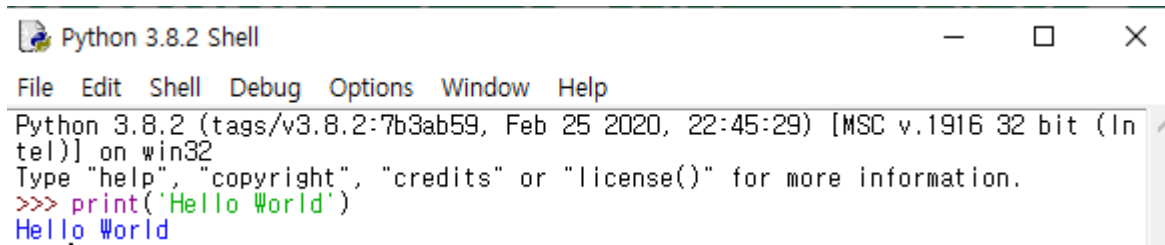


파이썬의 기초

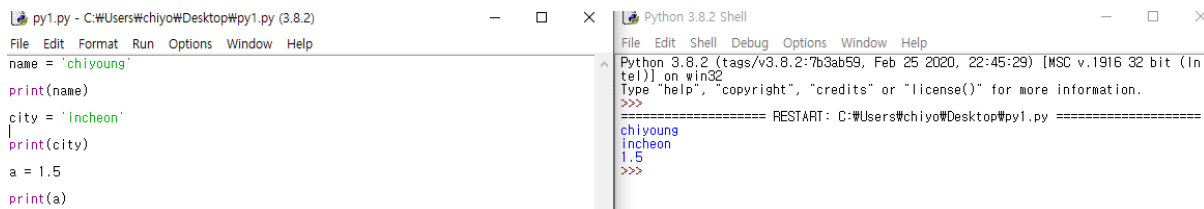
'Hello World'를 출력하기



```
Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print('Hello World')
Hello World
```

- 출력하고자 하는 문자열 앞뒤에 작은따옴표를 붙임
- 프린트 앞에 공백을 넣으면 무조건 오류가 발생함.
- 편집에서 파일 저장 후 오픈하여 저장 소스 실행 가능

파이썬 변수



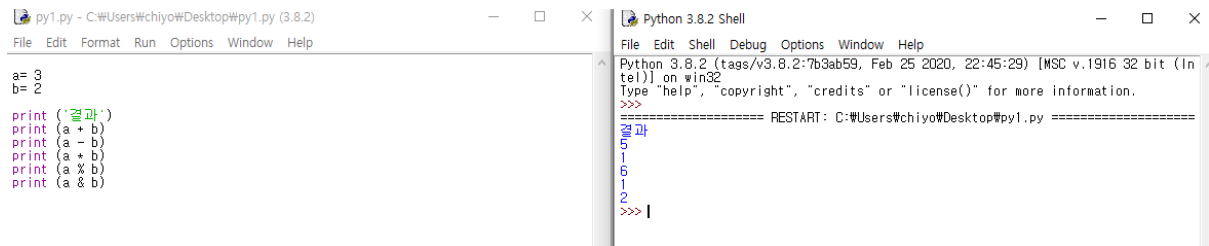
```
py1.py - C:\Users\chiyo\Desktop\py1.py (3.8.2)
File Edit Format Run Options Window Help
name = 'chiyoung'
print(name)
city = 'incheon'
print(city)
a = 1.5
print(a)

Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\chiyo\Desktop\py1.py =====
chiyoung
incheon
1.5
>>>
```

