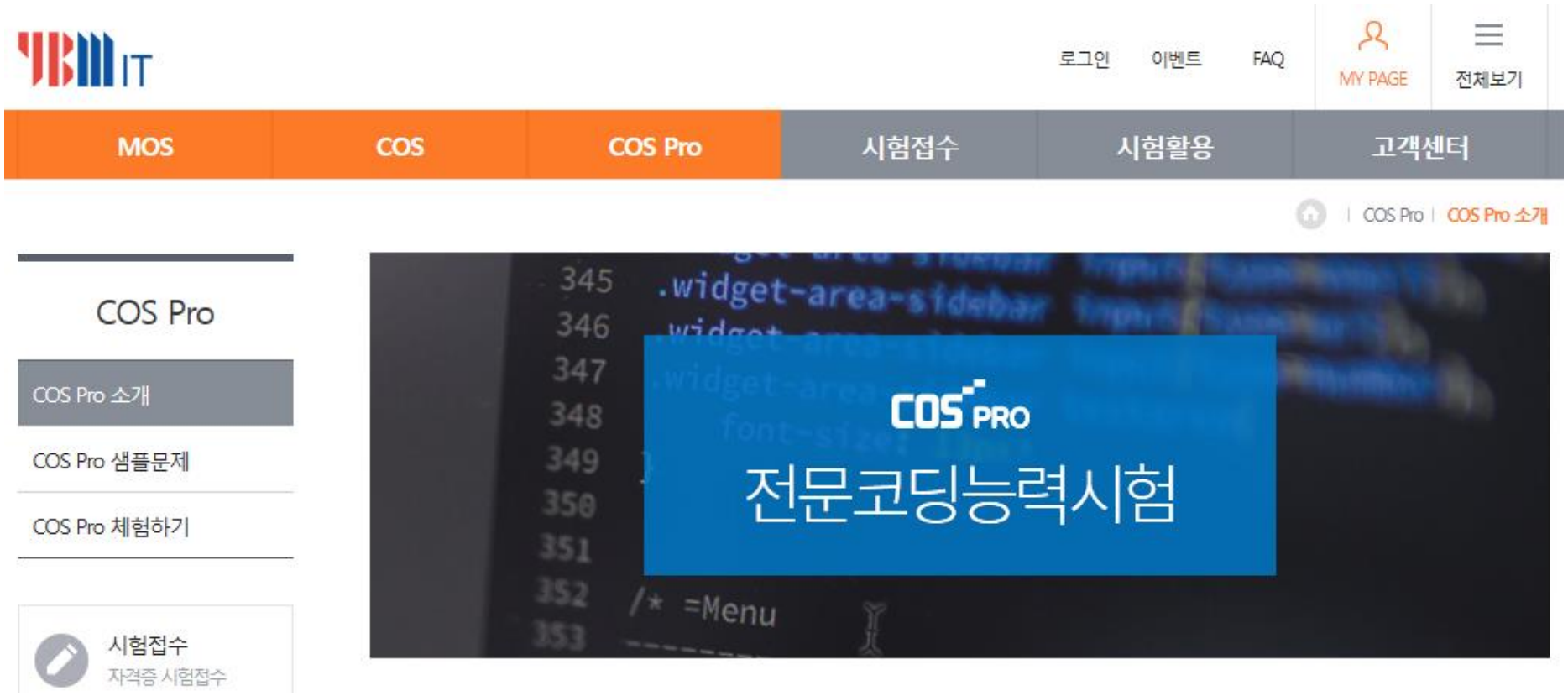
COS-PRO 파이썬

문혜영 교수

목차

시험 안내

- 시험안내
- https://www.ybmit.com



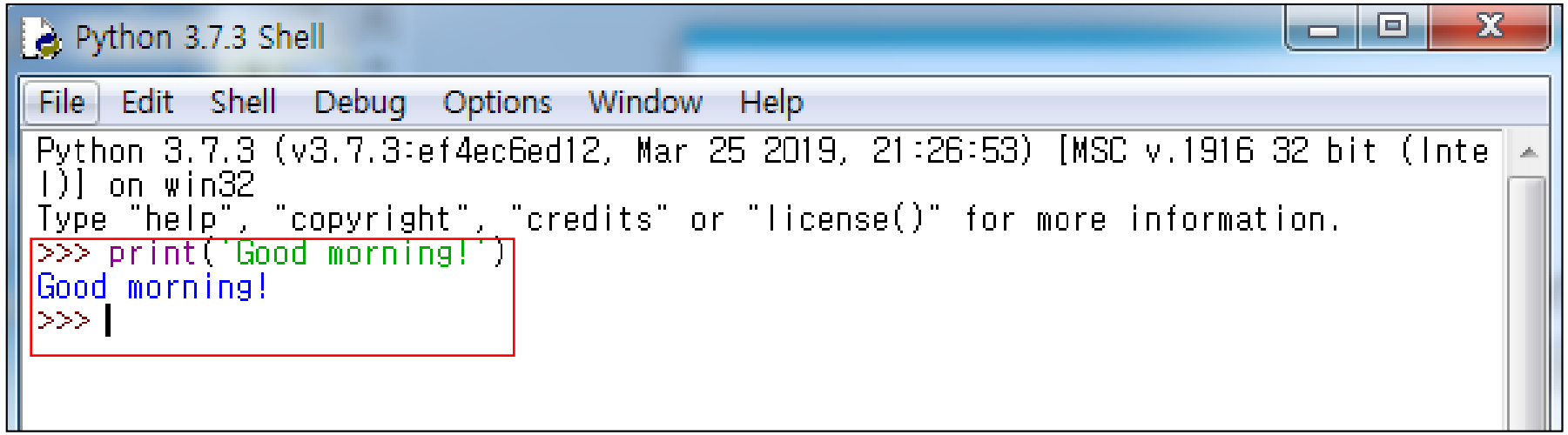
파이썬 설치하기

- https://www.python.org

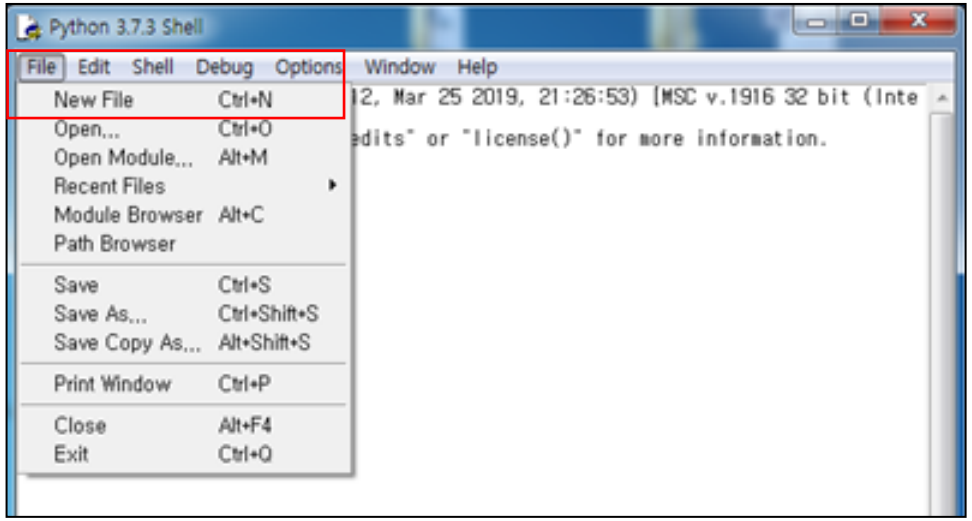
- 파이썬의 특징
  - 쉽고 간단한 문법, 배우기 쉽다.
  - 객체지향
  - 다양한 패키지
  - 오픈 소스, 무료

Good morning!로 시작하기

- IDEL에서 Good morning! 출력해보기
  - ✓IDLE의 >>> 부분에 다음 내용을 입력한 뒤 엔터 키를 누릅니다.



