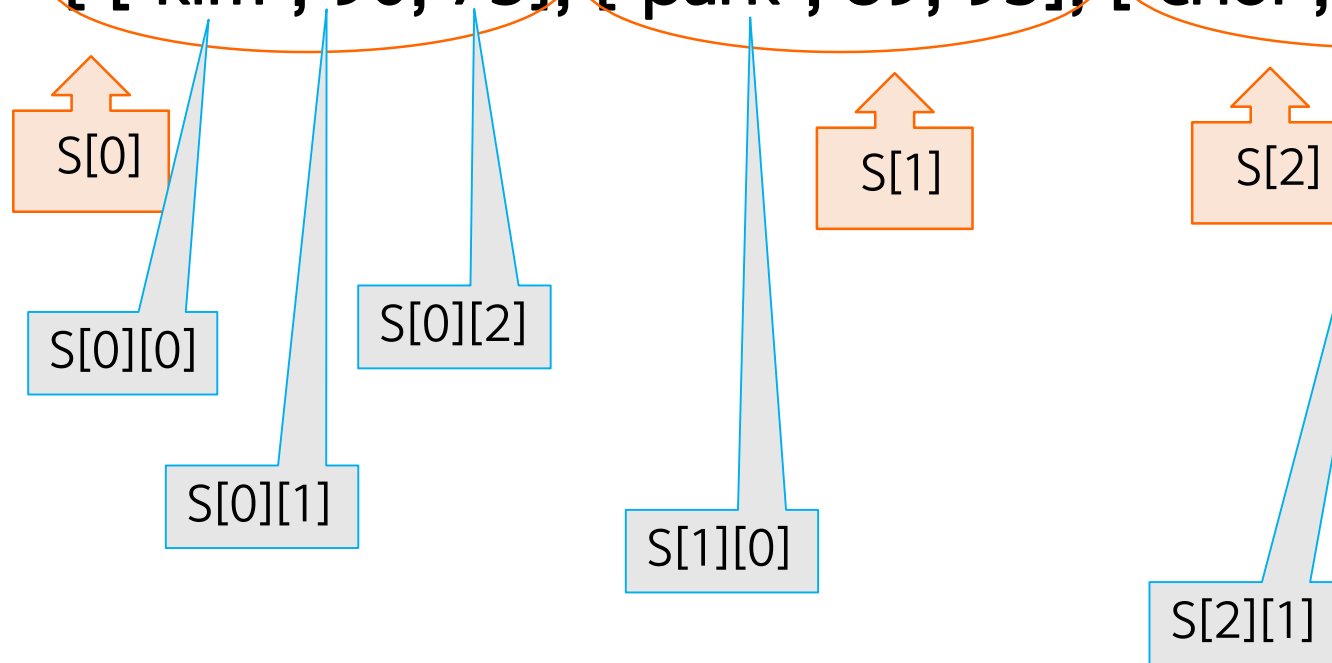
```
while wordList[0] == '':
    wordList.remove('')
```

```
print(wordList)
```

```
☞ 단어를 입력하세요: Life is short. You need python.
['l', 'i', 'f', 'e', ' ', 'i', 's', ' ', 's', 'h', 'o', 'r', 't', ' ', ' ', 'y', 'o', 'u', ' ', ' ', 'n', 'e', 'e', 'd', ' ', ' ', 'p', 'y', 't', 'h', 'o', 'n', ' ', '.']
['.', ' ', ' ', 'd', 'e', 'e', 'e', 'f', 'h', 'h', 'i', 'i', 'l', 'n', 'n', 'o', 'o', 'o', 'p', 'r', 's', 's', 't', 't', 'u', 'y', 'y']
```

## 2차원 리스트

- 2차원 목록(list) 생성
- 리스트 안에 리스트가 또 만들어진다
- `s = [ ["kim", 90, 75], ["park", 89, 95], ["choi", 76, 85] ]`