- 변수에는 정수 뿐만 아니라 실수나 문자도 넣을 수 있다.
- 대입 연산자 = > 값을 변수에 저장, = 의 의미는 왼쪽 화살표 라고 생각

파이썬의 기본 연산

변수를 만들어 기본 연산하기



```
py1.py - C:\Users\chiyo\Desktop\py1.py (3.8.2)
File Edit Format Run Options Window Help

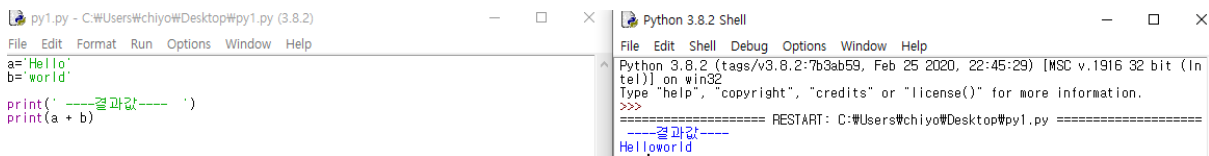
a= 3
b= 2

print ('결과')
print (a + b)
print (a - b)
print (a * b)
print (a % b)
print (a & b)
```

```
Python 3.8.2 Shell
File Edit Shell Debug Options Window Help

Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\chiyo\Desktop\py1.py =====
결과
5
1
6
1.5
>>> |
```

여러 문자열들 연결하기



```
py1.py - C:\Users\chiyo\Desktop\py1.py (3.8.2)
File Edit Format Run Options Window Help

a='Hello'
b='world'

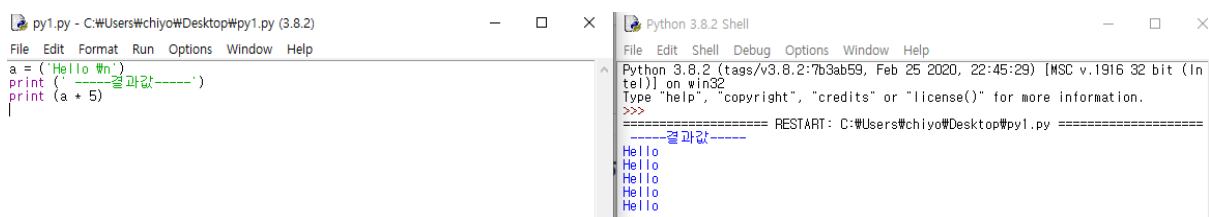
print ('-----결과값----- ')
print (a + b)
```

```
Python 3.8.2 Shell
File Edit Shell Debug Options Window Help

Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\chiyo\Desktop\py1.py =====
-----결과값-----
Hello world
>>> |
```

- 변수로 지정하여 + 를 이용해 연결하고 싶은 문자열 변수값을 더해준다.
- 변수가 지정 아닐 경우 ' ' 를 이용하여 + 로 연결해줌 * 숫자 숫자는 반복횟수

문자열 반복하기



```
py1.py - C:\Users\chiyo\Desktop\py1.py (3.8.2)
File Edit Format Run Options Window Help

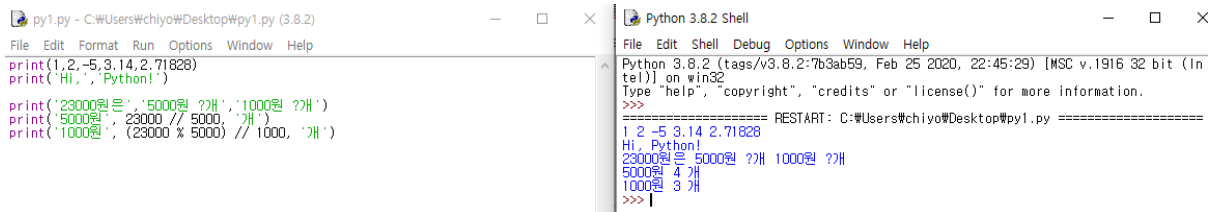
a = ('Hello\n')
print ('-----결과값-----')
print (a * 5)
```

```
Python 3.8.2 Shell
File Edit Shell Debug Options Window Help

Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\chiyo\Desktop\py1.py =====
-----결과값-----
Hello
Hello
Hello
Hello
Hello
>>> |
```

- \n은 줄을 바꾸라는 의미 이다.