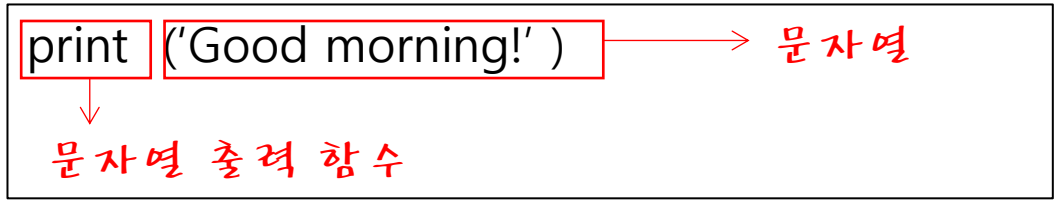
- IDLE에서 소스 파일 실행하기
  - ✓Print('Good morning!') 코드를 파일로 만들어서 실행해보자.
  - ✓IDLE는 파이썬 셀을 통해 코드의 결과를 보는 기능뿐만 아니라 소스 파일을 편집하고 실행도 해 볼 수 있습니다. IDEL의 메뉴에서 File > New File을 선택합니다.



- 명령 프롬프트에서 Good morning! 출력하기
  - ✓IDLE를 사용하지 않고 명령 프롬프트에서 파이썬 셸을 실행한 뒤 Good morning! 를 출력해보겠습니다.
    1.  + R을 누른 뒤 cmd를 입력하여 명령 프롬프트를 실행합니다.
    2. Python을 입력하여 파이썬 셸을 실행합니다.
    3. Print('Good morning!')를 입력한 뒤 엔터 키를 누릅니다.

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:₩Users₩Administrator>python
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 21:26:53) [MSC v.1916 32 bit (Inte
l)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print('Good morning!')
Good morning!
>>> exit()

C:₩Users₩Administrator>
```

- 소스 코드 살펴보기
  - ✓‘ ’(작은 따옴표)로 묶은 부분을 문자열이라 하며, print는 문자열을 화면에 출력합니다.
  - ✓Print처럼 단어 뒤에 ( )(괄호)가 붙은 것을 함수(function)라고 하며, 함수란 정해진 일을 수행하는 단위입니다.



퀴즈

- 다음 중 ‘Good morning!’를 출력하는 방법으로 올바른 것을 고르세요..
  a. Print ‘Good morning!’
  b. Print Good morning!
  c. Print(Good morning!)
  d. Print(‘Good morning!’)
  e. Print[‘Good morning!’]

COS-PRO 파이썬

02 기본 문법

문혜영 교수

목차

- 연습문제: 문자열 출력하기
  - ✓다음 소스 코드를 완성하여 ‘Good morning!’과 ‘1000x2000’이 각 줄에 출력되게 만드세요.

```
Print(‘Good morning!’)
Print( _____ )
```

기본 문법 알아보기

- 세미콜론
  - ✓많은 프로그래밍 언어들은 구문이 끝날 때 ;(세미콜론)을 붙여야 합니다. 하지만 파이선은 세미콜론을 붙이지 않습니다.

```
Print('Good morning!')
```

  - ✓세미콜론을 붙여도 문법 에러는 발생하지 않습니다. 보통 한 줄에 여러 구문을 사용할 때 세미콜론으로 구분해줍니다.

```
Print('Good') ; print('morning!')
```

- 주석
  - ✓파이썬에서 사람만 알아볼 수 있도록 작성하는 부분을 주석(comment)이라고 합니다. 즉, 주석은 파이썬 인터프리터가 처리하지 않으므로 프로그램의 실행에는 영향을 주지 않습니다.

```
# Good morning! 출력
Print('Good morning!')
```

- 들여쓰기
  - ✓들여쓰기는 코드를 읽기 쉽도록 일정한 간격을 띄워서 작성하는 방법입니다. if의 다음 줄은 항상 들여쓰기를 해야 합니다.

```
if a == 20:
Print('20입니다.')      # 들여쓰기 문법 에러
```

```
if a == 20:
    Print('20입니다.')
    ↓
    들여쓰기 공백 4칸
```

- 코드 블록
  - ✓코드 블록은 특정한 동작을 위해서 코드가 모여 있는 상태를 뜻하며, 파이썬은 들여쓰기를 기준으로 코드 블록을 구성합니다.

```
if a == 20:
    print('20')
    print ('입니다.')
```

들여쓰기로 코드 블록을 나타냄

숫자 계산하기

- 정수 계산하기
  - >>>에 3 + 5을 입력한 뒤 엔터 키를 누르면 결괏값 8이 나옵니다.

```
>>> 3+ 5
8
```

- 실수 계산하기
  - ✓실수(소수점이 붙은 수)끼리 계산을 해 보겠습니다.

```
>>> 2.5 + 3.2
5.7
>>> 5.3 - 3.7
1.5999999999999996
>>> 1.5 * 3.1
4.65
>>> 6.6 / 4.1
1.6097560975609757
>>>
```

- 스크립트 파일에서 계산하기
  - ✓파이썬 셀에서는 숫자를 계산한 결과를 바로 출력할 수 있지만, .py 스크립트 파일에서는 계산식을 입력해도 결과가 나오지 않습니다.
  - ✓스크립트 파일에서는 계산 결과를 출력하려면 print 함수를 사용해야 합니다.

```
1 + 2

실행 결과
(아무것도 출력되지 않음)
```

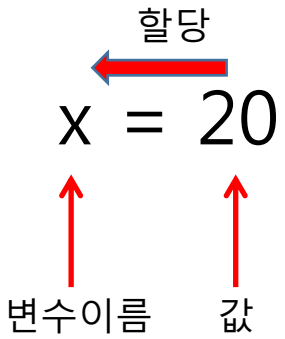
```
Print(1 + 2)

실행 결과
3
```

퀴즈

- sub.py 파일에서 12-3의 결과를 출력하려고 합니다. 올바른 것을 고르세요(파이선 3)
  a. 12-3
  b. (12-3)
  c. Print(12-3)
  d. Print(12)-print(3)
  e. Print 12-3

변수 사용하기

- 변수 만들기
  - ✓파이썬에서는 다음 그림과 같은 형식으로 코드를 입력하여 변수를 만듭니다. (즉, 변수이름 = 값)



- 빈 변수 만들기
  ✓값이 들어있지 않은 빈 변수를 만들려면 None을 할당해주면 됩니다.

```
>>> x = None
>>> x
>>> print(x)
None
```

- 변수의 값 변경하기
  - ✓이미 만들어진 변수의 값을 바꿔보겠습니다.

```
>>> x = 20
>>> x
20
>>> x = 30
>>> x
30
```

- 변수 여러 개를 한 번에 만들기
  - ✓변수 여러 개를 한 번에 만들어 보겠습니다.

```
>>> x, y, z = 10, 30, 50
>>> x
10
>>> y
30
>>> z
50
```

- 변수에 변수 할당하기
  - ✓변수에는 값뿐만 아니라 다른 변수도 할당할 수 있습니다.

```
>>> a, b = 20, 40
>>> x, y = a, b
>>> x
20
>>> y
40
```

퀴즈

- 다음 중 변수를 만드는 방법으로 올바른 것을 고르세요.
  a. int x = 20
  b. x = 20
  c. x. 20
  d. x := 20
  e. x <= 20

- 연습문제: 변수 만들기
  - ✓다음 소스 코드를 완성하여 20, 20, 60, None이 각 줄에 출력되도록 만드세요..