## 2차원 리스트 활용

```
fq= [[0,0,0,0,0,0],[0,0,0,0,0,0],[0,0,0,0,0,0],[0,0,0,0,0,0],[0,0,0,0,0,0]]

for i in range(5):
    for j in range(6):
        fq[i][j] = i+j
    print(i,"th row : ", fq[i])

print("="*50)
print("all : ", fq)
print("="*50)
```

```
➞ 0 th row :  [0, 1, 2, 3, 4, 5]
   1 th row :  [1, 2, 3, 4, 5, 6]
   2 th row :  [2, 3, 4, 5, 6, 7]
   3 th row :  [3, 4, 5, 6, 7, 8]
   4 th row :  [4, 5, 6, 7, 8, 9]

=====
all :  [[0, 1, 2, 3, 4, 5], [1, 2, 3, 4, 5, 6], [2, 3, 4, 5, 6, 7], [3, 4, 5, 6, 7, 8], [4, 5, 6, 7, 8, 9]]
=====
```

## 2차원 리스트-성적처리

```
s = [ ["kim", 90, 75], ["park", 89, 95], ["choi", 76, 85] ]  
print( s )
```

```
for i in range( len(s) ) :  
    print( s[i][0] )  
    sum=0  
    for j in range( 1, len(s[i]) ) :  
        sum = sum + s[i][j]  
    print("sum =", sum, "average =", sum/j, "\n")
```

```
[['kim', 90, 75], ['park', 89, 95], ['choi', 76, 85]]
```

```
kim
```

```
sum = 165 average = 82.5
```

```
park
```

```
sum = 184 average = 92.0
```

```
choi
```

```
sum = 161 average = 80.5
```



## 연습문제 3

---

- 2단에서 16단까지 구구단을 계산하여,
- 생성한 2차원 리스트에 저장한다
- 이 때, 각 단을 한 개의 row에 저장한다
- 저장한 결과를 출력한다

## 연습문제 3-코드와 결과

```
mul = [ 10 * [0] for i in range(15) ]
print(mul)
```

```
for i in range(15) :
    for j in range(10) :
        mul[i][j] = (i+2) * (j+1)
    print(mul[i])    #print by row
```

```
print("Done!!")
```

```
➡ [[0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
    [2, 4, 6, 8, 10, 12, 14, 16, 18, 20],
    [3, 6, 9, 12, 15, 18, 21, 24, 27, 30],
    [4, 8, 12, 16, 20, 24, 28, 32, 36, 40],
    [5, 10, 15, 20, 25, 30, 35, 40, 45, 50],
    [6, 12, 18, 24, 30, 36, 42, 48, 54, 60],
    [7, 14, 21, 28, 35, 42, 49, 56, 63, 70],
    [8, 16, 24, 32, 40, 48, 56, 64, 72, 80],
    [9, 18, 27, 36, 45, 54, 63, 72, 81, 90],
    [10, 20, 30, 40, 50, 60, 70, 80, 90, 100],
    [11, 22, 33, 44, 55, 66, 77, 88, 99, 110],
    [12, 24, 36, 48, 60, 72, 84, 96, 108, 120],
    [13, 26, 39, 52, 65, 78, 91, 104, 117, 130],
    [14, 28, 42, 56, 70, 84, 98, 112, 126, 140],
    [15, 30, 45, 60, 75, 90, 105, 120, 135, 150],
    [16, 32, 48, 64, 80, 96, 112, 128, 144, 160]]
Done!!
```

## 연습문제 4

---

- 단위행렬을 만든다
- 먼저 단위행렬의 크기를 입력 받는다
- 2D list로 단위행렬을 만들고 출력한다

## 연습문제 4-코드

```
Size = int(input("행렬의 크기: "))
temp_row = []
Unit_Matrix = []
for a in range(Size):
    for b in range(Size):
        temp_row.append(0)

    Unit_Matrix.append(temp_row)
    temp_row = []

for i in range(Size):
    for j in range(Size):
        if i == j:
            Unit_Matrix[i][j] = 1
        else:
            Unit_Matrix[i][j] = 0

print(Unit_Matrix)
```

```
➡ 행렬의 크기 : 7
[[1, 0, 0, 0, 0, 0, 0],
 [0, 1, 0, 0, 0, 0, 0],
 [0, 0, 1, 0, 0, 0, 0],
 [0, 0, 0, 1, 0, 0, 0],
 [0, 0, 0, 0, 1, 0, 0],
 [0, 0, 0, 0, 0, 1, 0],
 [0, 0, 0, 0, 0, 0, 1]]
```