연산자를 이용한 지폐 계산 실습 예제



```
py1.py - C:\Users\chiyo\Desktop\py1.py (3.8.2)
File Edit Format Run Options Window Help
print(1,2,-5,3,14,2,71828)
print('Hi','Python!')

print('23000원은', 5000원 ??개, '1000원 ??개')
print('5000원', 23000 // 5000, '개')
print('1000원', (23000 % 5000) // 1000, '개')

Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\chiyo\Desktop\py1.py =====
1 2 -5 3.14 2.71828
Hi, Python!
23000원은 5000원 ??개 1000원 ??개
5000원 4 개
1000원 3 개
>>>
```

- 콤마로 구분하여 연산하여 위치럼 응용이 가능하다

문자열 함수

len(문자열)	문자열의 길이 반환
count(찾을 글자)	문자열안에 찾을 글자 개수 반환
index(찾을 글자)	찾을 글자의 위치 반환
upper()	문자열 대문자로 바꾸기
lower()	문자열 소문자로 바꾸기
strip()	양쪽 공백 지우기
lstrip()	왼쪽 공백 지우기
rstrip()	오른쪽 공백 지우기
replace(바꿀 글자)	문자열 개수 세기
split(구분자)	구분자를 기준으로 나눈다, 없으면 공백으로 나눠서 리스트로 반환

리스트

- 리스트는 여러 개의 자료형을 담을수 있는 배열

리스트의 구조

- 리스트명 = [요소1,요소2,요소3,...]

기본문법

```
test.py - C:/Users/chiyo/Desktop/test.py (3.8.2)
File Edit Format Run Options Window Help
sampleList= [1,2,3, 'python', 'java', 'C']
print('-----결과값-----')
print(sampleList)

Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/chiyo/Desktop/test.py =====
-----결과값-----
[1, 2, 3, 'python', 'java', 'C']
>>>

gg.py - C:/Users/chiyo/Desktop/gg.py (3.8.2)
File Edit Format Run Options Window Help
sampleList = [1,2,3,[4,5,6]]
print('-----결과값-----')
print(sampleList[0])
print(sampleList[1])
print(sampleList[2])
print(sampleList[3])

Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/chiyo/Desktop/gg.py =====
-----결과값-----
1
2
3
```

- 위와 같이 리스트 안에 리스트가 들어 갈 수도 있다.

리스트 함수

append(추가할 값)

- 리스트에 맨 뒷부분에 추가할 값을 추가

sort()

- 리스트 정렬

index(찾을 값)

- 리스트에서 찾을 값의 위치를 반환

insert(넣을 위치, 넣을 값)

- 리스트에 넣을 위치에 넣을 값을 넣어준다.

remove(삭제할 값)

- 리스트에서 삭제할 값을 찾아서 삭제 (가장 먼저 나온 값)

count(찾을 값)

- 리스트에 찾을 값이 몇 개 있는지 개수를 세서 반환

extend(확장할 리스트)

- 리스트에 확장할 리스트를 붙임

리스트 함수 실습해보기

```
gg.py - C:/Users/chiyo/Desktop/gg.py (3.8.2)
File Edit Format Run Options Window Help
list = [12, 5, 3, 12]
print (list)
list.append(27)
print (list)
list.sort()
print (list)
list.sort(reverse=True)
print (list)
print (list.index(12))
list.insert(0, 3)
print (list)
list.remove(3)
print (list)
print (list.count(12))
list.extend(['p', 's', 'y'])
print (list)

Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29
tel)) on win32
type "help", "copyright", "credits" or "license()" for m
>>>
===== RESTART: C:/Users/chiyo/Desktop/gg
[12, 5, 3, 12]
[12, 5, 3, 12, 27]
[3, 5, 12, 12, 27]
[27, 12, 12, 5, 3]
[3, 27, 12, 12, 5, 3]
[27, 12, 12, 5, 3]
2
[27, 12, 12, 5, 3, 'p', 's', 'y']
>>>
```

튜플

튜플은 몇 가지 점을 제외하곤 리스트와 비슷하고 리스트와 다른점은

	리스트	튜플
괄호	[and]	(and)
수정	생성,삭제,수정 가능	수정 불가

- 값이 변경이 되면 안 될 때는 튜플을 쓰고, 변화가 필요하면 리스트를 사용하자

딕셔너리

{key1:value1, key2:value2, key3:value3 ...}

Key와 values의 쌍 여러 개가 {}로 둘러 싸여있으며 각각의 요소는 key : value 형태로 이루어져 있고 쉼표로 구분되어있음.

