1번. AirConditional()

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

* 핵심코드

1-1. 전원 on/off

def Control\_Boolean(self):

        Boolean=input('전원 On/Off 중에 골라주세요: ')

        self.Boolean=Boolean

전원의 유무를 Boolean에 입력받은 후 self.Boolean=Boolean으로 멤버변수화합니다.

1-2. power return

def power(self):

        if  self.Boolean== '전원 On':

            self.Boolean=True

            return self.Boolean

        elif self.Boolean == '전원 Off':

            self.Boolean=False

            return self.Boolean

입력받은 전원이 on일 경우 True로 바꾼 후 return시켜주고 off일 경우 False로 바꾼 후

Return 시킵니다

1-3. 온도 입력

def Control\_Temp(self):

        self.temp=int(input('온도를 입력하세요: '))

온도를 입력받아 self.temp에 저장합니다.

1-4. 온도 상승

def up(self):

        self.temp+=1

        print(self.temp)

온도를 1도 상승시키고 이를 출력합니다

1-5. 온도 하강

def down(self):

        self.temp-=1

        print(self.temp)

온도를 1도 하강시키고 이를 출력합니다

1-6. 온도 리턴

def re(self):

        temp=self.temp

        return temp

‘

현재온도 temp를 return시킵니다

1-7. 현재 온도 출력

def present(self):

        print('현재 온도는 {}도입니다'.format(self.temp))

format을 활용하여 현재 온도 self.temp를 출력합니다.

1-8 전원 on/off 판별 example

if power == True:

            a.Control\_Temp()

        else:

            print('전원이 꺼져있습니다')

만약 전원이 켜져 있다면 해당 함수를 호출하고 꺼져 있다면 전원이 꺼져 있다고 출력하고 함수를 호출하지 않습니다.

<구현방식에 대한 간단한 설명>

Class Air에 모든 요구되는 함수를 구현합니다

While True를 사용하여 무한 반복문을 돌리고 에어컨 리모컨의 설정들을 해당 숫자를 입력하면 전원이 켜져 있다면 해당 함수를 호출하고 전원이 꺼져 있다면 전원이 꺼져 있다고 출력한 후

해당 함수를 호출하지 않습니다.

2번. Csv 파일 불러오기

테이블이(가) 표시된 사진

자동 생성된 설명

테이블이(가) 표시된 사진

자동 생성된 설명

테이블이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

2-1. csv 파일 불러오기

import pandas as pd

csv\_test=pd.read\_csv("D:\파이썬기말고사 파일\diamonds.csv")

csv\_test로 파이썬기말고사 파일에 있는 diamonds.csv 파일을 읽습니다.

2-2 상위 5개의 row 출력

row\_5=csv\_test.loc[0:5, :]

print(row\_5)

상위 열 5개를 ioc를 활용하여 불러옵니다

2-3 2번째 column인 ‘cut’을 삭제 후 상위 5개 출력

new=csv\_test.drop(['cut'], axis=1)

new\_id=new.iloc[0:5, :]

print(new\_id)

drop을 사용하여 cut을 삭제한 후 iloc를 활용하여 상위 5개 열을 불러오고 출력합니다

2-4. price column을 내림차순으로 정렬하여 출력

re=csv\_test.sort\_values(by='price', ascending=False)

col=re['price']

print(col)

re에 price를 기준으로 accending=False를 사용하여 내림차순으로 정렬합니다.

Col에 re[‘price’]를 넣은 후 이를 출력합니다.

2-5. 2~4번 row 및 1~2번 column을 출력

special = csv\_test.iloc[2:5, 0:2]

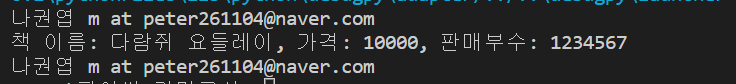
print(special)

iloc을 사용하여 해당 열과 행을 기준으로 special에 저장 후 출력합니다.