```
a= b = ① _____
c, d = ② _____

Print(a)
Print(b)
Print(c)
print(d)

실행 결과
20
20
60
None
```

COS-PRO 파이썬

03 연산자

문혜영 교수

목차

입력 값을 변수에 저장하기

- 문자열 입력받기
  - ✓input 함수를 사용하여 표준 입력으로 사용자의 입력을 받아보겠습니다.
  - ✓>>>에서 input( )을 입력한 뒤 엔터 키를 누르면 다음 줄로 넘어갑니다. 그리고 Good morning을 입력한 뒤 엔터키를 누르면 입력한 그대로 출력됩니다.

```
>>> input()
Good morning
'Good morning'
```

- 숫자 입력받기
  - ✓입력받은 두 숫자를 더하는 프로그램을 만들어보겠습니다. 다음 내용을 IDLE의 소스 코드 편집 창에 입력하세요.

```
a = input('첫 번째 숫자를 입력하세요:')
b = input('두 번째 숫자를 입력하세요:')

print(a + b)
```

실행 결과

```
첫 번째 숫자를 입력하세요: 2 (입력)
두 번째 숫자를 입력하세요: 3 (입력)
23
```

- 한 번에 값 두 개 입력받기
  - ✓input 한 번에 값을 여러 개 입력받으려면 input에서 split을 사용한 뒤 여러 개의 변수에 저장해 주면 됩니다.(각 변수는 콤마로 구분해줍니다.)

```
a, b = input('문자열 두 개를 입력하세요: ') . Split( )    # 입력받은 값을 공백을 기준으로 분리

print(a)
print(b)
```

실행 결과

```
문자열 두 개를 입력하세요: Good Morning  (입력)
Good
Morning
```

퀴즈

- 다음 중 문자열을 입력받아 변수 s에 저장하는 방법으로 올바른 것을 모두 고르세요.
  a. s = input
  b. s = input('문자열을 입력하세요: ')
  c. s = input( )
  d. input(s1)
  e. input('문자열을 입력하세요: ')

- 연습문제: 한 번에 정수 세 개 입력받기
  - ✓다음 소스 코드를 완성하여 정수 세 개를 입력받고 합계가 출력되게 만드세요.

```
_____

print(a + b + c)


실행 결과
20 30 40
60
```

출력 방법 알아보기

- 값을 여러 개 출력하기
  - ✓print에는 변수나 값 여러 개를 ,(콤마)로 구분하여 넣을 수 있습니다.

```
>>> print(2, 4, 6)
2 4 6
>>> print('Good', 'Morning')
Good Morning
```

- 줄바꿈 활용하기
  - ✓print에 값을 여러 개 지정하면 한 줄에 모든 값이 출력됩니다.
  - ✓print의 sep에 개행 문자(\n)라는 특별한 문자를 지정하면 값을 여러 줄로 추력할 수 있습니다.

```
>>> print(1, 2, 3)
1 2 3
>>> print(1, 2, 3, sep='\n')
1
2
3
```

퀴즈

- 다음 중 2.5 Python 50을 한 줄에 출력하는 방법으로 올바른 것을 고르세요..
  a. print(2.5 Python 50)
  b. print(2.5)
     print('Python')
     print(50)
  c. print(2.5, Python, 50)
  d. print(2.5, 'Python', 50)
  e. print(2.5; 'Python'; 50)

COS-PRO 파이썬

04 리스트

문혜영 교수

목차

문자열 사용하기

- 문자열을 변수에 저장하기
  - ✓변수에 문자열을 할당하면 문자열이 저장됩니다.
  - ✓문자열 'Good morning!'을 변수에 저장하고 출력해보겠습니다.

```
>>> good1 = 'Good morning!'
>>> print(good1)
Good morning!
```

- 문자열의 길이 구하기
  - ✓len 함수를 사용하여 문자열의 길이를 구해보겠습니다.

```
>>> good = 'Good morning!'
>>> len(good)
13
```

- 문자열을 연결하고 반복하기
  - ✓문자열을 연결하는 방법과 반복하는 방법을 알아보겠습니다. 문자열에 +를 사용하면 문자열이 연결되고, *를 사용하면 문자열이 반복된다.

```
>>> hello = 'Hello, '
>>> world = 'world!'
>>> hello + world
'Hello, world!'
>>> hello * 3
'Hello, Hello, Hello, '
```

퀴즈

- 다음 중 문자열의 길이를 구하는 함수를 고르세요.
  a. length
  b. print
  c. encode
  d. len
  e. int

- 연습문제: 문자열의 길이 출력하기
  - ✓다음 소스 코드를 완성하여 문자열의 길이가 출력되게 만드세요.

```
s = 'Good morning'
print( _____ )
```

실행 결과
s의 길이는 12입니다.

리스트와 튜플 사용하기

- 리스트 만들기
  - ✓변수에 값을 저장할 때 [ ](대괄호)로 묶어주면 리스트가 되며, 각 값은 ,(콤마)로 구분해줍니다.

```
>>> a = [30, 25, 50, 35, 40]
>>> a
[30, 25, 50, 35, 40]
```

- 리스트의 요소 접근하고 값 할당하기
  - ✓리스트에 저장된 각 값을 요소(element)라고 합니다. 요소에 접근할 때는 리스트 뒤에 [ ](대괄호)를 사용하여 [ ] 안에 각 요소의 인덱스를 지정해주면 됩니다.

```
>>> a = [10, 20, 30, 40, 50]
>>> a[0]        # 리스트의 첫 번째(인덱스 0) 요소 출력
10
>>> a[2]        # 리스트의 세 번째(인덱스 2) 요소 출력
30
>>> a[4]        # 리스트의 다섯 번째(인덱스 4) 요소 출력
50
```

- 리스트의 요소 개수 구하기
  - ✓요소의 개수는 함수를 사용하여 구합니다.
  - ✓len(a)와 같이 len에 리스트 변수를 넣어서 요소의 개수를 구해도 되고, len에 리스트를 그대로 넣어도 됩니다.

```
>>> a = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
>>> len(a)
10
>>> len([10, 20, 30, 40, 50, 60, 70, 80, 90, 100])
10
```

- 리스트의 인덱스 활용하기
  - ✓인덱스로 범위를 지정하여 리스트의 일부만 가져와보겠습니다.

```
>>> a = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
>>> a[0:4]
[10, 20, 30, 40]
>>> a[0:10]
[10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
```

- 튜플 사용하기
  - ✓변수에 값을 저장할 때 ( )(괄호)로 묶어주면 튜플이 되며, 각 값은 ,(콤마)로 구분해줍니다. 또는 괄호를 묶지 않고 값만 콤마로 구분해도 튜플이 됩니다.

```
>>> a = (10, 20, 30, 40, 50)
>>> a
(10, 20, 30, 40, 50)
```

퀴즈

- 리스트 a = [11, 22, 33, 44, 55, 66]에서 인덱스로 요소를 가져왔을 때 값이 올바르지 않은 것을 모두 고르세요.
  a. a[0]은 11
  b. a[1]은 11
  c. a[3]은 44
  d. a[-1]은 55
  e. a[-1]은 66

- 연습문제: 리스트의 일부만 가져오기
  - ✓리스트 a와 b 두 개가 주어집니다. 다음 소스 코드를 완성하여 a는 세 번째 요소부터 마지막 요소까지, b는 첫 번째 요소부터 네 번째 요소까지 출력되게 만드세요. 출력 결과는 리스트 형태여야 합니다.