- Key는 변하지 않는 값을 사용함, value에는 변하는 값과 변하지 않는 값 모두 사용 가능

딕셔너리 기능이해 하기

```
gg.py - C:/Users/chiyo/Desktop/gg.py (3.8.2)
File Edit Format Run Options Window Help
parkchiyoung = {'name': 'chiyoung', 'phone': '01011111111', 'birth': '1209'}

print ('\n----- Result -----')
print (parkchiyoung)
print (parkchiyoung['name'])
print (parkchiyoung['phone'])
print (parkchiyoung['birth'])

print ('\n----- Result2 -----')
# 추가
print (parkchiyoung)
# 같은 key에 추가
print (parkchiyoung)
# 삭제
print (parkchiyoung)
# Key만 출력
print (parkchiyoung.keys())
# 값만 출력
print (parkchiyoung.values())

Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/chiyo/Desktop/gg.py =====
---- Result ----
{'name': 'chiyoung', 'phone': '01011111111', 'birth': '1209'}
chiyoung
01011111111
1209
---- Result2 ----
{'name': 'chiyoung', 'phone': '01011111111', 'birth': '1209'}
{'name': 'chiyoung', 'phone': '01011111111', 'birth': '1209'}
{'name': 'chiyoung', 'phone': '01011111111', 'birth': '1209'}
dict_keys(['name', 'phone', 'birth'])
dict_values(['chiyoung', '01011111111', '1209'])
>>>
```

딕셔너리를 이용한 월 랜덤 출력 실습해보기

```
gg.py - C:/Users/chiyo/Desktop/gg.py (3.8.2)
File Edit Format Run Options Window Help
month = {1: 'January', 2: 'February', 3: 'March', 4: 'April'}
month[5] = 'May'
month[6] = 'June'
month[7] = 'July'
month[8] = 'August'
month[9] = 'September'
print(month)
print()

from random import randint
#임의로 5번의 월 단어 출력

for i in range(5):
    r = randint(1, 9)
    print('%d: %s' % (r, month[r]))

Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/chiyo/Desktop/gg.py =====
{1: 'January', 2: 'February', 3: 'March', 4: 'April', 5: 'May', 6: 'June', 7: 'July', 8: 'August', 9: 'September'}
8: August
6: June
9: September
3: March
5: May
>>>
```

If문

if문은 조건을 주고 참과 거짓을 나눌 때 사용한다.

- 다른 언어에서는 (' { ' , ' } '로 하지만 파이썬에선 탭을 이용.

기본 예

If 조건 :

조건이 만족할 때 들어오는 명령어

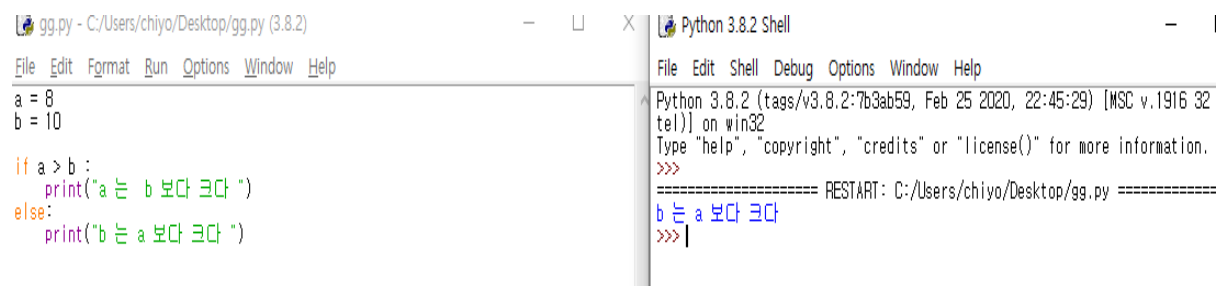
elif 조건2:

조건2가 만족할 때 들어오는 명령어

Else

조건이 만족하지 않을 때 들어오는 명령어

실습 코드



```
gg.py - C:/Users/chiyo/Desktop/gg.py (3.8.2)
File Edit Format Run Options Window Help
a = 8
b = 10
if a > b :
    print("a는 b보다 크다 ")
else:
    print("b는 a보다 크다 ")

Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32
tel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/chiyo/Desktop/gg.py =====
b는 a보다 크다
>>> |
```

- if문을 쓸 때 : 와 들여쓰기를 주의하자.

