DS 부트 캠프 2024

한상곤(Sangkon Han)
부산대학교 정보컴퓨터공학부
2024/02/08

목차

파일과 예외처리

예외(exception)

- 예상하지 못한 상황이 발생하여 프로그램이 정상적으로 동작하지 않는 것
- 예측 가능한 예외, 발생 여부를 개발자가 사전에 인지할 수 있는 예외
  - 개발자는 예외를 예측하여 예외가 발생할 때는 대응책을 마련해 놓을 수 있음
  - 대표적으로 사용자 입력란에 값이 잘못 입력되었다면 if문을 사용하여 잘못 입력하였다고 응답하는 방법이 있음
  - 아주 쉽게 대응이 가능함
- 예측 불가능한 예외
  - 대표적으로 매우 많은 파일을 처리할 때 문제가 발생할 수 있음
  - 예측 불가능한 예외 발생시 인터프리터가 자동으로 사용자에게 알려줌
  - 예외 처리는 제품의 완성도를 높이는 차원에서 매우 중요한 개념임

try-except

- 파이썬에서 예외 처리의 기본 문법이 try-except문
- `try`문에 예외 발생이 예상되는 코드를 적고, `except`문에 예외 발생 시 대응하는 코드를 작성함

```
try:
    예외 발생 가능 코드
except 예외 타입:
    예외 발생 시 실행되는 코드
```

```
for i in range(10):
    try:
        print(10 / i)
    except ZeroDivisionError as e:
        print(e)
        print("Not divided by 0")
```

try-except-else / finally

```
for i in range(10):
    try
        result = 10 / i
    except ZeroDivisionError:
        print("Not divided by 0")
    else:
        print(10 / i)
```

```
try:
    for i in range(1, 10):
        result = 10 // i
        print(result)
except ZeroDivisionError:
    print("Not divided by 0")
finally:
    print("종료되었다.")
```

raise / assert

```
def get_binary_number(decimal_number):
    assert isinstance(decimal_number, int)
    return bin(decimal_number)
print(get_binary_number(10))
print(get_binary_number("10"))
```

```
while True:
    value = input("변환할 정수값을 입력해 주세요: ")\
    for digit in value:
        if digit not in "0123456789":
        raise ValueError("숫자값을 입력하지 않았습니다.")
    print("정수값으로 변환된 숫자 -", int(value))
```

파일 종류

- 바이너리 파일(binary file)
  - 컴퓨터만 이해할 수 있는 이진 정보로 저장된 파일
  - 비트(bit) 형태로 저장되어 메모장으로 열면 내용이 보이지 않거나 내용을 확인할 수 없는 파일
- 텍스트 파일(text file)
- 사람이 이해할 수 있는 문자열로 저장된 파일
- 메모장으로 그 내용을 확인할 수 있음

파일 읽기

```
# f = open("파일명, "모드")
f = open("dream.txt", "r")
contents = f.read()
print(contents)
f.close()

with open("dream.txt","r") as my_file:
    content_list = my_file.readlines()  # 파일 전체를 리스트로 반
    print(type(content_list))           # 자료형 확인
    print(content_list)                 # 리스트값 출력

with open("dream.txt", "r") as my_file:
    i = 0
    while 1:
        line = my_file.readline()
        if not line:
            break
        print(str(i)+" === "+ line.replace("\n","")) # 한 줄씩 값 출력
        i = i + 1
```

모듈과 활용

모듈

- 프로그래밍에서의 모듈: 작은 프로그램 조각
  - 하나하나 연결해 어떤 목적을 가진 프로그램을 만들기 위한 작은 프로그램
- 인터페이스: 함수에서 매개변수를 입력하는 약속
  - 해당 모듈을 사용하기 위해서는 모듈 간의 연결을 위한 약속이 필요한데, 이것을 인터페이스라고 함
- 내장 모듈
- 파이썬에서는 .py 파일 자체가 모듈임.

