

File Handling and Exception Handling

- File Handling
- Exception Handling

- 파일 처리 (File Handling)

- text 파일
 - 1) txt 파일 생성하기
 - 2) txt 파일 읽기
- csv / excel 파일
 - 1) csv, xlsx 파일 생성하기
 - 2) xlsx 파일 읽기
 - 3) csv 파일 읽기

- 예외 처리
(Exception Handling)

- Exercise 1
- Exercise 2

- open 함수

- 파일을 생성하거나 읽을 때 사용하는 함수
- 함수 사용법 : open("file_name", "처리 모드")

기호	처리 모드
r	읽기
w	쓰기
a	추가해서 쓰기
+	읽기, 쓰기

- 파일 처리 (File Handling)

- text 파일

- 1) txt 파일 생성하기

- 2) txt 파일 읽기

- csv / excel 파일

- 1) csv, xlsx 파일 생성하기

- 2) xlsx 파일 읽기

- 3) csv 파일 읽기

- 예외 처리

- (Exception Handling)

- Exercise 1

- Exercise 2

■ text 파일

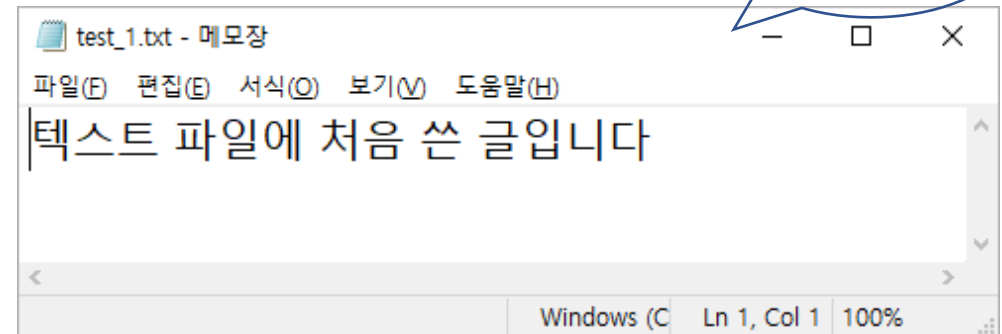
1) txt 파일에 내용 쓰기

```
1 import os
2 os.makedirs("D:/temp33")
3 os.chdir("D:/temp33")
4 print( os.getcwd( ) )
5
6 file1 = open("test_1.txt", "w")
7 file1.write("텍스트 파일에 처음 쓴 글입니다")
8 file1.close( )
```

D:\temp33



D:/temp33



- 파일 처리 (File Handling)

- text 파일

- 1) txt 파일 생성하기

- 2) txt 파일 읽기

- csv / excel 파일

- 1) csv, xlsx 파일 생성하기

- 2) xlsx 파일 읽기

- 3) csv 파일 읽기

- 예외 처리

- (Exception Handling)

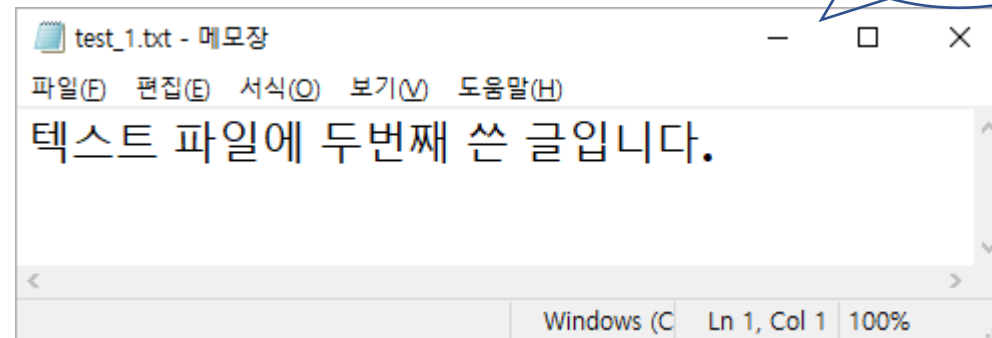
- Exercise 1

- Exercise 2

```
1 file2 = open("test_1.txt", "w")
2 file2.write("텍스트 파일에 두번째 쓴 글입니다.")
3 file2.close()
```



D:/temp33



- 파일 처리 (File Handling)

- text 파일

- 1) txt 파일 생성하기

- 2) txt 파일 읽기

- csv / excel 파일

- 1) csv, xlsx 파일 생성하기

- 2) xlsx 파일 읽기

- 3) csv 파일 읽기

- 예외 처리

- (Exception Handling)

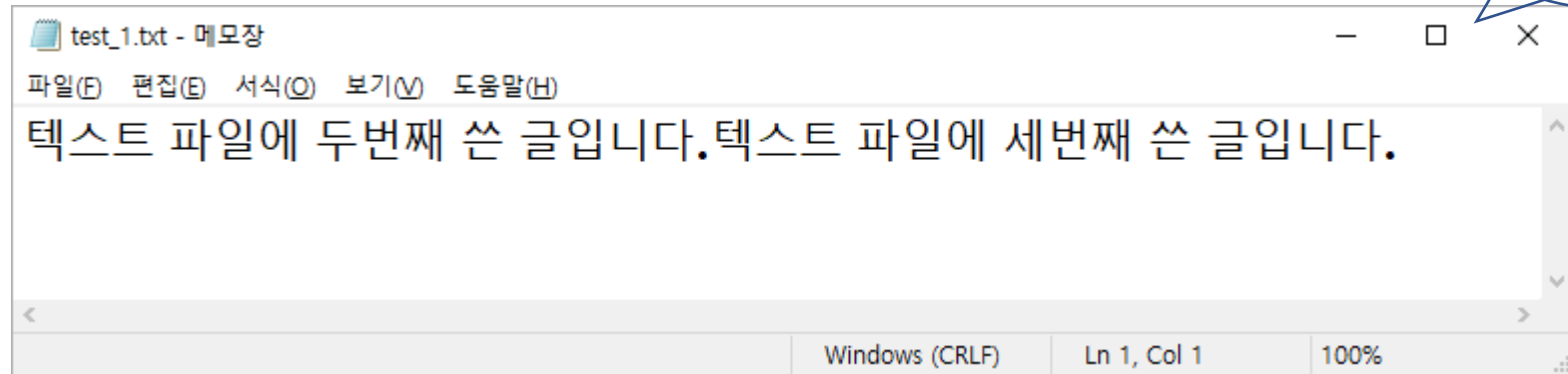
- Exercise 1

- Exercise 2

```
1 file3 = open("test_1.txt", "a")
2 file3.write("텍스트 파일에 세번째 쓴 글입니다.")
3 file3.close( )
```



D:/temp33



- 파일 처리 (File Handling)

- text 파일

- 1) txt 파일 생성하기

- 2) txt 파일 읽기

- csv / excel 파일

- 1) csv, xlsx 파일 생성하기

- 2) xlsx 파일 읽기

- 3) csv 파일 읽기

- 예외 처리

- (Exception Handling)