비교 연산자

비교 연산자

설명

$a > b$

a가 b보다 크다

$a < b$

a가 b보다 작다

$a \leq b$

a가 b보다 크거나 같다

<code>a <= b</code>	a가 b보다 작거나 같다
<code>a == b</code>	a와 b는 같다
<code>a != b</code>	a와 b는 같지 않다

논리 연산자

논리 연산자	설명
조건1 and 조건2	조건1과 조건2가 모두 만족해야 True
조건1 or 조건2	조건1이나 조건2 둘중 하나만 만족해도 True
not 조건	조건이 만족하지 않아야 True

논리 연산자 실습 코딩

```

gg.py - C:/Users/chiyo/Desktop/gg.py (3.8.2)
File Edit Format Run Options Window Help
money = 5000
card = 1

if money >= 6000 or card:
    print("택시를 이용 할 수 있다")
else:
    print("대중교통을 이용해야 한다.")

Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/chiyo/Desktop/gg.py =====
택시를 이용 할 수 있다
>>>
  
```

While 문

While 문은 조건문이 true인 동안은 계속 반복을 해주는 구문이다.

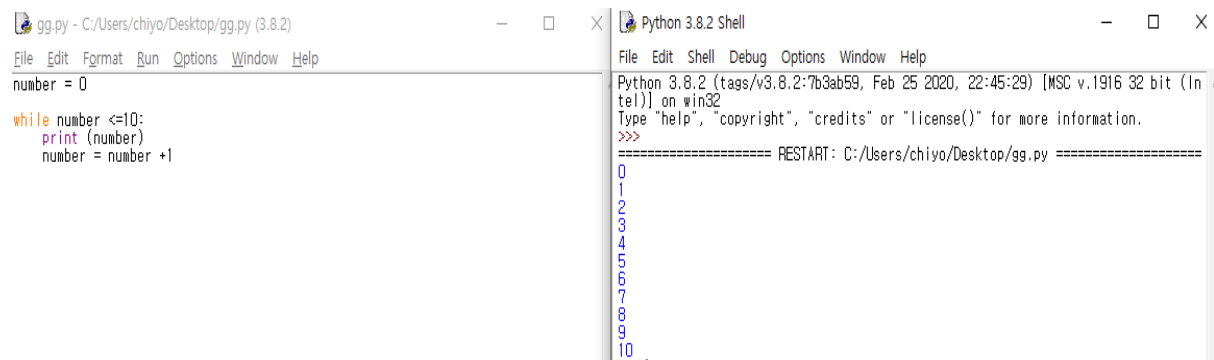
While문 기본 구조

While 조건문:

수행할 명령어

...

예)



```
gg.py - C:/Users/chiyo/Desktop/gg.py (3.8.2)
File Edit Format Run Options Window Help
number = 0
while number <=10:
    print (number)
    number = number +1

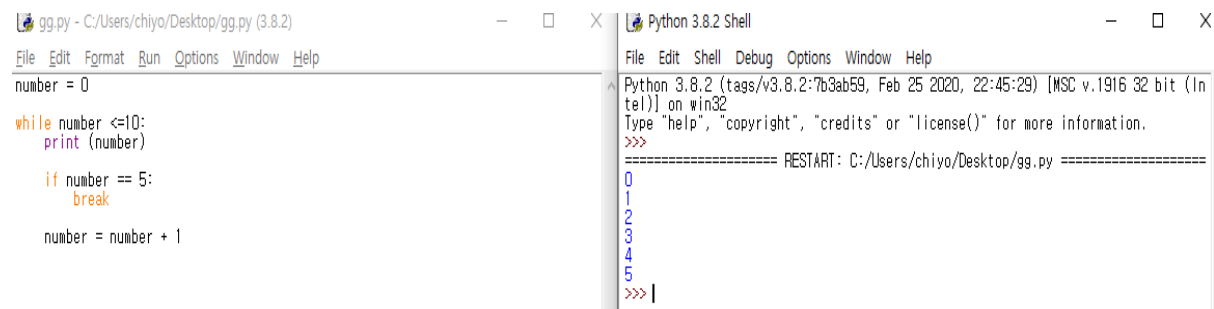
Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/chiyo/Desktop/gg.py =====
0
1
2
3
4
5
6
7
8
9
10
...
>>>
```

- While문을 사용하여 간단하게 조건이 참 이라면 계속 반복 시킬 수 있음.