람다와 리스트 축약

람다(lambda) 함수

- 람다(lambda) 함수: 함수의 이름 없이 함수처럼 사용할 수 있는 익명의 함수

```
def f(x, y):
    return x + y
print(f(1, 4)) # 5

f = lambda x, y: x + y
print(f(1, 4))
```

파이썬 스타일 코드

- for문을 사용하여 단어들을 연결하여 출력하는 가장 일반적인 코드

```
# This is not pythonic code.
colors = ['red', 'blue', 'green', 'yellow']
result = ''
for s in colors:
    result += s
print(result) # redbluegreenyellow
```

- 우리는 이렇게 작성해야 함

```
# It's pythonic!
colors = ['red', 'blue', 'green', 'yellow']
result = ''.join(colors)
print(result) # redbluegreenyellow
```

왜? 파이썬 스타일 코드

- 파이썬의 철학
  - "인간의 시간이 컴퓨터의 시간보다 더 중요하다."
- 다양한 파이썬 스타일 코드
  - `split()`, `join()`, `리스트 컴프리헨션(list comprehension)`, `enumerate()`, `zip()`과 같은 기본적인 개념부터 `map()`과 `reduce()`와 같은 상위 개념까지 포함
- 파이썬 스타일 코드는 왜 배워야 할까?
  - 파이썬으로 작성된 프로그램 대부분은 파이썬 스타일 코드의 특징을 포함하므로 파이썬 스타일 코드를 알아야 다른 사람이 작성 한 코드를 쉽게 이해할 수 있음
- 효율적인 프로그래밍 측면
  - 코드 자체도 간결해지고 코드 작성 시간도 단축시킴

리스트 컴프리헨션(list comprehension)

- 리스트 컴프리헨션(list comprehension)
  - 기존 리스트형을 사용하여 간단하게 새로운 리스트를 만드는 기법

```
result = [ ]
for i in range(10):
    result.append(i)
print(result) # [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

result = [i for i in range(10)]
print(result) # [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

필터링

- if문과 함께 사용하는 리스트 컴프리헨션

```
result = [ ]
for i in range(10):
    if i % 2 == 0:
        result.append(i)
print(result) # [0, 2, 4, 6, 8]

result = [i for i in range(10) if i % 2 == 0]
print(result)
result = [i if i % 2 == 0 else 10 for i in range(10)]
print(result)
```

중첩 반복문(nested loop)

- 리스트 컴프리헨션에서도 리스트 2개를 섞어 사용할 수 있음

```
word_1 = "Hello"
word_2 = "World"
result = [i + j for i in word_1 for j in word_2]   # 중첩 반복문
print(result) # ['HW', 'Ho', ..., 'od']
```

과제3. CSV 파일 처리 (1)

assignment03.csv 파일을 사용해서 다음과 같은 자료구조 형태로 재구성하는 코드를 작성하세요.

- CSV 파일의 내용은 다음과 같다.
  - `chart_week`, 해당 주차의 시작일(`2022-01-01`)
  - `current_week`, 해당 주차의 순위(`1`)
  - `title`, 노래 제목(`All I Want For Christmas Is You`)
  - `performer`, 가수명(`Mariah Carey`)
  - `last_week`, 이전 주차의 순위(`1`)
  - `peak_pos`, 차트의 가장 높은 순위(`1`)
  - `wks_on_chart`, 차트에 유지된 주차(`50`)

과제4. CSV 파일 처리 (2)

아래 질문을 해결할 수 있는, 함수를 작성하세요.

1. 발매된 곡들의 목록을 출력하세요(중복은 제거하세요).
2. 각 가수들의 1위한 곳을 합산하여, 내림차순으로 출력하세요. 단, 동일한 1위 횟수를 가진 가수들은, 발매년도가 빠른 순으로 출력하세요.
3. 노래 제목에 포함된 단어의 빈도수를 출력하세요.
4. 년도별 가장 오래 차트에 머무른 곡의 제목을 출력하세요.