## 강의 요약

---

- **리스트 이해하기**
  - 값들의 나열(sequence)
  - 다양한 종류의 데이터타입으로 구성 가능
  - 리스트 내용 변경 가능
- **리스트에서 연산자 활용하기**
  - + : 두개의 리스트를 합함
  - \* : (리스트) \* (정수) 형태, 정수만큼 배로 증가
- **리스트에서 메소드 활용하기**
  - .append, .insert, .extend, .sort, .pop, .remove ...
- **2차원 리스트를 이해하기**
  - 리스트 안에 리스트를 만들어 활용

## 퀴즈 1

---

다음 코드 실행시 결과는?

```
n = [1,3,5]  
print(n * 2)
```

- ① [2, 6, 10]
- ② [2, 6, 10, 2, 6, 10]
- ③ [1, 3, 5]
- ④ [1, 3, 5, 1, 3, 5]

## 퀴즈 1-답안

다음 코드 실행시 결과는?

```
n = [1,3,5]  
print(n * 2)
```

```
[1, 3, 5, 1, 3, 5]
```

- ① [2, 6, 10]
- ② [2, 6, 10, 2, 6, 10]
- ③ [1, 3, 5]
- ④ [1, 3, 5, 1, 3, 5]

## 퀴즈 2

다음 코드 실행시 결과는?

```
t1=['a', 'b', 'c']  
t2=['A', 'B']  
t1.insert(1, 'x')  
t1.extend(t2)  
print(t1)
```

- ① ['a', 'b', 'c', 'x', 'A', 'B']
- ② ['x', 'a', 'b', 'c', 'A', 'B']
- ③ ['a', 'x', 'b', 'c', 'A', 'B']
- ④ ['A', 'B', 'a', 'x', 'b', 'c']



## 퀴즈 2-답안

다음 코드 실행시 결과는?

```
t1=['a', 'b', 'c']  
t2=['A', 'B']  
t1.insert(1, 'x')  
t1.extend(t2)  
print(t1)
```

```
➞ ['a', 'x', 'b', 'c', 'A', 'B']
```

- ① ['a', 'b', 'c', 'x', 'A', 'B']
- ② ['x', 'a', 'b', 'c', 'A', 'B']
- ③ ['a', 'x', 'b', 'c', 'A', 'B']
- ④ ['A', 'B', 'a', 'x', 'b', 'c']

# 10. 문자열과 리스트의 활용

## [2차시]

## 학습목표

---

- 문자열과 리스트 관련 메소드 이해하기
- 문자열로 구성된 리스트 활용하기
- Shallow copy와 Deep copy의 차이 이해하기

# 문자열과 리스트

---

- Method를 사용하여 문자열과 리스트를 쉽게 활용 가능하다
  - .split()
    - 문자열을 워드 단위로 잘라서, 리스트로 생성한다
    - 리스트의 아이템은 문자열의 각각의 워드로 구성된다
  - .join()
    - 리스트를 문자열로 생성해 준다
    - .split 과 반대 기능

## 문자열과 리스트 list()

```
# 문자열을 한 글자씩 나누어서, 리스트로 만드는 함수 list
>>> s = 'apple'
>>> t = list(s)
>>> t
['a', 'p', 'p', 'l', 'e']
```

## 문자열과 리스트 .split()

```
# 문자열을 단어 단위로 리스트를 만드는, .split
>>> s = 'every morning you greet to me'
>>> t = s.split()
>>> t
['every', 'morning', 'you', 'greet', 'to', 'me']

>>> s = 'apple-tree-beans'
>>> delimiter = '-'
>>> s.split(delimiter)
['apple', 'tree', 'beans']
```

## 문자열과 리스트 .join()

```
# 리스트를 문자열로 만드는, .join
```

```
>>> t = ['every', 'morning', 'you', 'greet', 'to', 'me']
```

```
>>> delimiter = ' '
```

```
>>> t.join(delimiter)
```

```
'every morning you greet to me'
```

```
>>> delimiter = '+'
```

```
>>> t.join(delimiter)
```

```
'every+morning+you+greet+to+me'
```

## 연습문제 1

---

- 문자열을 입력 받는다
- 문자열을 분리하여 리스트에 저장한다
- 그 중 제거할 단어를 입력 받아 제거한다
- 추가할 단어를 입력 받아 추가한다
- 리스트를 다시 문자열로 바꾼 뒤 출력한다