- Exercise 1

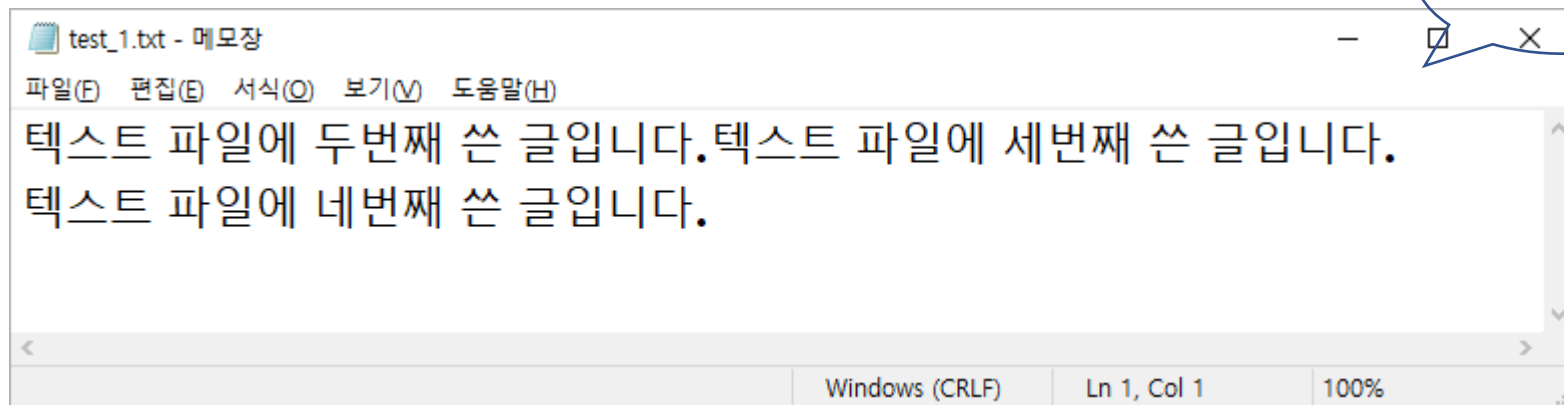
- Exercise 2

줄바꿈 되는 엔터키 사용하기

```
1 file4 = open("test_1.txt", "a")
2 file4.write("ㄱ\n" + "텍스트 파일에 네번째 쓴 글입니다.")
3 file4.close()
```



D:/temp33



- 파일 처리 (File Handling)

- text 파일

- 1) txt 파일 생성하기

- 2) txt 파일 읽기

- csv / excel 파일

- 1) csv, xlsx 파일 생성하기

- 2) xlsx 파일 읽기

- 3) csv 파일 읽기

- 예외 처리

- (Exception Handling)

- Exercise 1

- Exercise 2

D:\temp33 디렉토리에 readme.txt 만들기

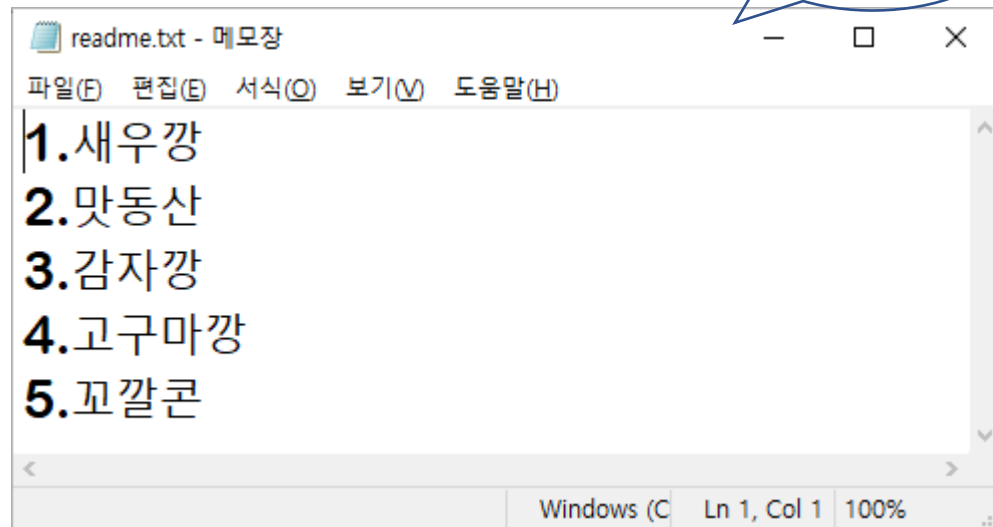
```
1 os.getcwd()
```

'D:\\\\temp33'

```
1 file1 = open("readme.txt", "w")
2 file1.write("1.새우깡\\n2.맛동산\\n3.감자깡\\n4.고구마깡\\n5.꼬깔콘")
3 file1.close()
```



D:/temp33



- 파일 처리 (File Handling)
- text 파일
 - 1) txt 파일 생성하기
 - 2) txt 파일 읽기
- csv / excel 파일
 - 1) csv, xlsx 파일 생성하기
 - 2) xlsx 파일 읽기
 - 3) csv 파일 읽기
- 예외 처리 (Exception Handling)
- Exercise 1
- Exercise 2

2) txt 파일의 내용 읽기

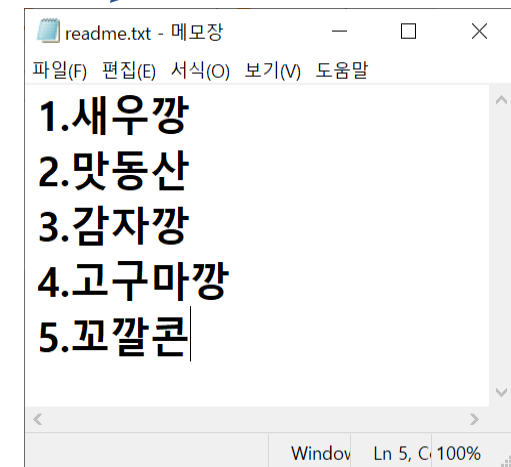
```
1 f = open("readme.txt", "r")
2 f.readlines()
```

```
['1.새우깡\n', '2.맛동산\n', '3.감자깡\n', '4.고구마깡\n', '5.꼬깔콘']
```

```
1 f = open("readme.txt", "r")
2 snack = f.readlines()
3 print(snack)
```

```
['1.새우깡\n', '2.맛동산\n', '3.감자깡\n', '4.고구마깡\n', '5.꼬깔콘']
```

D:/temp33



- 파일 처리 (File Handling)

- text 파일

- 1) txt 파일 생성하기

- 2) txt 파일 읽기

- csv / excel 파일

- 1) csv, xlsx 파일 생성하기

- 2) xlsx 파일 읽기

- 3) csv 파일 읽기

- 예외 처리

- (Exception Handling)

- Exercise 1

- Exercise 2

```
snack ['1.새우깡\n', '2.맛동산\n', '3.감자깡\n', '4.고구마깡\n', '5.꼬깔콘']
```

```
1 print(snack[0])
2 print(snack[1])
3 print(snack[3])
4 print(snack[4])
```

1.새우깡

2.맛동산

4.고구마깡

5.꼬깔콘

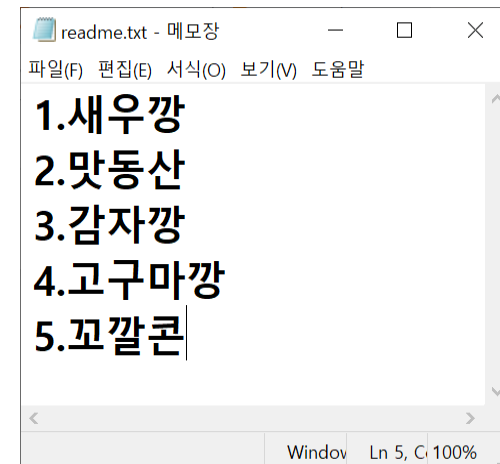
```
1 print(snack[0]+snack[1] )
2 print(snack[2]+snack[3] )
```