```
A = ['one', 'two', 'three', 'four', 'five', 'six']
B = [1000, 2000, 3000, 4000, 5000, 6000]

Print(① _____ )
Print(② _____ )
```

실행 결과

```
['three', 'four', 'five', 'six']
[1000, 2000, 3000, 4000]
```

COS-PRO 파이썬

05 리스트 응용

문혜영 교수

목차

리스트와 튜플 응용하기

- 리스트 조작하기
  - ✓리스트를 조작하는 매서드(method)를 설명하겠습니다.
  - ✓append(값)은 이름 그대로 리스트의 맨 뒤에 값을 추가합니다. 다음은 리스트 [20, 40]에 60을 추가하여 리스트는 [20, 40, 60]이 됩니다.

```
>>> a = [20, 40]
>>> a.append(60)
>>> a
[20, 40, 60]
```

- 리스트의 할당과 복사
  - ✓할당과 복사는 비슷한 것 같지만 큰 차이점이 있습니다.

<할당>

```
>>> a = [10, 10, 10, 10, 10]
>>> b = a
>>> b[2] = 99
>>> a
[10, 10, 99, 10, 10]
>>> b
[10, 10, 99, 10, 10]
```

<복사>

```
>>> a = [10, 10, 10, 10, 10]
>>> b = a.copy( )
>>> b[2] = 99
>>> a
[10, 10, 10, 10, 10]
>>> b
[10, 10, 99, 10, 10]
```

- 리스트 연산하기
  - ✓리스트에 덧셈과 곱셈 연산자를 사용해보겠습니다.
  - ✓+ 연산자는 리스트를 서로 연결하며 extend와 동작이 같습니다.
  - ✓* 연산자는 특정 횟수만큼 리스트의 요소를 반복합니다. (0 또는 음수를 곱하면 빈 리스트가 나오며, 실수는 곱할 수 없습니다.

```
>>> a = [1, 2, 3]
>>> b = [10, 20, 30]
>>> c = a + b
>>> c
[1, 2, 3, 10, 20, 30]
```

```
>>> a = [11, 22, 33]
>>> c = a * 3
>>> c
[11, 22, 33, 11, 22, 33, 11, 22, 33
```

- 반복문으로 리스트의 요소를 모두 출력하기
  - ✓리스트와 for 반복문을 사용하여 간단하게 모든 요소를 출력해보겠습니다. for in 뒤에 리스트를 지정하면 됩니다.

```
for 변수 in 리스트:
    반복할 코드
```

들여쓰기 나란

```
>>> a = [10, 20, 30, 40, 50]
>>> for i in a:
        print(i)

10
20
30
40
50
```

- 연습문제: 리스트에서 특정 요소만 뽑아내기
  - ✓다음 소스 코드를 완성하여 리스트 a에 들어있는 문자열 중에서 길이가 4인 것들만 출력되게 만드세요.

```
a = ['banana', 'bravo', 'piano', 'golf', 'chart', 'integer', 'good', 'sort', 'hotel']




실행 결과
golf
good
sort
```

- 리스트의 가장 작은 수, 가장 큰 수, 합계 구하기..

가장 작은수 구하기

```
>>> a = [30, 20, 50, 60, 10]
>>> smallest = a[0]
>>> for i in a:
        if i < smallest:
            smallest = i

>>> smallest
10
```

가장 큰 수 구하기

```
>>> a = [30, 20, 50, 60, 10]
>>> largest = a[0]
>>> for i in a:
        if i > largest:
            largest = i

>>> largest
60
```

합계 구하기

```
>>> a = [30, 20, 20, 30, 20]
>>> x = 0
>>> for i in a:
        x += i

>>> x
120
```

```
>>> a = [30, 20, 50, 60, 10]
>>> min(a)
10
>>> max(a)
60
```

```
>>> a = [30, 20, 20, 30, 20]
>>> sum(a)
120
```

COS-PRO 파이썬

06 2차원 리스트

문혜영 교수

목차

2차원 리스트와 튜플 사용하기

- 2차원 리스트를 만들고 요소에 접근하기
  - ✓2차원 리스트는 리스트 안에 리스트를 넣어서 만들 수 있으며, 안쪽의 각 리스트는 ,(콤마)로 구분합니다.

```
>>> a = [[10, 15], [20, 25], [30, 35]]
>>> a
[[10, 15], [20, 25], [30, 35]]
>>> a[0][0]      # 세로 인덱스 0, 가로 인덱스 0인 요소 출력
10
>>> a[1][1]      # 세로 인덱스 1, 가로 인덱스 1인 요소 출력
25
>>> a[2][1]      # 세로 인덱스 2, 가로 인덱스 1인 요소 출력
35
>>> a[0][1] = 300      # 세로 인덱스 0, 가로 인덱스 1인 요소에 값 할당
>>> a[0][1]
300
```

가로 인덱스

세로 인덱스

```
A = [[10, 15],
     [20, 25],
     [30, 35]]
```

- 반복문으로 2차원 리스트의 요소를 모두 출력하기
  - ✓ 반복문을 사용하여 2차원 리스트의 요소를 모두 출력해보겠습니다.

for 반복문을 한 번만 사용하는 방식

```
>>> a = [[20, 30], [40, 50], [60, 70]]
>>> for x, y, in a:
        print(x, y)

20 30
40 50
60 70
```

for 반복문을 두 번 사용하는 방식(다음 내용을 IDLE의 소스 코드 편집창에 입력한 뒤 실행)

```
a = [[20, 30], [40, 50], [60, 70]]

for i in a:
    for j in i:
        print(j, end=' ')
    print( )
```

```
실행 결과
20 30
40 50
60 70
```

- 반복문으로 2차원 리스트 만들기

```
a= [ ]

for i in range(3):
    line = [ ]
    for j in range(2):
        line.append(0)
    a.append(line)

print(a)
```

실행 결과

```
[[0, 0], [0, 0], [0,0]]
```

퀴즈

- 다음 중 리스트 a = [[60, 50, 40], [30, 20, 10]]에서 10을 출력하는 방법으로 올바른 것을 고르세요..
  a. Print(a[0][1])
  b. Print(a[0][2])
  c. Print(a[1][2])
  d. Print(a[1][3])
  e. Print(a[2][3])

COS-PRO 파이썬

07 조건문

문혜영 교수

목차

- 연습문제: 합격 여부 출력하기
  - ✓국어, 영어, 수학, 과학 점수가 있을 때 한 과목이라도 60점 미만이면 불합격이라고 정했습니다. 다음 소스 코드를 완성하여 합격이면 True, 불합격이면 False가 출력되게 만드세요.

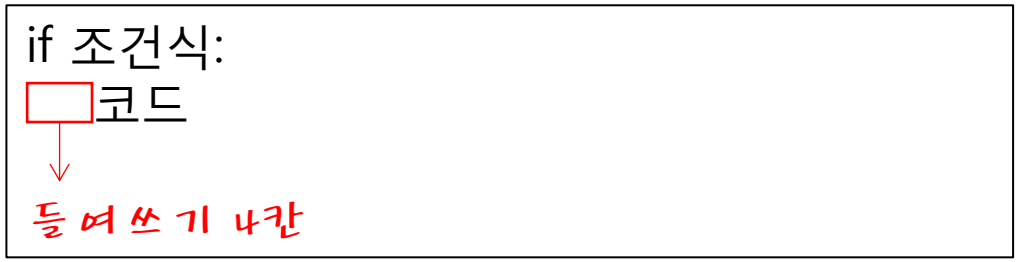
```
kor = 88
eng = 56
math = 95
scien = 76

print( _____ )
```

실행 결과

```
False
```

if 조건문으로 특정 조건일 때 코드 실행하기

- if 조건문 사용하기
  - ✓if 조건문은 if 다음에 조건식을 지정하고 :(콜론)을 붙이며 다음 줄에 실행할 코드가 옵니다. 이때 실행할 코드는 반드시 들여쓰기를 해야 합니다.



- if 조건문과 들여쓰기
  - ✓파이썬은 들여쓰기도 문법으로 정해져 있으며, if 조건문도 들여쓰기가 중요합니다.