While문 연속으로 증감하는법

- 익숙한 자바 에서는 a++ 이지만 파이썬에서는 a+=1, a-=1

While문 강제로 빠져나가기 (=break)



```
gg.py - C:/Users/chiyo/Desktop/gg.py (3.8.2)
File Edit Format Run Options Window Help
number = 0
while number <=10:
    print (number)
    if number == 5:
        break
    number = number + 1

Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/chiyo/Desktop/gg.py =====
0
1
2
3
4
5
>>>
```

- 아까와 비슷한 코드지만 break를 이용하여 원하는 조건에서 While문을 강제로 중단시킬 수 있다.

현재 돌고있는 순번만 건너 뛰기(=continue)

- Break와 달리 반복문 자체를 나가는 개념이 아니라 해당 싸이클만 건너 뛰게됨.

실습 코딩해보기

```
number = 0
while number < 10:
    number = number + 1
    if number % 2 == 0:
        continue
    print(number)
```

Python 3.8.2 (tags/v3.8.2:7
tel)] on win32
Type "help", "copyright", "
>>>
===== RESTA
1
3
5
7
9
>>> |

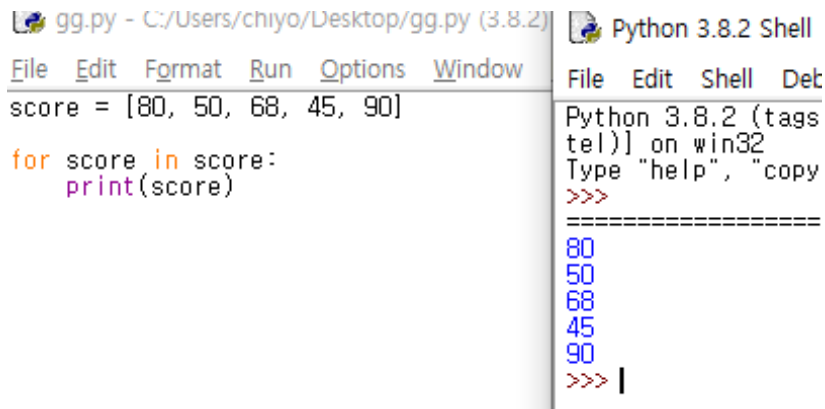
For문

For문 기본 구조

For 변수 in 리스트(튜플, 문자열):

수행할 명령어 ...

For문 기본 개념 코드



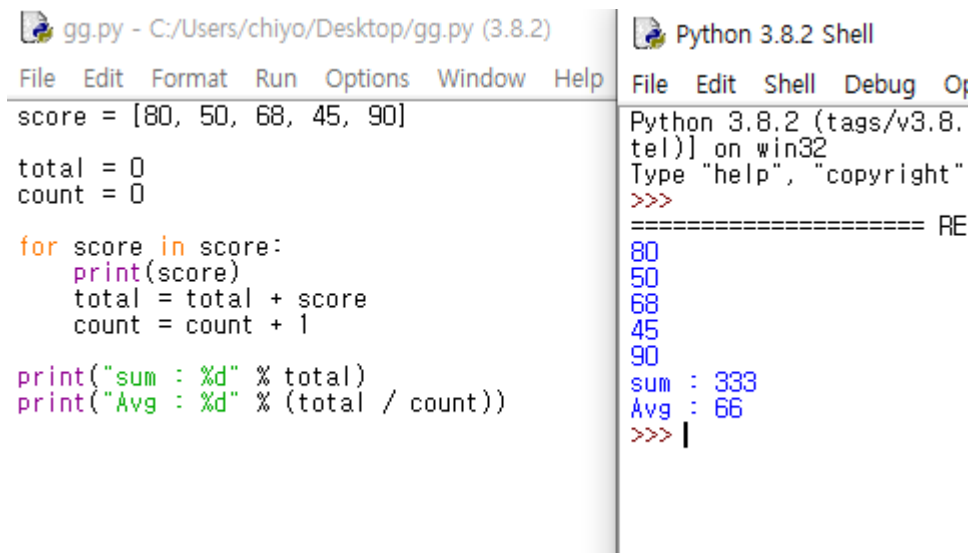
The image shows a Python IDE window titled 'gg.py - C:/Users/chiyo/Desktop/gg.py (3.8.2)' and a 'Python 3.8.2 Shell' window. The IDE window contains the following code:

```
score = [80, 50, 68, 45, 90]
for score in score:
    print(score)
```