1.새우깡

2.맛동산

3.감자깡

4.고구마깡



- 파일 처리 (File Handling)

- text 파일

- 1) txt 파일 생성하기

- 2) txt 파일 읽기

- csv / excel 파일

- 1) csv, xlsx 파일 생성하기

- 2) xlsx 파일 읽기

- 3) csv 파일 읽기

- 예외 처리

- (Exception Handling)

- Exercise 1

- Exercise 2

```

1 print("="*30, "\n", " for statement (1)")
2 print("="*30)
3
4 for i in snack :
5     print(i) # 엔터키까지 읽음
6     print('-'*20)

```

```

=====
for statement (1)
=====

```

1. 새우깡

2. 맛동산

3. 감자깡

4. 고구마깡

5. 꼬깔콘

snack ['1.새우깡\n', '2.맛동산\n', '3.감자깡\n', '4.고구마깡\n', '5.꼬깔콘']

- 파일 처리 (File Handling)

- text 파일

- 1) txt 파일 생성하기

- 2) txt 파일 읽기

- csv / excel 파일

- 1) csv, xlsx 파일 생성하기

- 2) xlsx 파일 읽기

- 3) csv 파일 읽기

- 예외 처리

- (Exception Handling)

- Exercise 1

- Exercise 2

```

1 print("="*30, "ㄴ", " for statement (2)")
2 print("="*30)
3
4 for i in snack :
5     print(i.rstrip()) # 엔터키까지 읽은 후 오른쪽 공백 제거
6     print('-'*20)

```

```

=====
    for statement (2)
=====
1.새우깡
-----
2.맛동산
-----
3.감자깡
-----
4.고구마깡
-----
5.꼬깔콘
-----

```

snack ['1.새우깡ㄴ', '2.맛동산ㄴ', '3.감자깡ㄴ', '4.고구마깡ㄴ', '5.꼬깔콘']

- 파일 처리 (File Handling)
- text 파일
 - 1) txt 파일 생성하기
 - 2) txt 파일 읽기
- **csv / excel 파일**
 - 1) csv, xlsx 파일 생성하기
 - 2) xlsx 파일 읽기
 - 3) csv 파일 읽기
- 예외 처리
(Exception Handling)
- Exercise 1
- Exercise 2

■ csv / excel 파일

- csv : comma-separated values
- excel 형식과 csv 형식으로 저장하기 위해서는 table형태(데이터 프레임)로 데이터를 구성해야 하며, numpy, pandas 등의 패키지 필요

- 파일 처리 (File Handling)
- text 파일
 - 1) txt 파일 생성하기
 - 2) txt 파일 읽기
- **csv / excel 파일**
 - 1) csv, xlsx 파일 생성하기
 - 2) xlsx 파일 읽기
 - 3) csv 파일 읽기
- 예외 처리 (Exception Handling)
- Exercise 1
- Exercise 2

■ numpy 패키지 확인 및 설치

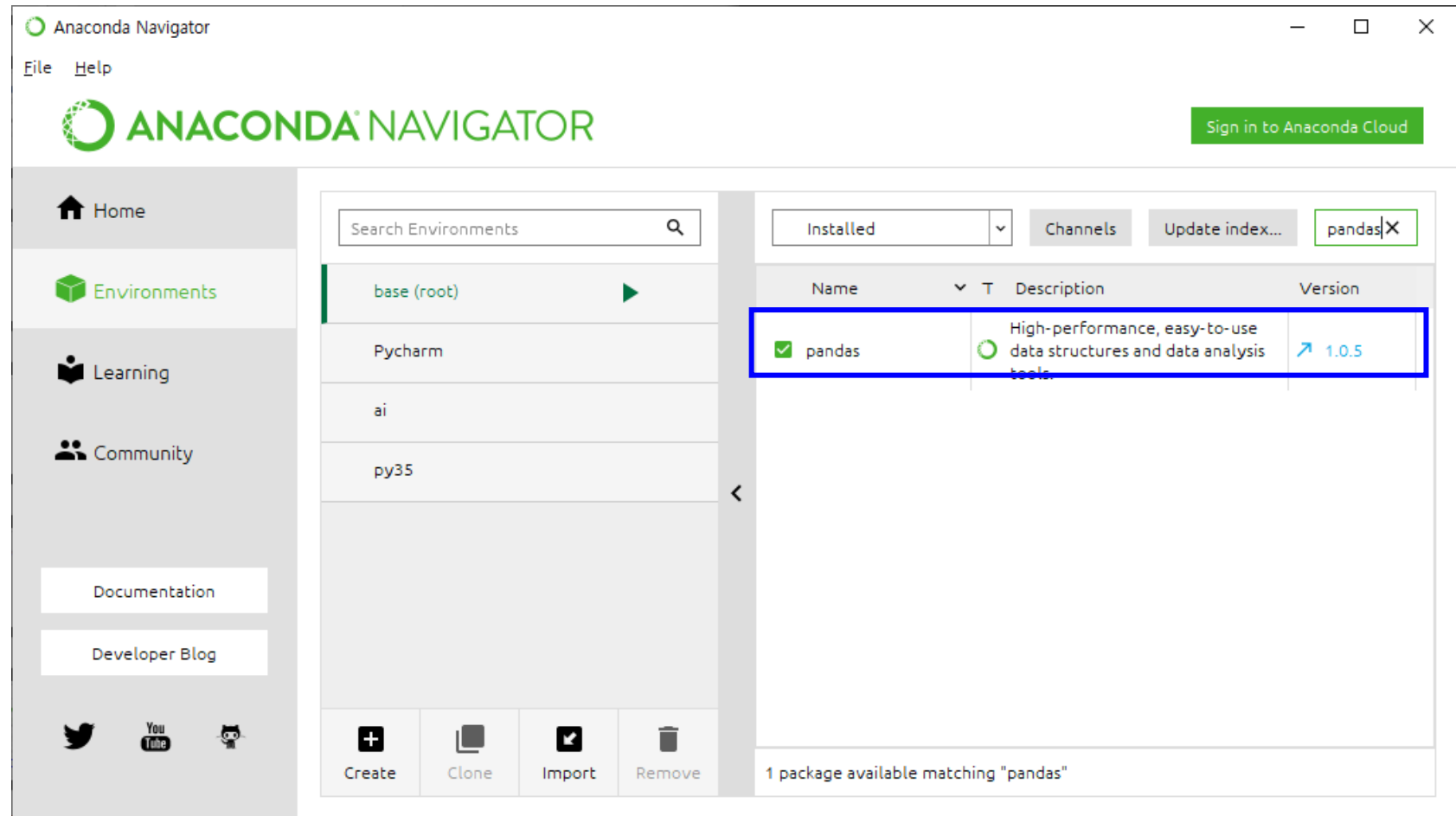
The screenshot shows the Anaconda Navigator interface. On the left sidebar, the 'Environments' tab is selected. The main panel displays a list of installed packages for the 'base (root)' environment. The 'numpy' package is highlighted with a blue box, showing its version as 1.18.5. Below the list, it indicates that 9 packages are available matching 'numpy'.