<구현방식에 대한 간단한 설명>

Diamonds.csv 파일을 불러온 후 이를 각 변수에 원하는 조건에 맞추어 출력합니다

3번. Author 클래스와 Book 클래스



3-1. Author class

class Author:

    def get\_name(self,name):

        self.name=name

    def get\_email(self, email):

        for i in range(len(email)):

            global count

            count=0

            if(email[i]=='@'):

                self.email=email

                count+=1

                break

            else:

                continue

        if count==0:

            print('잘못된 입력입니다 강제종료합니다')

            exit()

    def set\_email(self, email):

        self.email=email

    def get\_gender(self, gender):

        if(gender == 'm' or 'f'):

            self.gender=gender

        else:

            gender='u'

            self.gender=gender

    def print(self):

        print('{0} {1} at {2}'.format(self.name, self.gender, self.email))

get\_email에서 @가 없다면 예외 처리로 강제 종료시킵니다.

Get\_gender에서 젠더가 m이나 f라면 그대로 집어넣고 아니라면 gender는 u로 바꾼후 self.gender에 넣습니다.

print에서 정해진 형식에 따라 출력시킵니다.

3-2. Book class

class Book(Author):

    def \_\_init\_\_(self, name, price, QtyInStock):

        self.name=name

        self.price=price

        self.QtyInStock=QtyInStock

    def get\_name(self, name):

        self.name=name

    def get\_Author(self):

        a.print()

    def get\_price(self, price):

        self.price=price

    def set\_price(self):

        return self.price

    def get\_QtyInStock(self, QtyInStock):

        self.QtyInStock=QtyInStock

    def set\_QtyInStock(self):

        return self.QtyInStock

    def print(self):

        print(f"책 이름: {self.name}, 가격: {self.price}, 판매부수: {self.QtyInStock}")

get\_x로 변수값을 입력받는 함수를 만들고 set함수로 변수를 return 시켜줍니다

get\_Author에서 a.print()로 author 클래스의 print 함수를 출력시킵니다

<구현방식에 대한 간단한 설명>

각 클래스의 get함수를 통해 필요한 정보를 입력받은 후 이 값을 멤버변수를 사용하여 print함수에서 꺼낸 후 이를 출력한다.

Get\_author의 경우 author의 print 함수를 호출하기 위해 a=author()을 한 후 a.print()를 사용하여

A의 print함수를 호출하여 book 클래스의 print 함수와 차별을 둔다

4번. 음원 추천

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

4-1. 음악 셋

def set\_Title(self):

        title=input('노래 제목을 입력하세요: ')

        self.title = title

노래 제목을 입력받음

4-2. 아티스트 셋

def set\_Artist(self):

        artist=input('아티스트 이름을 입력하세요: ')

        self.artist = artist

아티스트를 입력받음 + 예외처리

4-3. 장르 셋

def set\_Genre(self):

        try:

            genre=input('장르를 입력하세요: ')

            if genre != '발라드' or '댄스' or '힙합' or '록' or '트로트':

                raise Exception('지원하지 않는 장르입니다')

            self.genre = genre

        except Exception as e:

            print('예외가 발생했습니다.', e)

장르를 입력받음 +예외처리

4-4. 언어 셋

def set\_Language(self):

        try:

            language=input('언어를 입력하세요: ')

            if language != '한국어' or '영어' or '일본어':

                raise Exception()

            self.language=language

        except Exception as e:

            print('예외가 발생했습니다', e)

언어를 입력받음

4-5. 음원 저장

def save(self):

        list=[]

        list.append(self.title)

        list.append(self.genre)

        list.append(self.artist)

        list.append(self.language)

        with open("D:\파이썬 기말고사\music.dat", mode='a') as file:

            file.write('\n{}, {}, {}, {}'.format(list[0], list[1], list[2], list[3]))

            file.close()

저장할 음원을 일시 저장한 후 music.dat을 불러와 이어서 써서 저장한다

4-6. 음원 조회

def recall(self):

        print('음원 조회를 시작합니다'+'\n')

        print('1. 음원 제목' + '\n')

        print('2. 음원 장르' + '\n')

        print('3. 음원 아티스트' + '\n')

        print('4. 음원 언어' + '\n')

        obtion=int(input('조회할 음원의 옵션을 선택해주세요: '))

        if obtion == 1:

            compared=input('조회할 음원 제목을 입력하세요: ')

        elif obtion == 2:

            compared=input('조회할 음원 장르를 입력하세요: ')