```
x = 20

if x == 20:
    print('x에 들어있는 숫자는')
        print('20입니다. ')
```

들여쓰기 4칸

unexpected indent 에러 발생

들여쓰기 8칸

```
x = 20

if x == 20:
    print('x에 들어있는 숫자는')
    print('20입니다. ')
```

들여쓰기 4칸

실행 결과

X에 들어있는 숫자는
20입니다.

- 중첩 if 조건문 사용하기
  - ✓if를 여러 번 사용하는 중첩 if 조건문을 사용해보겠습니다. 다음은 변수의 값이 20 이상이면 '20 이상입니다. '를 출력한 뒤 30이면 '30입니다. ', 40이면 '40입니다. '를 출력합니다.

```
x = 30

if x >= 20:
    print('20 이상입니다. ')
    if x == 30:
        print('30입니다. ')
    if x == 40:
        print('40입니다. ')
```

들여쓰기 4칸

들여쓰기 8칸

들여쓰기 4칸

들여쓰기 8칸

실행 결과

```
20 이상입니다.
30입니다.
```

- 사용자가 입력한 값에 if 조건문 사용하기
  - ✓input을 사용하여 사용자가 입력한 값을 변수에 저장하고, if 조건문으로 값을 비교해 보겠습니다. 다음 내용을 IDEL의 소스 코드 창에 입력하세요.

```
x = int(input( ))              # 입력받은 값을 변수에 저장

if x == 20:                    # x가 20이면
    print('20입니다. ')        # '20입니다. '를 출력

if x == 30:                    # x가 30이면
    print('30입니다. ')        # '30입니다. '를 출력

실행 결과
20 (입력)
20입니다.
```

퀴즈

- 다음 중 if 조건문의 사용 방법으로 올바른 것을 고르시오.
  a. if (x == 20)
     print('20입니다. ')
  b. if x == 20
     print('20입니다. ')
  c. if x == 20:
     print('20입니다. ')
  d. if x == 20:
  print('20입니다. ')
  e. if x = 20:
     print('20입니다. ')

- 연습문제: if 조건문 사용하기
  - ✓다음 소스 코드를 완성하여 x의 값이 20과 다를 때(20이 아닐 때) 'ok'가 출력되게 만드세요.

```
x = 10

if _____ :
    print('ok')


실행 결과
ok
```

Else를 사용하여 두 방향으로 분기하기

- else 사용하기
  - ✓else는 if 조건문 뒤에 오며 단독으로 사용할 수 없습니다. 그리고 if와 마찬가지로 else도 :(콜론)을 붙이며 다음 줄에 실행할 코드가 옵니다.

```
if 조건식:
    코드1
else:
    코드2
```

들여쓰기 4칸

- else와 들여쓰기
  - ✓else는 if와 들여쓰기 규칙이 같습니다.

```
X = 10

if x == 20:
    print('20입니다. ')
else:
print('x에 들어있는 숫자는')          # unexpected indent 에러 발생
    print(''20이 아닙니다. ')
```

  - ✓else도 코드가 여러 줄일 때는 들여쓰기 깊이가 같게 만들어 주어야 합니다.

```
x = 10

if x == 20:
    print('20입니다. ')
else:
    print('x에 들어있는 숫자는')
    print(''20이 아닙니다. ')
```

- if 조건문의 동작 방식 알아보기
  - ✓조건식이 아닌 값으로 if와 else의 코드를 동작시켜 보겠습니다. 다음 내용을 IDEL의 소스 코드 편집 창에 입력한 뒤 실행해보세요.

```
if True:
    print('참')          # True는 참
else:
    print('거짓')

if False:
    print('참')          # False는 거짓
else:
    print('거짓')

if None:
    print('참')          # None은 거짓
else:
    print('거짓')
```

실행 결과
참
거짓
거짓

- 조건식을 여러 개 지정하기
  - ✓if 조건문에는 논리 연산자를 사용하여 조건식을 여러 개 지정할 수 있습니다. 다음 내용을 IDLE의 소스 코드 편집 창에 입력한 뒤 실행해보세요.

```
x = 20
y = 30

if x == 20 and y ==30:          # x가 20이면서 y가 30일 때
    print('참')
else:
    print('거짓')
```

실행 결과

```
참
```

퀴즈

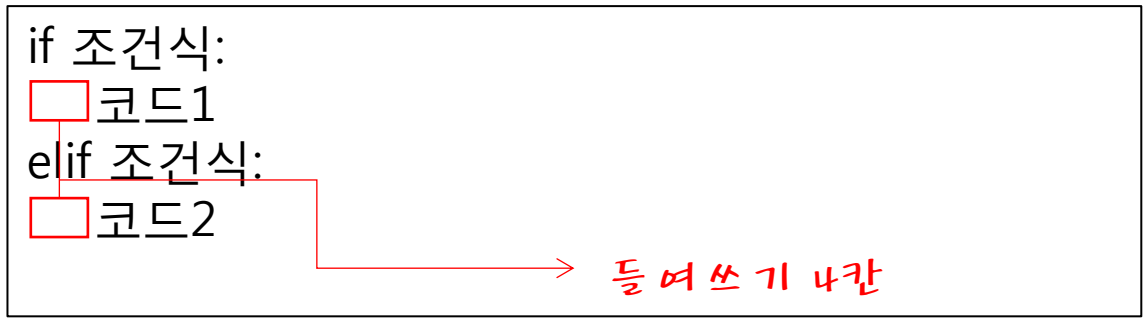
- 다음 중 if 조건문에 대한 설명으로 올바른 것을 고르세요.
  a. if의 코드는 조건식이 만족하지 않을 때 실행된다.
  b. else의 코드는 조건문이 참일 때 실행된다.
  c. else는 단독으로 사용할 수 없다.
  d. else에서 실행되는 코드는 다음 줄에서 들여쓰기를 하지 않아야 한다.
  e. if는 항상 else가 있어야 한다.

- 연습문제: 합격 여부 판단하기
  - ✓A 기업은 입사 시험에서 필기 시험 점수가 85점 이상이면서 토익 점수가 900점 이상이라야 합경이라고 기준을 정했습니다. 다음 소스 코드를 완성하여 '합격', '불합격'이 출력되게 만드세요.

```
writtenTest = 80
toeic = 920


①   writtenTest                    toeic          :
    print('합격')
else:
    print('불합격')


실행 결과
불합격
```

elif를 사용하여 여러 방향으로 분기하기

- elif 사용하기
  - ✓elif는 else인 상태에서 조건식을 지정할 때 사용하며 else if라는 뜻입니다. 물론 if, else와 마찬가지로 조건식 뒤에 :(콜론)을 붙여야 하고, elif 단독으로 사용할 수 없습니다.



- if, elif, else를 모두 사용하기
  - ✓elif는 else와 함께 사용할 수 있습니다.

```
if 조건식:
    코드1
elif 조건식:
    코드2
else:
    코드3
```

퀴즈

- 다음 중 if 조건문에 대한 설명으로 잘못된 것은?
  a. elif는 여러 번 사용할 수 있다.
  b. else는 elif 앞에 올 수 없다.
  c. elif는 조건식을 지정할 수 있다.
  d. elif는 단독으로 사용할 수 있다.
  e. elif는 항상 else가 있어야 한다.

- 연습문제: if, elif, else 모두 사용하기
  - ✓다음 소스 코드를 완성하여 변수 x가 1과 10 사이면 '1~10', 11과 20 사이면 '11~20', 아무것도 해당하지 않으면 '아무것도 해당하지 않음'이 출력되게 만드세요..