## 연습문제 1-코드와 결과

```
msg = input("좋아하는 영어문장을 입력하세요 : ")
msgList = msg.split()
print("List: ", msgList)

Remove_Word = input("제거할 단어를 입력하세요 : ")
msgList.remove(Remove_Word)
print("제거 후 List: ", msgList)

Add_Word = input("추가할 단어를 입력하세요 : ")
msgList.append(Add_Word)
print("추가 후 List: ", msgList)

delimiter = " "
Str = delimiter.join(msgList)

print("join 후 문자열: ", Str)
```

```
➤ 좋아하는 영어문장을 입력하세요 : Life is short
List: ['Life', 'is', 'short']
제거할 단어를 입력하세요 : short
제거 후 List: ['Life', 'is']
추가할 단어를 입력하세요 : long
추가 후 List: ['Life', 'is', 'long']
join 후 문자열: Life is long
```

# Shallow copy and Deep Copy

## ■ Shallow copy

```
>>> c1=[2,4,6,8,10]
```

```
>>> c2=c1
```

- 연산자 '='로만 복사하는 경우
- 동일한 **메모리를 공유**한다

## ■ Deep copy

```
>>> c2=deepcopy(c1)
```

- 서로 다른 메모리에 내용이 복사된다
- 같은 내용이 복사되면서, **새로운 메모리 할당** 받음

## Shallow copy 예제

```
>>> c1=["red", "green"]
>>> c2=c1
>>> c1
['red', 'green']
>>> c2
['red', 'green']
>>> print(id(c1), id(c2))
51580400 51580400
```

- 함수 **id**(변수명)
  - 해당하는 변수의 메모리 주소를 알려준다

## Shallow copy 예제

```
>>> c1=["red", "green"]
>>> c2=["blue", "orange"]
>>> c1
['red', 'green']
>>> c2
['blue', 'orange']
>>> print(id(c1), id(c2))
51580400 52929960
```

## Deep copy 예제

```
# deepcopy

>>> from copy import *

>>> c1=["red","green"]
>>> c2=deepcopy(c1)
>>> c1
['red', 'green']
>>> c2
['red', 'green']
>>> print(id(c1), id(c2))
54278544 54285336
```

- 2개의 리스트가 별도로 관리되어야 하는 경우에 사용한다

## 연습문제 2

---

- 유저의 키가 'zzz'가 나올 때까지 유저로부터 좋아하는 과일을 입력 받는다
- 과일 이름을 목록(list)으로 만든다
- 과일의 이름 목록과 개수를 출력한다

## 연습문제 2-코드와 결과

```
fruit=[]
print("you want to stop, input 'zzz' !!")

while True:
    f = input("write your favorite fruit : ")
    if f == 'zzz':
        break
    else :
        fruit.append(f)
print("list f :",fruit, "count of f: ", len(fruit))
```

```
➡ you want to stop, input 'zzz' !!
write your favorite fruit : apple
write your favorite fruit : lemon
write your favorite fruit : lime
write your favorite fruit : banana
write your favorite fruit : tomato
write your favorite fruit : melon
write your favorite fruit : zzz
list f : ['apple', 'lemon', 'lime', 'banana', 'tomato', 'melon'] count of f: 6
```

## 연습문제 3

---

- 두 개의 List를 생성한다
- 중복되는 아이템을 제거한다
- 두 개의 List를 합쳐 출력한다



## 연습문제 3-코드와 결과

```
List01 = ['apple', 'banana', 'quiz', 'hi', 'bye']  
List01_modified = List01  
List02 = ['Korea', 'hi', 'LOL', 'Python', 'apple']
```

```
for i in range(len(List01)):  
    if List01[i] in List02:  
        List02.remove(List01[i])
```

```
List03 = List01 + List02  
print("List01: ", List01)  
print("List02: ", List02)  
print("합친 후: ", List03)
```

```
List03.sort()  
print("정렬 후: ", List03)
```

```
☞ List01 : ['apple', 'banana', 'quiz', 'hi', 'bye']  
   List02 : ['Korea', 'LOL', 'Python']  
   합친 후 : ['apple', 'banana', 'quiz', 'hi', 'bye', 'Korea', 'LOL', 'Python']  
   정렬 후 : ['Korea', 'LOL', 'Python', 'apple', 'banana', 'bye', 'hi', 'quiz']
```