Name	Description	Version
bottleneck	Fast numpy array functions written in cython.	1.3.2
mkl_fft	Numpy-based implementation of fast fourier transform using intel (r) math kernel library.	1.1.0
mkl_random	Intel (r) mkl-powered package f...	1.1.1
numba	Numpy aware dynamic python c...	0.50.1
numexpr	Fast numerical expression evaluator for numpy.	2.7.1
numpy	Array processing for numbers, strings, records, and objects.	1.18.5
numpy-base		1.18.5

9 packages available matching "numpy"

- 파일 처리 (File Handling)
- text 파일
 - 1) txt 파일 생성하기
 - 2) txt 파일 읽기
- **csv / excel 파일**
 - 1) csv, xlsx 파일 생성하기
 - 2) xlsx 파일 읽기
 - 3) csv 파일 읽기
- 예외 처리 (Exception Handling)
- Exercise 1
- Exercise 2

■ pandas 패키지 확인 및 설치



- 파일 처리 (File Handling)
- text 파일
 - 1) txt 파일 생성하기
 - 2) txt 파일 읽기
- **csv / excel 파일**
 - 1) csv, xlsx 파일 생성하기
 - 2) xlsx 파일 읽기
 - 3) csv 파일 읽기
- 예외 처리
(Exception Handling)
- Exercise 1
- Exercise 2

- pip를 이용하여 설치할 수 있음
- 아나콘다의 경우 conda를 이용하여 설치할 수도 있음

원하는 버전이
있을 경우

Install of numpy

```
(base) C:\Users\tina>conda install numpy==1.18.5
.
.
Proceed ([y]/n)? y
```

Install of pandas

```
(base) C:\Users\tina>conda install pandas
.
.
Proceed ([y]/n)? y
```

- 파일 처리 (File Handling)

- text 파일
 - 1) txt 파일 생성하기
 - 2) txt 파일 읽기

- csv / excel 파일
 - 1) csv, xlsx 파일 생성하기
 - 2) xlsx 파일 읽기
 - 3) csv 파일 읽기

- 예외 처리
(Exception Handling)

- Exercise 1
- Exercise 2

- csv / excel 파일

```
1 import pandas as pd
```

표 (데이터 프레임) 만들기

```
1 no = [ ]
2 no.append(100)
3 no.append(200)
4 no.append(300)
5 print(no)
```

[100, 200, 300]

```
1 subject_name = [ ]
2 subject_name.append('수학')
3 subject_name.append('과학')
4 subject_name.append('빅데이터')
5 print(subject_name)
```

['수학', '과학', '빅데이터']

```
1 table = pd.DataFrame( )
2 table['과목코드'] = no
3 table['과목명'] = subject_name
4 print(table)
```

	과목코드	과목명
0	100	수학
1	200	과학
2	300	빅데이터

- 파일 처리 (File Handling)

- text 파일
 - 1) txt 파일 생성하기
 - 2) txt 파일 읽기

- csv / excel 파일
 - 1) csv, xlsx 파일 생성하기
 - 2) xlsx 파일 읽기
 - 3) csv 파일 읽기

- 예외 처리
(Exception Handling)

- Exercise 1
- Exercise 2

- csv 형식으로 저장하기

데이터를 저장할 디렉토리 만들기

D:/ai/DATA

```
1 os.makedirs("D:/ai/DATA") # 데이터를 저장할 디렉토리
2 print( os.path.exists("D:/ai/DATA") )
```

True

```
1 os.getcwd()
```

'D:\\\\temp33'

```
1 os.chdir('D:/ai')
2 os.getcwd()
```

'D:\\\\ai '

- 파일 처리 (File Handling)

- text 파일
 - 1) txt 파일 생성하기
 - 2) txt 파일 읽기

- csv / excel 파일
 - 1) csv, xlsx 파일 생성하기
 - 2) xlsx 파일 읽기
 - 3) csv 파일 읽기

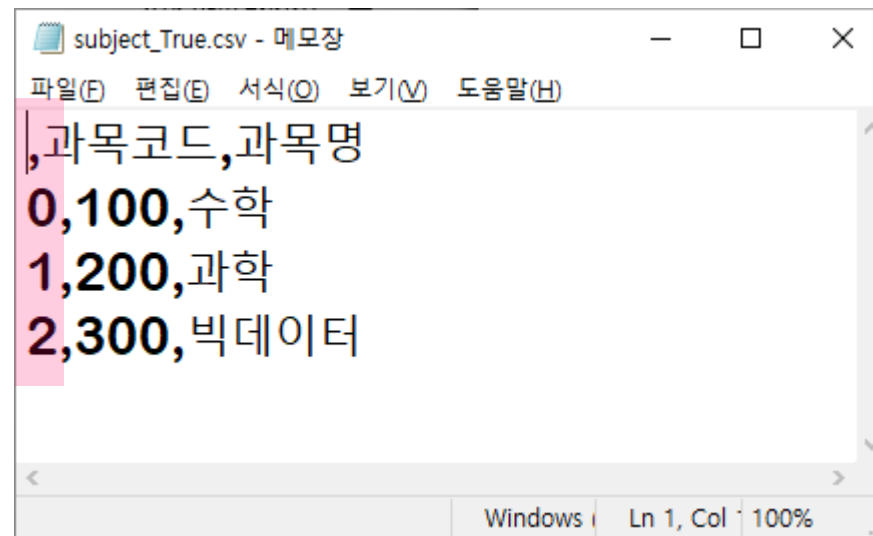
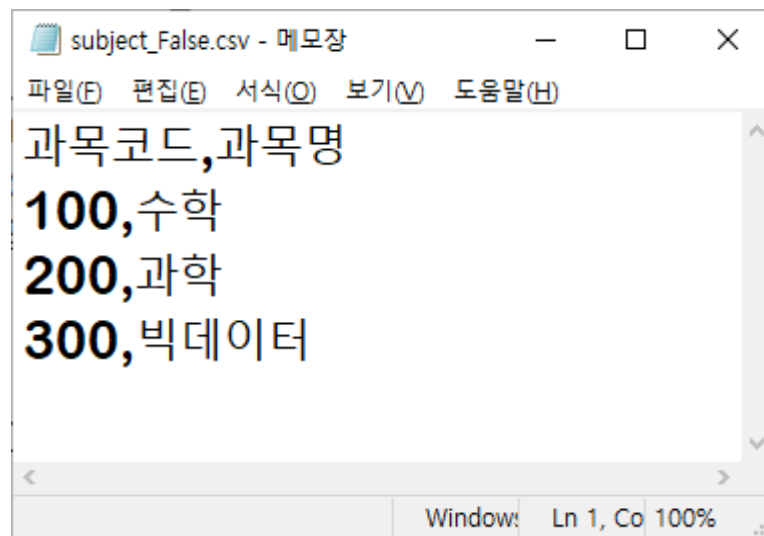
- 예외 처리
(Exception Handling)

- Exercise 1
- Exercise 2

- csv 형식으로 저장하기

```
1 table.to_csv("DATA/subject_False.csv", encoding="utf-8-sig", index=False)
```

```
1 table.to_csv("DATA/subject_True.csv", encoding="utf-8-sig", index=True)
```



- 파일 처리 (File Handling)

- text 파일
 - 1) txt 파일 생성하기
 - 2) txt 파일 읽기

- csv / excel 파일
 - 1) csv, xlsx 파일 생성하기
 - 2) xlsx 파일 읽기
 - 3) csv 파일 읽기

- 예외 처리
(Exception Handling)

- Exercise 1
- Exercise 2

- excel 형식으로 저장하기

```
1 table.to_excel("DATA/subject_False.xlsx", sheet_name="Sheet1", index=False)
```

```
1 table.to_excel("DATA/subject_True.xlsx", sheet_name="Sheet1", index=True)
```