```
x = 30

_____

_____

_____

_____

_____

_____

실행 결과
아무것도 해당하지 않음
```

COS-PRO 파이썬

08 for 반복문

문혜영 교수

목차

for 반복문으로 Hello, world! 10번 출력하기

- for와 range 사용하기
  - ✓for 반복문은 range에 반복할 횟수를 지정하고 앞에 in과 변수를 입력합니다. 그리고 끝에 :(콜론)을 붙인 뒤 다음 줄에 반복할 코드를 넣습니다.
  - ✓for 다음 줄에 오는 코드는 반드시 들여쓰기를 해줍니다.(들여쓰기 규칙은 if, elif, else와 같습니다.)

```
For 변수 in range(횟수):
    반복할 코드
```

- 반복문에서 변수의 변환 알아보기
  - ✓range에서 꺼낸 숫자를 눈으로 확인해보겠습니다.

```
>>> for i in range(10):
        print('Hello, world!', i)


Hello, world! 0
Hello, world! 1
Hello, world! 2
Hello, world! 3
Hello, world! 4
Hello, world! 5
Hello, world! 6
Hello, world! 7
Hello, world! 8
Hello, world! 9
```

- 숫자 범위와 증가 폭 지정하기
  - ✓range는 기본적으로 0부터 시작하지만 시작하는 숫자와 끝나는 숫자를 지정할 수도 있습니다.

```
>>> for i in range(5, 10):                # 5부터 9까지 반복
        print('Good morning!', i)

Good morning!  5
Good morning!  6
Good morning!  7
Good morning!  8
Good morning!  9
```

- 숫자를 감소시키기
  - ✓for와 range를 사용하면 숫자가 증가하면서 반복합니다. 하지만 숫자를 역순으로 생성하려면 증가 폭을 음수로 지정하면 됩니다.

```
>>> for i in range(5, 0, -1):                # 5에서 1까지 역순으로 숫자 생성
        print('Good morning!', i)

Good morning! 5
Good morning! 4
Good morning! 3
Good morning! 2
Good morning! 1
```

- 입력한 횟수대로 반복하기
  - ✓입력한 횟수대로 반복을 해보겠습니다. 다음 내용을 IDLE의 소스 코드 편집 창에 입력하세요.

```
count = int(input('반복할 횟수를 입력하세요: '))

for i in range(count):
    print('Good morning!', i)
```

실행 결과

```
반복할 횟수를 입력하세요: 3 (입력)
Good morning! 0
Good morning! 1
Good morning! 2
```

퀴즈

- 다음 중 for로 10번 반복하는 방법으로 올바른 것을 모두 고르세요.
  a. for i in range(10):
  b. for i in range(5, 16):
  c. for i in range(10, 0):
  d. for i in range(20, 40, 2):
  e. for i in range(1, 10, 1):

- 연습문제: 팩토리얼 구하기
  - ✓다음 소스 코드를 완성하여 n의 패토리얼(factorial)이 출력되게 만드세요. 팩토리얼은 1부터 n까지 숫자를 차례대로 곱한 값입니다..

```
n = 4
factorial = 1


_____________________________________

_____________________________________

print(factorial)
```

실행 결과

24

COS-PRO 파이썬

09 while 반복문

문혜영 교수

목차

while 반복문으로 hello, world! 10번 출력하기

- while 반복문은 조건식을 지정하고 끝에 :(콜론)을 붙인 뒤 다음 줄에 반복할 코드와 변화식을 넣습니다. 초기식은 특별한 것이 없고 보통 변수에 값을 저장하는 코드입니다.

- 초깃값을 1부터 시작하기
  - ✓i에 0이 아닌 1을 할당하여 ‘Hello, world!’를 10번 출력해보겠습니다.

```
>>> i = 1
>>> while i <= 10:
        print('Good morning!', i)
        i +=1

Good morning! 1
Good morning! 2
Good morning! 3
... (생략)
Good morning! 8
Good morning! 9
Good morning! 10
```

- 초깃값 감소시키기
  - ✓초깃값을 크게 주고, 변수를 감소시키면서 반복할 수도 있습니다. 다음은 10부터 1까지 10번 반복합니다.

```
>>> i = 10
>>> while i > 0:
        print('Good morning!', i)
        i -= 1

Good morning! 10
Good morning! 9
Good morning! 8
...(생략)
Good morning! 2
Good morning! 1
```

- 입력한 횟수대로 반복하기
  - ✓입력한 횟수대로 반복을 해보겠습니다. 다음 내용을 IDLE의 소스 코드 편집 창에 입력하세요.

```
count = int(input('반복할 횟수를 입력하세요: '))

i = 0
while i < count:
    print('Good morning! %d' % i)
    i += 1
```

실행 결과

```
반복할 횟수를 입력하세요: 3(입력)
Good morning! 0
Good morning! 1
Good morning! 2
```

- 반복횟수가 정해지지 않은 경우
  - ✓while 반복문은 반복 횟수가 정해지지 않았을 때 주로 사용합니다.
  - ✓다음은 while 반복문 안에서 무작위로 정수를 생성한 뒤 5가 나오면 반복을 끝냅니다.

```
import random                    # random 모듈을 가져옴

i = 0
while i != 5:                    # 5가 아닐 때 계속 반복
    i = random.randint(0, 9)     # randint를 사용하여 0부터 9까지 무작위로 정수를 생성
    print(i)
```

실행 결과

```
7
3
1
5
```

→ 무작위 생성이므로 실행할 때마다 달라짐

퀴즈

- 다음 중 while 반복문에 대한 설명으로 잘못된 것을 모두 고르세요.
  a. while 반복문에는 조건식 또는 값을 지정하면 된다.
  b. while 반복문은 반드시 변화식이 있어야 한다.
  c. while 반복문은 반복 횟수가 정해져 있을 때만 사용할 수 있다.
  d. while 반복문의 다음 줄은 반드시 들여쓰기를 해야 한다.
  e. while 반복문의 조건식에 True를 지정하면 무한 루프가 된다.

- 연습문제: 변수 두 개를 다르게 반복하기
  - ✓다음 소스 코드를 완성하여 정수 2 5, 4 4, 8 3, 16 2, 32 1이 각 줄에 출력되게 만드세요.

```
i = 2
j = 5

① _____
print(i, j)
② _____

  _____
```

break와 continue로 반복문 제어하기

- break로 반복문 끝내기
  - ✓무한 루프에서 숫자를 증가시키다가 10이 나오면 반복문을 끝내도록 만들어보겠습니다. 다음 내용을 IDLE의 소스 코드 편집 창에 입력한 뒤 실행해보세요.

```
i = 0
while True:          # 무한 루프
    i += 1           # i를 1씩 증가시킴
    print(i)
    if i ==10:       # i가 10일 때
        break        # 반복문을 끝냄. While의 제어 흐름을 벗어남

실행 결과
...(생략)
8
9
10
```

- continue로 코드 실행 건너뛰기
  - ✓continue를 사용하여 일부 코드를 실행하지 않고 건너뛰어 보겠습니다. 다음은 1부터 10까지 숫자 중에서 짝수만 출력합니다.

```
for i in range(1, 11):
    if i % 2 != 0:
        continue
    print(i)
```

실행 결과

```
2
4
6
8
10
```

COS-PRO 파이썬

10 FizzBuzz

문혜영 교수

목차

FizzBuzz

- 1부터 50까지 숫자 출력하기
  - ✓FizzBuzz 문제는 반복문, 조건문, 나머지 연산자, 비교 연산자를 모두 동원해야 풀 수 있습니다. 1부터 50까지 숫자를 출력해보겠습니다.

```
for i in range(1, 51):          # 1부터 50까지 50번 반복
    print(i)


실행 결과
... (생략)
45
46
47
48
49
50
```

- 3의 배수일 때와 5의 배수일 때 처리하기
  - ✓3의 배수와 5의 배수일 때 숫자 대신 'Fizz'와 'buzz'를 출력해보겠습니다.