## 연습문제 4

---

- deepcopy 연습하기
- 사용하려면, `from copy import *` 써야 한다
  - numList 에 정수 5개를 저장한다
  - numshallow에 numList 복사한다
  - numdeep에 numList를 deepcopy한다
  - numshallow, numdeep 에 각각 한 개의 아이템을 추가한다
  - numList, numshallow, numdeep를 출력하여 비교한다

## 연습문제 4-코드와 결과

```
from copy import *
```

```
numList = [1,3,5,7,9]
```

```
numshallow = numList
```

```
numdeep = deepcopy(numList)
```

```
print("numList = ", numList)
```

```
print("numshallow = ", numshallow)
```

```
print("numdeep = ", numdeep)
```

```
numshallow.append(99)
```

```
numdeep.append(111)
```

```
print("after appending", "="*20)
```

```
print("numList = ", numList)
```

```
print("numshallow = ", numshallow)
```

```
print("numdeep = ", numdeep)
```

```
➞ numList = [1, 3, 5, 7, 9]
   numshallow = [1, 3, 5, 7, 9]
   numdeep = [1, 3, 5, 7, 9]
   after appending =====
   numList = [1, 3, 5, 7, 9, 99]
   numshallow = [1, 3, 5, 7, 9, 99]
   numdeep = [1, 3, 5, 7, 9, 111]
```

## 연습문제 5

---

- 문자열을 입력받는다
- 입력받은 문자열의 모든 글자 사이에 공백을 추가하여 출력한다
  - 예: 'apple' => 'a p p l e'

## 연습문제 5-코드와 결과

```
s=input("문자열을 입력하세요: ")  
  
t=list(s)  
  
delimiter=" "  
str = delimiter.join(t)  
print("바뀐 문자열: "+ str)
```

```
☞ 문자열을 입력하세요: hello world  
바뀐 문자열: h e l l o   w o r l d
```

## 강의 요약

---

- 문자열과 리스트 관련 메소드 이해하기
  - .split() 문자열을 잘라서 리스트로 생성
  - .join() 리스트를 문자열로 생성
- 문자열로 구성된 리스트 활용하기
- Shallow copy와 deepcopy의 차이 이해하기
  - shallow copy: 동일한 메모리를 공유
  - deepcopy: 새로운 메모리 할당

## 퀴즈 1

다음 코드의 실행 결과는?

```
>>> s = 'every morning you greet to me'  
>>> t = s.split()  
>>> t
```

- ① 'every morning you greet to me'
- ② ['every morning you greet to me']
- ③ ['every' 'morning' 'you' 'greet' 'to' 'me']
- ④ ['every', 'morning', 'you', 'greet', 'to', 'me']

## 퀴즈 1-답안

다음 코드의 실행 결과는?

```
>>> s = 'every morning you greet to me'  
>>> t = s.split()  
>>> t
```

- ① 'every morning you greet to me'
- ② ['every morning you greet to me']
- ③ ['every' 'morning' 'you' 'greet' 'to' 'me']
- ④ ['every', 'morning', 'you', 'greet', 'to', 'me']

```
➞ ['every', 'morning', 'you', 'greet', 'to', 'me']
```



## 퀴즈 2

---

다음 문장 중 적절하지 않은 것은?

- ① Shallow copy와 deepcopy는 서로 다르다
- ② Shallow copy 복사된 변수들은 메모리를 공유한다
- ③ deep copy 복사된 변수들은 메모리를 공유하지 않는다
- ④ 복사된 2개의 변수를 별개로 처리하려면, shallow copy를 사용한다

## 퀴즈 2-답안

---

다음 문장 중 적절하지 않은 것은?

- ① Shallow copy와 deepcopy는 서로 다르다
- ② Shallow copy 복사된 변수들은 메모리를 공유한다
- ③ deep copy 복사된 변수들은 메모리를 공유하지 않는다
- ④ 복사된 2개의 변수를 별개로 처리하려면, shallow copy를 사용한다

**감사합니다**  
**[2차시]**