	A	B	C	D
1	과목코드	과목명		
2	100	수학		
3	200	과학		
4	300	빅데이터		
5				

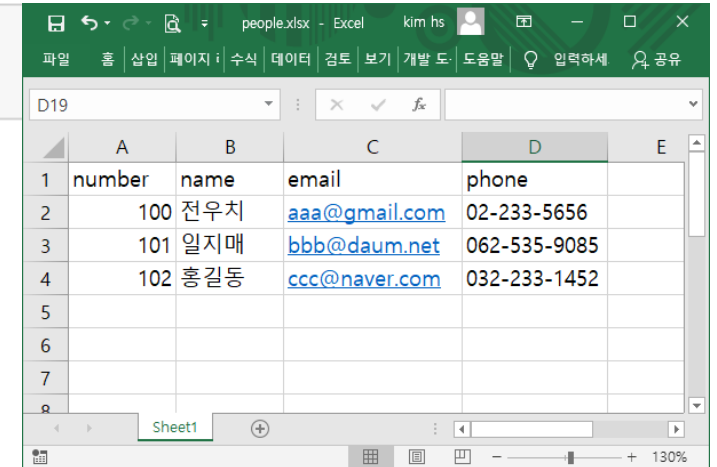
	A	B	C	D
1		과목코드	과목명	
2	0	100	수학	
3	1	200	과학	
4	2	300	빅데이터	
5				

- 파일 처리 (File Handling)
- text 파일
 - 1) txt 파일 생성하기
 - 2) txt 파일 읽기
- csv / excel 파일
 - 1) csv, xlsx 파일 생성하기
 - 2) **xlsx 파일 읽기**
 - 3) csv 파일 읽기
- 예외 처리 (Exception Handling)
- Exercise 1
- Exercise 2

- excel 형식 파일 내용 읽기

제공된 people.xlsx 파일을 데이터 디렉토리 [D:/ai/DATA] 에 저장한 후 실습하기

```
1 import openpyxl
2 wb = openpyxl.load_workbook("DATA/people.xlsx")
```



	A	B	C	D	E
1	number	name	email	phone	
2	100	전우치	aaa@gmail.com	02-233-5656	
3	101	일지매	bbb@daum.net	062-535-9085	
4	102	홍길동	ccc@naver.com	032-233-1452	
5					
6					
7					

<참고>

ModuleNotFoundError: No module named 'openpyxl' 발생하면 openpyxl 패키지를 설치한다.

- 파일 처리 (File Handling)
- text 파일
 - 1) txt 파일 생성하기
 - 2) txt 파일 읽기
- csv / excel 파일
 - 1) csv, xlsx 파일 생성하기
 - 2) xlsx 파일 읽기
 - 3) csv 파일 읽기

- 예외 처리 (Exception Handling)

- Exercise 1
- Exercise 2

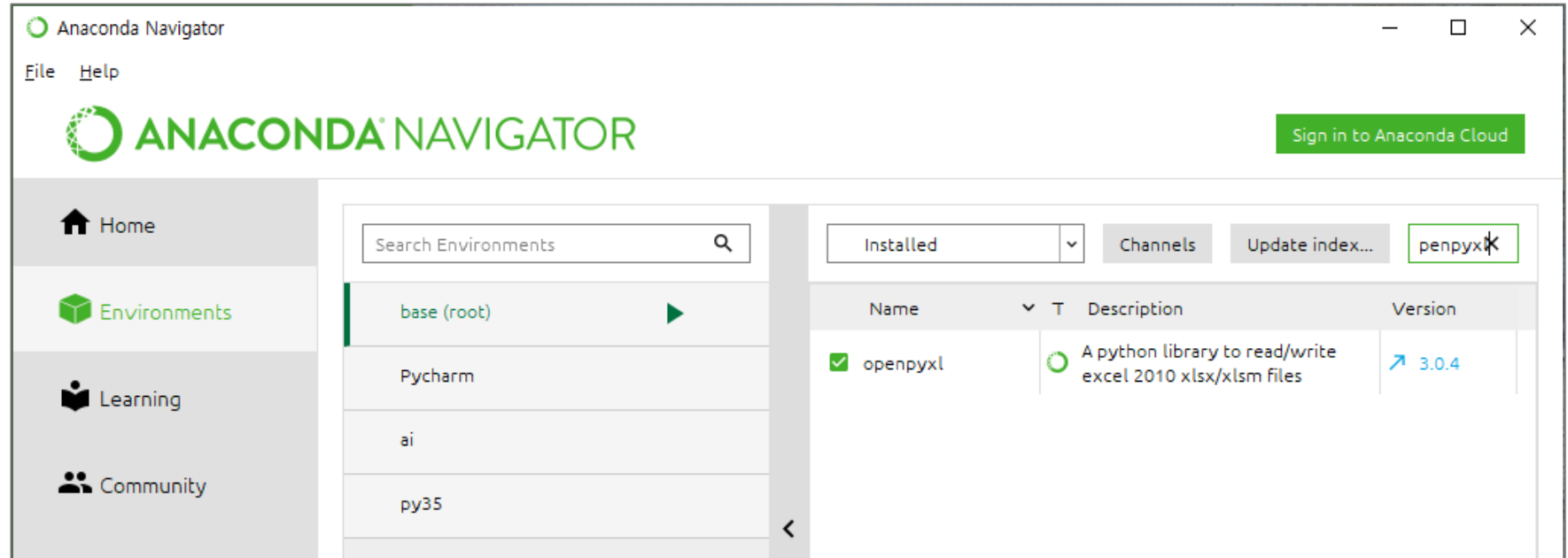
Install of openpyxl

```
(base) C:\Users\tina>conda install openpyxl
```

```
.
```

```
.
```

```
Proceed ([y]/n)? y
```



- 파일 처리 (File Handling)

- text 파일

- 1) txt 파일 생성하기
- 2) txt 파일 읽기

- csv / excel 파일

- 1) csv, xlsx 파일 생성하기
- 2) **xlsx 파일 읽기**
- 3) csv 파일 읽기

```
1 import openpyxl
2 wb = openpyxl.load_workbook("DATA/people.xlsx")
```

```
1 sheet = wb['Sheet1']
2 print("오픈한 엑셀 파일의 워크시트 이름 : ", sheet)
3 print("오픈한 엑셀 파일의 워크시트에서 마지막 데이터 행 번호 : ", sheet.max_row)
```

오픈한 엑셀 파일의 워크시트 이름 : <Worksheet "Sheet1">

오픈한 엑셀 파일의 워크시트에서 마지막 데이터 행 번호 : 4

- 예외 처리

(Exception Handling)

- Exercise 1

- Exercise 2

```
1 a = sheet.cell(2, 2).value
2 print(a)
3 b = sheet.cell(4, 3).value
4 print(b)
```

전우치

ccc@naver.com

엑셀 데이터 사용법 테스트하기

	A	B	C	D	E
1	number	name	email	phone	
2	100	전우치	aaa@gmail.com	02-233-5656	
3	101	일지매	bbb@daum.net	062-535-9085	
4	102	홍길동	ccc@naver.com	032-233-1452	
5					
6					
7					
8					

- 파일 처리 (File Handling)

- text 파일

- 1) txt 파일 생성하기
- 2) txt 파일 읽기

- csv / excel 파일

- 1) csv, xlsx 파일 생성하기
- 2) xlsx 파일 읽기**
- 3) csv 파일 읽기

- 예외 처리

(Exception Handling)