        elif obtion == 3:

            compared=input('조회할 음원 아티스트를 입력하세요: ')

        elif obtion == 4:

            compared=input('조회할 음원 언어를 입력하세요: ')

        else:

            print('잘못된 입력입니다')

        with open("D:\파이썬 기말고사\music.dat", mode='r', encoding='UTF-8') as file:

            while True:

                line=file.readline()

                print(line)

                if not line:

                    break

                list.append(line.strip())

            file.close()

        for i in range(len(list)):

            compare=list[i]

            compare\_list = compare.split(', ')

            if compare\_list[obtion-1] == compared:

                print(compare\_list)

음원 조회의 옵션을 정한 후 내용을 입력하면 리스트에 파일에 저장되어 있는 음원을 임시 저장한 후 split으로 일차원 리스트로 만든 후 옵션에 따라 비교한 후 만약 옵션이 일치한다면

출력한다.

4-7. 음원 수정

def input(self):

        print('수정할 음원의 정보를 입력하세요' + '\n')

        m.set\_Title()

        m.set\_Genre()

        m.set\_Artist()

        m.set\_Language()

        list=[]

        list.append(self.title)

        list.append(self.genre)

        list.append(self.artist)

        list.append(self.language)

        compare=(', '.join(list))

        self.compare=compare

    def input\_re(self):

        print('\n')

        print('수정 정보를 입력하세요' +'\n')

        m.set\_Title()

        m.set\_Genre()

        m.set\_Artist()

        m.set\_Language()

        modify\_list=[]

        modify\_list.append(self.title)

        modify\_list.append(self.genre)

        modify\_list.append(self.artist)

        modify\_list.append(self.language)

        modified=(', '.join(modify\_list))

        self.modified=modified

    def modify(self):

        with open("D:\파이썬 기말고사\music.dat", mode='r') as file:

            lines = file.readlines()

        with open("D:\파이썬 기말고사\music.dat", mode='w') as file:

            for line in lines:

                if line.strip("\n") != self.compare:

                    file.write(line)

                else:

                    file.write(self.modified)

                    file.write('\n')

수정할 음원과 바꿀 내용을 각각 다른 함수를 사용하여 각각 self.compare과 self.modified에 저장한다. 파일을 불러온 후 만약 두 변수가 다르다면 그대로 line을 출력시키고, 같다면 수정하는

음원이 있는 줄이기 때문에 수정된 내용을 출력시키고 줄변환한다

4-8. 음원 삭제

def delate(self):

        list=[]

        list.append(self.title)

        list.append(self.genre)

        list.append(self.artist)

        list.append(self.language)

        compare=(', '.join(list))

        with open("D:\파이썬 기말고사\music.dat", mode='r') as file:

            lines = file.readlines()

        with open("D:\파이썬 기말고사\music.dat", mode='w') as file:

            for line in lines:

                if line.strip("\n") != compare:

                    file.write(line)

compare에 삭제할 음원을 저장한 후 파일에서 불러와 compare과 비교하여 만약 같지 않다면

대상이 아니므로 그대로 출력하고 아니라면 출력하지 않는다.

4-9. 음원 추천 코사인 함수

def cosine(self):

        list\_h=[]

        with open("D:\파이썬 기말고사\history.dat", encoding= 'UTF-8',mode='r') as file:

            while True:

                line=file.readline()

                if not line:

                    break

                list\_h.append(line.strip())

            file.close()

        h\_list=[]

        hh\_list=[]

        for i in range(len(list\_h)):

            h\_list=list\_h[i].split(', ')

            h\_list.pop(0)

            hh\_list.append(h\_list)

        print(hh\_list)

        with open("D:\파이썬 기말고사\music.dat", mode='r') as file:

            while True:

                line=file.readline()

                if not line:

                    break

                list.append(line.strip())

            file.close()

        dat\_list=[]

        for i in range(len(list)):

            dat\_list=list[i].split(', ')

            dat\_list.pop(0)

            print(dat\_list)

        file.close()

hh\_list에 히스토리 닷의 음원의 제목 장르 언어 정보를 저장했다

dat\_list에 뮤직 닷의 음원의 제목 장르 언어 정보를 저장했다.