The shell window shows the output of the code:

```
Python 3.8.2 (tags/
tel)] on win32
Type "help", "copy
>>>
=====
80
50
68
45
90
>>> |
```

For문 응용 실습코드



The image shows a Python IDE window titled 'gg.py - C:/Users/chiyo/Desktop/gg.py (3.8.2)' and a 'Python 3.8.2 Shell' window. The IDE window contains the following code:

```
score = [80, 50, 68, 45, 90]
total = 0
count = 0
for score in score:
    print(score)
    total = total + score
    count = count + 1
print("sum : %d" % total)
print("Avg : %d" % (total / count))
```

The shell window shows the output of the code:

```
Python 3.8.2 (tags/v3.8.
tel)] on win32
Type "help", "copyright
>>>
===== RE
80
50
68
45
90
sum : 333
Avg : 66
>>> |
```

range(start, end, step(default=1))함수

for문 에서 range 사용

```

File Edit Shell Debug Output
for a in range(10):
    print(a)
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020) on win32
Type "help", "copyright", "credits" or "license()"
>>>
===== RESTART: C:\Users\chiyo\
0
1
2
3
4
5
6
7
8
9
>>> |

```

```

for a in range(1,10):
    print(a)
1
2
3
4
5
6
7
8
9
>>> |

```

내장함수

내장함수 enumerate()

내장함수 zip()

시퀀스 간의 변환

튜플과 시퀀스 간의 변환

```

space = '밤', '낮', '해', '달'
print(space)
print(list(space))
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020) on win32
Type "help", "copyright", "credits" or "license()"
>>>
===== RESTART: C:\Users\chiyo\
('밤', '낮', '해', '달')
>>> |

```

리스트와 집합 간의 변환

```

File Edit Format Run Options Window Help
singer = ['BTS', '볼빨간사춘기', 'BTS', '블랙핑크']
print(singer)
print(set(singer))
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020) on win32
Type "help", "copyright", "credits" or "license()"
>>>
===== RESTART: C:\Users\chiyo\
['BTS', '볼빨간사춘기', 'BTS', '블랙핑크']
{'블랙핑크', '볼빨간사춘기', 'BTS'}
>>> |

```

파이썬 함수 정의 구문 : 함수 이름과 인자, 반환 값

함수 머리는 키워드 `def`

- 함수 이름과 괄호, 괄호 사이의 인자들
- 함수 정의의 몸체인 블록을 예고하는 콜론(:) 이 필요

함수 몸체는 들여쓰기로 시작

- 기능을 수행하는 문장들
- 마지막으로 키워드 `return`에 의한 반환 값을 돌려 주는 구문이 필요

함수로 필요한 값을 전달하는 인자

- 함수 정의에서 인자 사용

```
def hello(name):  
    print('안녕, {}'.format(name))  
  
hello('수빈')  
sname='현주'  
hello(sname)
```

Python 3.8.2 (tags
tel)] on win32
Type "help", "copy
>>>
=====

```
안녕, 수빈!  
안녕, 현주!  
>>> |
```

- 인자의 기본값 활용

```
def hello(name='여러분'):  
    print('안녕, {}'.format(name))  
  
hello()  
hello('현철')
```

Python 3.8.2 (tags
tel)] on win32
Type "help", "copyright", "crec
>>>
===== RESTART:
안녕, 여러분!
안녕, 현철!
>>> |

지역 변수와 전역 변수

지역과 전역변수

- 파이썬에서는 변수가 메모리에 생성되고 유지되는 범위

지역 변수

- 함수 내부에서 대입돼 생성된 변수
- 지역 변수는 함수 외부에서 절대 사용 X

전역 변수

- 함수 외부에서 대입돼 생성된 변수