- Exercise 1

- Exercise 2

엑셀 데이터를 딕셔너리에 저장하기

```

1 dic_xlsx = { } #딕셔너리 선언
2
3 for i in range(2, sheet.max_row+1) :
4     number = sheet.cell(row=i , column=1).value
5     print(number)
6
7     name = sheet.cell(row=i , column=2).value
8     print(name)
9
10    email = sheet.cell(row=i , column=3).value
11    print(email)
12
13    phone = sheet.cell(row=i , column=4).value
14    print(phone)
15
16    dic_xlsx[number] = [name, email, phone] # 반복문을 실행하는 동안 데이터가 딕셔너리에 추가됨
17    print(dic_xlsx)
18    print('-'*50)

```

people.xlsx - Excel kim hs

파일 홈 삽입 페이지 i 수식 데이터 검토 보기 개발 도 도움말 입력하세 공유

D19

	A	B	C	D	E
1	number	name	email	phone	
2	100	전우치	aaa@gmail.com	02-233-5656	
3	101	일지매	bbb@daum.net	062-535-9085	
4	102	홍길동	ccc@naver.com	032-233-1452	
5					
6					
7					
8					

Sheet1

130%



- 파일 처리 (File Handling)

- text 파일

- 1) txt 파일 생성하기
- 2) txt 파일 읽기

- csv / excel 파일

- 1) csv, xlsx 파일 생성하기
- 2) **xlsx 파일 읽기**
- 3) csv 파일 읽기

- 예외 처리

(Exception Handling)

- Exercise 1

- Exercise 2



100
전우치
aaa@gmail.com
02-233-5656

```
{100: ['전우치', 'aaa@gmail.com', '02-233-5656']}
```

101
일지매
bbb@daum.net
062-535-9085

```
{100: ['전우치', 'aaa@gmail.com', '02-233-5656'], 101: ['일지매', 'bbb@daum.net', '062-535-9085']}
```

102
홍길동
ccc@naver.com
032-233-1452

```
{100: ['전우치', 'aaa@gmail.com', '02-233-5656'], 101: ['일지매', 'bbb@daum.net', '062-535-9085'], 102: ['홍길동', 'ccc@naver.com', '032-233-1452']}
```

	A	B	C	D	E
1	number	name	email	phone	
2	100	전우치	aaa@gmail.com	02-233-5656	
3	101	일지매	bbb@daum.net	062-535-9085	
4	102	홍길동	ccc@naver.com	032-233-1452	
5					
6					
7					
8					

- 파일 처리 (File Handling)
- text 파일
 - 1) txt 파일 생성하기
 - 2) txt 파일 읽기
- csv / excel 파일
 - 1) csv, xlsx 파일 생성하기
 - 2) xlsx 파일 읽기**
 - 3) csv 파일 읽기
- 예외 처리
(Exception Handling)
- Exercise 1
- Exercise 2

최종 딕셔너리 출력하기

```
1 print(dic_xlsx)
2 print('-'*50)
3 print(type(dic_xlsx))
```

```
{100: ['전우치', 'aaa@gmail.com', '02-233-5656'], 101: ['일지매', 'bbb@daum.net', '062-535-9085'], 102: ['홍길동', 'ccc@naver.com', '032-233-1452']}
```

```
-----
<class 'dict'>
```

딕셔너리 키와 값을 출력하기

```
1 print("딕셔너리의 키 :", dic_xlsx.keys())
2 print("딕셔너리의 값 :", dic_xlsx.values())
```

```
딕셔너리의 키 : dict_keys([100, 101, 102])
```

```
딕셔너리의 값 : dict_values([['전우치', 'aaa@gmail.com', '02-233-5656'], ['일지매', 'bbb@daum.net', '062-535-9085'], ['홍길동', 'ccc@naver.com', '032-233-1452']])
```

- 파일 처리 (File Handling)

- text 파일

- 1) txt 파일 생성하기
- 2) txt 파일 읽기

- csv / excel 파일

- 1) csv, xlsx 파일 생성하기
- 2) xlsx 파일 읽기**
- 3) csv 파일 읽기

- 예외 처리

(Exception Handling)

- Exercise 1

- Exercise 2

딕셔너리에서 원하는 값 추출하기

1) 키를 기반으로 추출

```
1 print(dic_xlsx[100])
2 print(dic_xlsx[101])
3 print(dic_xlsx[102])
```

```
['전우치', 'aaa@gmail.com', '02-233-5656']
['일지매', 'bbb@daum.net', '062-535-9085']
['홍길동', 'ccc@naver.com', '032-233-1452']
```

2) 키와 리스트 내의 값을 인덱스를 기반으로 추출

```
1 print("102번 키의 값 : ", dic_xlsx[102])
2 print("102번 키에서 1번 인덱스의 값 : ", dic_xlsx[102][1])
3 print("102번 키에서 1번 인덱스의 3번 인덱스 값 : ", dic_xlsx[102][1][3])
4 print("102번 키에서 1번 인덱스의 4번부터 끝까지에 해당하는 인덱스 값 : ", dic_xlsx[102][1][4:])
```

102번 키의 값 : ['홍길동', 'ccc@naver.com', '032-233-1452']

102번 키에서 1번 인덱스의 값 : ccc@naver.com

102번 키에서 1번 인덱스의 3번 인덱스 값 : @

102번 키에서 1번 인덱스의 4번부터 끝까지에 해당하는 인덱스 값 : naver.com

- 파일 처리 (File Handling)

- text 파일

- 1) txt 파일 생성하기
- 2) txt 파일 읽기

- csv / excel 파일

- 1) csv, xlsx 파일 생성하기
- 2) xlsx 파일 읽기
- 3) csv 파일 읽기

- 예외 처리

(Exception Handling)

- Exercise 1

- Exercise 2

- csv 형식 파일 내용 읽기

제공된 people_1.csv, people_2.csv 파일을 데이터 디렉토리 [D:/ai/DATA] 에 저장한 후 실습하기

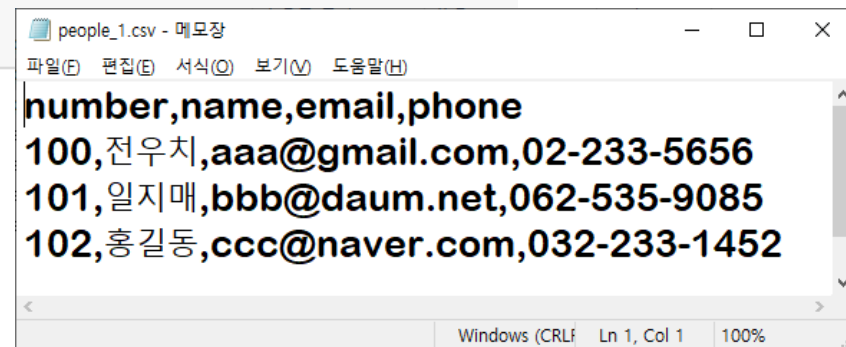
people_1.csv (ANSI 코드로 작성되어 있음)

people_2.csv (UTF-8 코드로 작성되어 있음)

1) people_1.csv (ANSI 코드로 작성되어 있음) 읽기

```
1 import csv
2 f = open('D:/ai/DATA/people_1.csv') # ANSI 코드
3
4 f_csv = csv.reader(f)
5
6 for i in f_csv :
7     print(i)
```

```
['number', 'name', 'email', 'phone']
['100', '전우치', 'aaa@gmail.com', '02-233-5656']
['101', '일지매', 'bbb@daum.net', '062-535-9085']
['102', '홍길동', 'ccc@naver.com', '032-233-1452']
```