```
for i in range(1, 51):      # 1부터 50까지 50번 반복
    if i % 3 ==0:           # 3의 배수일 때
        print('Fizz')       # Fizz 출력
    elif i % 5 == 0:        # 5의 배수일 때
        print('Buzz')       # Buzz 출력
    else:
        print(i)            # 아무것도 해당하지 않을 때 숫자 출력
```

실행 결과
… (생략)
Fizz
46
47
Fizz
49
Buzz

- 3과 5의 공배수 처리하기
  - ✓3과 5의 공배수는 다음과 같이 논리 연산자 and를 사용하면 됩니다.

```
for i in range(1, 51):                  # 1부터 50까지 50번 반복
    if i % 3 == 0 and i % 5 == 0:       # 3과 5의 공배수일 때
        print('FizzBuzz')               # FizzBuzz 출력
    elif i % 3 == 0:                    # 3의 배수일 때
        print('Fizz')                   # Fizz 출력
    elif i % 5 == 0:                    # 5의 배수일 때
        print('Buzz')                   # Buzz 출력
    else:
        print(i)                        # 아무것도 해당하지 않을 때 숫자 출력
```

실행 결과

```
... (생략)
FizzBuzz
46
47
Fizz
49
Buzz
```

- 논리 연산자를 사용하지 않고 3과 5의 공배수 처리하기
  - ✓3 * 5 = 15는 3과 5의 최소공배수이므로 15로 나눴을 때 나머지가 0인 값들은 3과 5의 공배수입니다.

```
for i in range(1, 51):                  # 1부터 50까지 50번 반복
    if i % 15 == 0:                     # 15의 배수(3과 5의 공배수)일 때
        print('FizzBuzz')               # FizzBuzz 출력
    elif i % 3 == 0:                    # 3의 배수일 때
        print('Fizz')                   # Fizz 출력
    elif i % 5 == 0:                    # 5의 배수일 때
        print('Buzz')                   # Buzz 출력
    else:
        print(i)                        # 아무것도 해당하지 않을 때 숫자 출력
```

실행 결과
... (생략)
FizzBuzz
46
47
Fizz
49
Buzz

- 입력한 횟수대로 반복하기
  - ✓입력한 횟수대로 반복을 해보겠습니다. 다음 내용을 IDLE의 소스 코드 편집 창에 입력하세요.

```
count = int(input('반복할 횟수를 입력하세요: '))

i = 1
while True:              # 무한 루프
    print(i)
    if i == count:       # i가 입력받은 값과 같을 때
        break            # 반복문을 끝냄
    i += 1
```

실행 결과

```
반복할 횟수를 입력하세요: 2(입력)
1
2
```

- 입력한 숫자까지 짝수 출력하기
  - ✓입력한 숫자가지 해당하는 짝수를 출력해보겠습니다.

```
count = int(input('반복할 횟수를 입력하세요: '))

for i in range(1, count + 1);     # 1부터 증가하면서 count까지 반복(count + 1)
    if i % 2 != 0:                # i를 2로 나누었을 때 나머지가 0이 아니면 홀수
        continue                  # 아래 코드를 실행하지 않고 건너뜀
    print(i)

실행 결과
반복할 횟수를 입력하세요: 10(입력)
2
4
6
8
10
```

퀴즈

- 다음 중 break와 continue에 대한 설명으로 올바른 것을 모두 고르세요.
  a. break는 if 조건문을 끝낸다.
  b. break는 while 반복문을 끝낸다.
  c. break는 for range 반복문에 사용할 수 없다.
  d. continue는 코드를 실행하지 않고 건너뛰며 루프를 중단하지 않는다.
  e. continue는 코드를 실행하지 않고 건너뛴 뒤 루프를 중단한다.

- 연습문제: 5로 끝나는 숫자만 출력하기
  - ✓다음 소스 코드를 완성하여 1과 50 사이의 숫자 중 5로 끝나는 숫자만 출력되게 만드세요.

```
i = 0
while True:
  ① _____
     _____
     _____
  ② _____
     _____
    print(i, end= ' ')
    i += 1
```

실행 결과

```
5 15 25 35 45
```

퀴즈

- 다음 중 변수 i가 7의 배수인지 확인하는 방법으로 올바른 것을 고르세요.
  a. i / 7 == 0
  b. i | 7 == 0
  c. i % 7 == 0
  d. i * 7 == 0
  e. i & 7 == 0

- 연습문제: 3과 7의 배수, 공배수 처리하기
  - ✓다음 소스 코드를 완성하여 1부터 50까지의 숫자를 출력하면서 3의 배수일 때는 'Fizz', 7의 배수일 때는 'Buzz', 3과 7의 공배수일 때는 'FizzBuzz'가 출력되게 만드세요..

```
for i in range(1, 51):
    if  ① _____ :
        print('FizzBuzz')
    elif ②_____ :
        print('Fizz')
    elif ③_____ :
        print('Buzz')
    else:
        print(i)
```

실행 결과

```
... (생략)
FizzBuzz
43
44
Fizz
46
47
Fizz
Buzz
50
```

COS-PRO 파이썬

11 N-gram,함수

문혜영 교수

목차

회문 판별과 N-gram 만들기

- 회문 판별하기
  - ✓회문(palindrome)은 순서를 거꾸로 읽어도 제대로 읽은 것과 같은 단어와 문장을 말합니다. (예, ‘level’, ‘sos’, ‘roraror’ 등)

```
word = input('단어를 입력하세요: '

tf = True                    # 회문 판별값을 저장할 변수, 초깃값은 True
for i in range(len(word) // 2):      # 0부터 문자열 길이의 절반만큼 반복
    if world[i] != word[-1 – i]:      # 왼쪽 문자와 오른쪽 문자를 비교하여 문자가 다르면
        tf = False          # 회문이 아님
        break

Print(tf)                    # 회문 판별값 출력
```

실행 결과

```
단어를 입력하세요: level (입력)
True
```

- N-gram 만들기
  - ✓N-gram은 문자열에세 N개의 연속된 요소를 추출하는 방법입니다.
  - ✓'Good'이라는 문자열을 문자(글자) 단위 2-gram으로 추출하면 문자열의 처음부터 문자열 끝까지 한 글자씩 이동하면서 2글자를 추출하는데 파이썬에서 문자 단위 2-gram을 출력해 보겠습니다.

```
text = 'Good'

for i in range(len(text) – 1):                # 2-gram이므로 문자열의 끝에서 한 글자 앞까지 반복함.
    print(text[i], text[i + 1], sep = '  ')   # 현재 문자와 그 다음 문자를 출력
```

실행 결과

```
G o
o o
o d
```

함수 사용하기

- Hello, world! 출력 함수 만들기
  - ✓함수는 def에 함수 이름을 지정하고 ( )(괄호)와 :(콜론)을 붙인 뒤 다음 줄에 원하는 코드를 작성하면 됩니다.(함수의 이름을 짓는 방법은 변수와 같습니다.) 이때 코드는 반드시 들여쓰기를 해야 합니다.(들여쓰기 규칙은 if, for, while과 같습니다.)

```
def 함수이름( ):
    코드
```

들여쓰기 나란

```
>>> def hello():
        print('hello, world!')

>>> hello()
hello, world!
```

들여쓰기 나란

빈 줄에서 엔터 키를 누름

- 덧셈 함수 만들기
  - ✓함수에서 값을 받으려면 ( )(괄호) 안에 변수 이름을 지정해주면 됩니다. 이렇게 받은 값을 저장하는 변수를 매개변수(parameter)라고 부릅니다.