- 파일 처리 (File Handling)

- text 파일

- 1) txt 파일 생성하기
- 2) txt 파일 읽기

- csv / excel 파일

- 1) csv, xlsx 파일 생성하기
- 2) xlsx 파일 읽기
- 3) csv 파일 읽기

- 예외 처리

(Exception Handling)

- Exercise 1

- Exercise 2

2) people_2.csv (UTF-8 코드로 작성되어 있음) 읽기

```
: 1 import csv
  2 f = open('D:/ai/DATA/people_2.csv') # UTF-8 코드
  3
  4 f_csv = csv.reader(f)
  5
  6 for i in f_csv :
  7     print(i)
```



UnicodeDecodeError

Traceback (most recent call last)

<ipython-input-16-8498690adc8e> in <module>

4 f_csv = csv.reader(f)

5

----> 6 for i in f_csv :

7 print(i)

UnicodeDecodeError: 'cp949' codec can't decode byte 0xec in position 32: illegal multibyte sequence

- 파일 처리 (File Handling)

- text 파일

- 1) txt 파일 생성하기
- 2) txt 파일 읽기

- csv / excel 파일

- 1) csv, xlsx 파일 생성하기
- 2) xlsx 파일 읽기
- 3) csv 파일 읽기**

- 예외 처리

(Exception Handling)

- Exercise 1

- Exercise 2

people_2.csv (UTF-8 코드로 작성되어 있음) 읽기 에러 해결

```
1 import csv
2 f = open('D:/ai/DATA/people_2.csv', 'r', encoding='utf-8') # UTF8, utf8, UTF-8 모두 동일한 의미
3
4 f_csv = csv.reader(f)
5
6 for i in f_csv :
7     print(i)
```

```
['\ufeffnumber', 'name', 'email', 'phone']
['100', '전우치', 'aaa@gmail.com', '02-233-5656']
['101', '일지매', 'bbb@daum.net', '062-535-9085']
['102', '홍길동', 'ccc@naver.com', '032-233-1452']
```

- 파일 처리 (File Handling)

- text 파일

- 1) txt 파일 생성하기
- 2) txt 파일 읽기

- csv / excel 파일

- 1) csv, xlsx 파일 생성하기
- 2) xlsx 파일 읽기
- 3) csv 파일 읽기**

- 예외 처리

(Exception Handling)

- Exercise 1

- Exercise 2

```
1 import csv
2 f = open('D:/ai/DATA/people_2.csv', 'r', encoding='utf-8-sig')
3                                     # UTF-8-SIG, UTF-8-sig, utf-8-SIG 모두 동일한 의미
4
5 f_csv = csv.reader(f)
6
7 for i in f_csv :
8     print(i)
```

```
['number', 'name', 'email', 'phone']
['100', '전우치', 'aaa@gmail.com', '02-233-5656']
['101', '일지매', 'bbb@daum.net', '062-535-9085']
['102', '홍길동', 'ccc@naver.com', '032-233-1452']
```

utf-8로 인코딩된 파일을 파이썬에서 읽을 경우,
인코딩을 utf-8로 하면 위와 같이 BOM(Byte Order Mark) signature가 생길 수 있다.
utf-8-sig은 BOM 상관없이 인코딩을 변환한 후 읽는다.

- 파일 처리 (File Handling)
- text 파일
 - 1) txt 파일 생성하기
 - 2) txt 파일 읽기
- csv / excel 파일
 - 1) csv, xlsx 파일 생성하기
 - 2) xlsx 파일 읽기
 - 3) csv 파일 읽기
- 예외 처리
(Exception Handling)
- Exercise 1
- Exercise 2

[문법]

try:

예외가 발생할 문장

except 예외 종류 :

예외가 발생하면 실행할 문장

else :

예외가 발생하지 않을 때 실행할 문장

예외 처리(Exception Handling)

- 파일 처리 (File Handling)

- text 파일
 - 1) txt 파일 생성하기
 - 2) txt 파일 읽기

- csv / excel 파일
 - 1) csv, xlsx 파일 생성하기
 - 2) xlsx 파일 읽기
 - 3) csv 파일 읽기

- 예외 처리
(Exception Handling)

- Exercise 1
- Exercise 2

1) 숫자를 입력해야 하는데 문자를 입력하는 예외 상황 발생

```
1 no1 = int(input('숫자를 입력하세요: '))
2
3 no2 = int(input('숫자를 입력하세요: '))
```

숫자를 입력하세요: 10
 숫자를 입력하세요: A

```
ValueError                                Traceback (most recent call last)
<ipython-input-4-150cc84c7401> in <module>
      1 no1 = int(input('숫자를 입력하세요: '))
      2
----> 3 no2 = int(input('숫자를 입력하세요: '))
```

ValueError: invalid literal for int() with base 10: 'A'

< ValueError 예외 처리 방법 >

```
1 try :
2     no1 = int(input('숫자 1개를 입력하세요: '))
3 except ValueError :
4     print('숫자를 입력해야 합니다. ')
5
```

숫자 1개를 입력하세요: A
 숫자를 입력해야 합니다.

- 파일 처리 (File Handling)

- text 파일
 - 1) txt 파일 생성하기
 - 2) txt 파일 읽기

- csv / excel 파일
 - 1) csv, xlsx 파일 생성하기
 - 2) xlsx 파일 읽기
 - 3) csv 파일 읽기

- 예외 처리
(Exception Handling)

- Exercise 1
- Exercise 2

2) 0으로 나누었을 경우 예외 상황 발생

```
1 no1 = 10
2 no2 = 0
```

```
1 print(no1 / no2)
```

```
ZeroDivisionError                                Traceback (most recent call last)
<ipython-input-7-444d309cb332> in <module>
----> 1 print(no1 / no2)
```

ZeroDivisionError: division by zero

< ZeroDivisionError 예외 처리 방법 >

```
1 try :
2     print( no1 / no2)
3 except ZeroDivisionError :
4     print(" 0으로 나눌수 없습니다")
```

0으로 나눌수 없습니다

예외 처리(Exception Handling)

- 파일 처리 (File Handling)

- text 파일

- 1) txt 파일 생성하기
- 2) txt 파일 읽기

- csv / excel 파일

- 1) csv, xlsx 파일 생성하기
- 2) xlsx 파일 읽기
- 3) csv 파일 읽기

- 예외 처리
(Exception Handling)**

- Exercise 1

- Exercise 2

3) 다음과 같이 2개의 예외 상황 처리

① ValueError

② ZeroDivisionError

```
1 try :
2     no1 = int(input('숫자 1개를 입력하세요:'))
3     no2 = int(input('숫자 1개를 입력하세요:'))
4     print(no1/no2)
5 except (ValueError, ZeroDivisionError) :
6     print("예외 상황이 발생했습니다.")
```

숫자 1개를 입력하세요:A
예외 상황이 발생했습니다.

```
1 try :
2     no1 = int(input('숫자 1개를 입력하세요:'))
3     no2 = int(input('숫자 1개를 입력하세요:'))
4     print(no1/no2)
5 except (ValueError, ZeroDivisionError) :
6     print("예외 상황이 발생했습니다.")
```

숫자 1개를 입력하세요:30
숫자 1개를 입력하세요:0
예외 상황이 발생했습니다.

```
1 try :
2     no1 = int(input('숫자 1개를 입력하세요:'))
3     no2 = int(input('숫자 1개를 입력하세요:'))
4     print(no1/no2)
5 except (ValueError, ZeroDivisionError) :
6     print("예외 상황이 발생했습니다.")
```

숫자 1개를 입력하세요:30
숫자 1개를 입력하세요:5
6.0

예외 처리(Exception Handling)

- 파일 처리 (File Handling)

- text 파일

- 1) txt 파일 생성하기
- 2) txt 파일 읽기

- csv / excel 파일

- 1) csv, xlsx 파일 생성하기
- 2) xlsx 파일 읽기
- 3) csv 파일 읽기

- 예외 처리
(Exception Handling)**

- Exercise 1

- Exercise 2

3) 다음과 같이 2개의 예외 상황 처리 (방법 2)

① ValueError

② ZeroDivisionError

```
1 try :
2     no1 = int(input('숫자 1개를 입력하세요:'))
3     no2 = int(input('숫자 1개를 입력하세요:'))
4     print(no1/no2)
5 except ValueError :
6     print("숫자를 입력해야 합니다.")
7 except ZeroDivisionError :
8     print("0으로 나눌 수 없습니다.")
```

숫자 1개를 입력하세요:A

숫자를 입력해야 합니다.

```
1 try :
2     no1 = int(input('숫자 1개를 입력하세요:'))
3     no2 = int(input('숫자 1개를 입력하세요:'))
4     print(no1/no2)
5 except ValueError :
6     print("숫자를 입력해야 합니다.")
7 except ZeroDivisionError :
8     print("0으로 나눌 수 없습니다.")
```

숫자 1개를 입력하세요:30

숫자 1개를 입력하세요:0

0으로 나눌 수 없습니다.

```
1 try :
2     no1 = int(input('숫자 1개를 입력하세요:'))
3     no2 = int(input('숫자 1개를 입력하세요:'))
4     print(no1/no2)
5 except ValueError :
6     print("숫자를 입력해야 합니다.")
7 except ZeroDivisionError :
8     print("0으로 나눌 수 없습니다.")
```

숫자 1개를 입력하세요:30

숫자 1개를 입력하세요:5

6.0

예외 처리(Exception Handling)

- 파일 처리 (File Handling)

- text 파일

- 1) txt 파일 생성하기
- 2) txt 파일 읽기

- csv / excel 파일

- 1) csv, xlsx 파일 생성하기
- 2) xlsx 파일 읽기
- 3) csv 파일 읽기

- 예외 처리**
(Exception Handling)

- Exercise 1
- Exercise 2

3) 다음과 같이 2개의 예외 상황 처리 (방법 3)

① ValueError

② ZeroDivisionError

```

1  try :
2      no1 = int(input('숫자 1개를 입력하세요:'))
3      no2 = int(input('숫자 1개를 입력하세요:'))
4      no3 = no1/no2
5  except ValueError :
6      print("숫자를 입력해야 합니다.")
7  except ZeroDivisionError :
8      print("0으로 나눌 수 없습니다.")
9  else :
10     print("%s / %s = %s " % (no1, no2, no3)) #예외 상황 아닐 경우 실행
11

```

숫자 1개를 입력하세요:30

숫자 1개를 입력하세요:5

30 / 5 = 6.0

예외 처리(Exception Handling)

- 파일 처리 (File Handling)

- text 파일

- 1) txt 파일 생성하기
- 2) txt 파일 읽기

- csv / excel 파일

- 1) csv, xlsx 파일 생성하기
- 2) xlsx 파일 읽기
- 3) csv 파일 읽기

- 예외 처리**
(Exception Handling)

- Exercise 1

- Exercise 2

4) 예외 상황을 명시하지 않고 모든 예외 상황에 적용하는 방법

```

1 try :
2     no1 = int(input('숫자 1개를 입력하세요:'))
3     no2 = int(input('숫자 1개를 입력하세요:'))
4     print(no1/no2)
5 except :
6     print("예외 상황이 발생했습니다.")

```

숫자 1개를 입력하세요:A
예외 상황이 발생했습니다.

```

1 try :
2     no1 = int(input('숫자 1개를 입력하세요:'))
3     no2 = int(input('숫자 1개를 입력하세요:'))
4     print(no1/no2)
5 except :
6     print("예외 상황이 발생했습니다.")

```

숫자 1개를 입력하세요:30
숫자 1개를 입력하세요:0
예외 상황이 발생했습니다.

```

1 try :
2     no1 = int(input('숫자 1개를 입력하세요:'))
3     no2 = int(input('숫자 1개를 입력하세요:'))
4     print(no1/no2)
5 except :
6     print("예외 상황이 발생했습니다.")

```

숫자 1개를 입력하세요:30
숫자 1개를 입력하세요:5
6.0

- 파일 처리 (File Handling)
- text 파일
 - 1) txt 파일 생성하기
 - 2) txt 파일 읽기
- csv / excel 파일
 - 1) csv, xlsx 파일 생성하기
 - 2) xlsx 파일 읽기
 - 3) csv 파일 읽기
- 예외 처리 (Exception Handling)
- **Exercise 1**
- Exercise 2

사용자에게 폴더명을 하나 입력 받아서 해당 폴더가 존재하지 않을 경우에는 해당 폴더를 만들고, 해당 폴더가 존재할 경우 “**폴더명_2**”의 이름으로 폴더를 만들도록 코드를 작성하시오.

[출력 형식]

폴더 이름을 입력하세요(예:c:\temp) :
입력하신 G:\report 경로의 폴더를 생성했습니다

폴더 이름을 입력하세요(예:c:\temp) :
입력하신 경로가 존재하여 g:\wex_2 로 폴더를 생성했습니다

- 파일 처리 (File Handling)
- text 파일
 - 1) txt 파일 생성하기
 - 2) txt 파일 읽기
- csv / excel 파일
 - 1) csv, xlsx 파일 생성하기
 - 2) xlsx 파일 읽기
 - 3) csv 파일 읽기
- 예외 처리
(Exception Handling)
- Exercise 1
- **Exercise 2**

다운받은 "**url.txt**" 파일을 읽어서 split('/'), append(), len() 함수 등을 활용하여 각 url 주소별로 카운트 한 후 [출력 형식]과 같이 출력되는 코드를 작성하시오.

[출력 형식]



```
url.txt - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말
https://blog.naver.com/abc123
https://blog.tistory.com/aaa123
https://blog.blog.me/bbb456
https://blog.tistory.com/abc1234
https://blog.blog.me/ddd432
https://blog.naver.com/bbaacc234
https://blog.naver.com/aacc5435
```

```
['https:', '', 'blog.naver.com', 'abc123#n']
['https:', '', 'blog.tistory.com', 'aaa123#n']
['https:', '', 'blog.blog.me', 'bbb456#n']
['https:', '', 'blog.tistory.com', 'abc1234#n']
['https:', '', 'blog.blog.me', 'ddd432#n']
['https:', '', 'blog.naver.com', 'bbaacc234#n']
['https:', '', 'blog.naver.com', 'aacc5435']
=====
['blog.naver.com', 'blog.naver.com', 'blog.naver.com']
['blog.tistory.com', 'blog.tistory.com']
['blog.blog.me', 'blog.blog.me']
=====
blog.naver.com 의 갯수는 모두 3 개입니다
blog.tistory.com 의 갯수는 모두 2 개입니다
blog.blog.me 의 갯수는 모두 2 개입니다
```