```
>>> 함수이름(매개변수1, 매개변수2):
        코드
```

  - ✓두 수를 더하는 함수

```
>>> def add(a, b):
        print(a+b)

>>> add(20, 30)
50
```

- 함수의 결과를 반환하기
  - ✓add 함수의 결과를 반환하는 방법을 알아보겠습니다. 다음과 같이 함수 안에서 return을 사용하면 값을 함수 바깥으로 전달할 수 있습니다.

```
def 함수이름(매개변수):
    return 반환값
```

  - ✓두 수를 더해서 반환하는 add 함수

```
>>> def add(a,b):
        return a+b

>>> x = add(20, 30)
>>> x
50
```

- 함수에서 값을 여러 개 반환하기
  - ✓함수에서 값을 여러 개 반환하는 방법

```
def 함수이름(매개변수):
    return 반환값1, 반환값2
```

  - ✓두 수를 더한 값과 뺀 값을 반환하는 함수

```
def add_sub(a,b):
    return a+b, a-b

>>> x, y = add_sub(20,30)
>>> x
50
>>> y
-10
```

- 함수의 호출 과정 알아보기
  - ✓덧셈 함수 add와 곱셈 함수 mul이 있고, add 함수 안에서 mul 함수를 호출하는 방식으로 만들어 보겠습니다.

```
Def mul(a, b):
    c = a * b
    return c

Def add(a, b):
    c = a + b
    print(c)
    d = mul(a, b)
    print(d)

X = 20
Y = 30
Add(x, y)
50
600
```

퀴즈

- 다음 중 매개변수가 없는 good 함수를 호출하는 방법으로 올바른 것을 고르세요.
  a. def good
  b. good
  c. good( )
  d. good[ ]
  e. def good:

- 연습문제: 몫과 나머지를 구하는 함수 만들기
  - ✓다음 소스 코드를 완성하여 x를 y로 나누었을 때의 몫과 나머지가 출력되게 만드세요..

```
x = 20
y = 3


_____

_____
q, r = div(x, y)
Print('몫: {0}, 나머지: {1}, format(q, r))



실행 결과
몫: 6, 나머지: 2
```

COS-PRO 파이썬

12.연습문제

문혜영 교수

- 1. 김남규 씨의 과목별 점수는 다음과 같다. 김남규 씨의 평균 점수를 구해 보자

과목	점수
전산	90
영어	75
수학	55

- 2. 자연수 17이 홀수인지 짝수인지 판별할 수 있는 방법에 대해 말해 보자.

- 3. 김남규 씨의 주민등록번호는 991120-1068234 이다. 김남규 씨의 주민등록번호를 연월일 (YYYYMMDD) 부분과 그 뒤의 숫자 부분으로 나누어 출력해 보자.

```
pin = "991120-1068234"
yyyymmdd =
num =
print(                    )        # 연월일 부분 출력
print(                    )        # 숫자 부분 출력
```

- 4. 주민등록번호 뒷자리의 맨 첫 번째 숫자는 성별을 나타낸다. 주민등록번호에서 성별을 나타내는 숫자를 출력해 보자.

```
pin = "990706-1652242"
print(          )
```

- 5. 다음과 같은 문자열 a:b:c:d가 있다. 문자열의 replace 함수를 사용하여 a#b#c#d로 바꿔서 출력해 보자.

```
a = "a:b:c:d"
b =
print(b)                    # 문자열 "a#b#c#d" 출력
```

- 6. [1,3,9,4,2] 리스트를 [9,4,3,2,1]로 만들어 보자

```
a = [1, 3, 9, 4, 2]
a.                                  # [1,2,3,4,5]로 변경
a.                                  # [5,4,3,2,1]로 변경
print(                  )
```

- 7. 파이썬은 다음처럼 동일한 값에 여러 개의 변수를 선언할 수 있다. 다음과 같이 a, b 변수를 선언한 후 a의 두 번째 요소값을 변경하면 b 값은 어떻게 될까?

```
>>> a = b = [1, 2, 3]
>>> a[1] = 4
>>> print(b)
```

- 8. 다음 코드의 결과값은 무엇일까?

```
a = “Life is too short, you need python”

if ‘wife’ in a: print(“wife”)
elif ‘python’ in a and ‘you’ not in a: print(“python”)
elif “shirt” not in a: print(“shirt”)
elif ‘need’ in a: print(“need”)
else print(“none”)
```

- 9. while 문을 사용해 1부터 1000까지의 자연수 중 3의 배수의 합을 구해 보자.

```
result = 0
i = 1
while i <= 1000:
    if [          ]          # 3으로 나누어 떨어지는 수는 3의 배수
        result += i
    i += i

print(result)          #166833 출력
```

- 10. while문을 사용하여 다음과 같이 별(*)을 표시하는 프로그램을 작성해 보자.

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

```
i = 0
while True:
    i += 1
    if [          ] :break
    print ( [          ] )
```

```
# while문을 수행할 때 1씩 증가
# i 값이 5를 초과하면 while 문을 벗어난다.
# i값 개수만큼 '*'을 출력
```

- 11. for문을 사용해 1부터 100까지의 숫자를 출력해 보자.

```
>>> for i in [          ]
...         print(i)
...
1
2
3
4
5
6
7
8
9
10
. . . 생략 . . .
```

- 12. A 학급에 총 10명의 학생이 있다. 이 학생들의 중간고사 점수는 다음과 같다.

70, 60, 55, 75, 95, 90, 80, 80,85,100

for 문을 사용하여 A 학급의 평균 점수를 구해 보자.

```
A = [70, 60, 55, 75, 95, 90, 80, 80,85,100]
total = 0
for score in A:
    total += [          ]   #A 학급의 점수를 모두 더한다.
average = [          ]   # 평균을 구하기 위해 총 점수를 총 학생수로 나눈다.
print (average)
```

- 13. 주어진 자연수가 홀수인지 짝수인지 판별해 주는 함수(odd)를 작성해 보자.

```
def odd(n):
    if [          ]      # 2로 나누었을 때 나머지가 1이면 홀수이다.
        return True
    else:
        return False
```

COS-PRO 파이썬

13 딕셔너리, 집합

문혜영 교수

목차

- 딕셔너리
  - 리스트나 튜플처럼 순차적으로 해당 요소의 값을 구하지 않고 키를 이용하여 값을 얻는다.
    - dic={'name':'moon', 'phone':'000-0000', 'birth':'1009'}
    - dic['name']
  - 중복되는 키를 설정하면 마지막것만 저장된다.
    - a={1:'a',1:'b'}
    - a
    - {1:'b'}

- 집합
  - 중복을 허용하지 않는다.
  - 순서가 없다.
  - & 합집합
  - | 교집합
  - - 차집합
  - Add( ) 추가하기
  - Update( ) 여러 개의 값을 한꺼번에 추가하기
  - Remove( ) 특정값을 제거

COS-PRO 파이썬

14 클래스

문혜영 교수

목차

- 클래스
  - 객체를 표현하기 위한 문법이다.

```
class 클래스이름:
    def 메서드(self):
        코드

class Person:
    def greeting(self):
        print('안녕하세요')

j=person( )
j.greeting( )
```

결과
안녕하세요

```
class Person:
    def __init__(self):
        self.hello='HI'

    def greeting(self):
        print(self.hello)

j=Person()
j.greeting()
```

```
class 클래스이름:
    def __init__(self, 매개변수, 매개변수):
        self.속성1=매개변수
        self.속성2=매개변수
```

```
class Per:
    def __init__(self,name,age,address):
        self.hello='안녕'
        self.name=name
        self.age=age
        self.address=address

    def gre(self):
        print('{0} 저는 {1}입니다.'.format(self.hello, self.name))

m=Per('문혜영','99','서울시')
m.gre()

print(m.name)
print(m.age)
print(m.address)
```