```
class Cac:
    def test(n1,n2,n3):
        print(2222222222222222)
        print(n1 + n2 )
        return n1 + n2

print(111111111111111111111)
print(Cac.test(1,2,3))
print(33333333333333333)


class Cac:
    def test(n1,n2,n3):
        a = n1+n2+n3
        print(a)
        # return n1+n2+n3

a = Cac.test(4,5,6)


class Cac:
    def sum(n1, n2):
        print(n1 + n2)
    def product(n1, n2):
        return n1 * n2
    def divide(n1, n2):
        return int(n1 / n2)

print(Cac.sum(4,2))
print(Cac.product(4,2))
print(Cac.divide(4,2))
```

# 주소록

## 주소록 만들기

```
def menu_print():
    print("====================================")
    print("1.입력", end="\t")
    print("2.조회", end="\t")
    print("3.전체", end="\t")
    print("4.삭제", end="\t")
    print("5.수정", end="\t")
    print("6.종료")
    print("====================================")



#addr_list = []
```

```
addr_list = [ {"name":"홍길동",  "tel":"010"},
              {"name":"홍길동",  "tel":"555"},
              {"name":"아무개",  "tel":"111"}
            ]
while(1):
    menu_print()
    # 입력명령받기
    cmd = input("명령어입력 : ")

    if cmd=="6" :
        break

    elif cmd=="1" :
        name = input("이름 : ")
        tel = input("전화번호 : ")
        addr_list.append({"name":name,"tel":tel})

    elif cmd=="3":
        print("총",len(addr_list),"건")
        for addr_dic in addr_list:
            print(addr_dic["name"],"\t",addr_dic["tel"] )

    elif cmd=="2":
        fd_name = input("찾을 이름 : ")
        search_list = []
        for fd in addr_list:
            if fd["name"] == fd_name:
                search_list .append(fd["tel"])
        if len(search_list )<=0:
            print("찾는 이름이 없습니다")
        else :
            for search_tel in search_list  :
             print(search_tel)

    elif cmd =="4":
        search_tel= input("삭제할 전화번호 :")
        #--1)
        for addr_dic  in addr_list:
            if addr_dic["tel"] == search_tel:
                # del dl["tel"]
                addr_list.remove(addr_dic )
                print("삭제되었습니다")
        #--2)
        # for addr_dic in addr_list:
        #     if addr_dic["tel"] == search_tel:
        #         addr_list.remove(addr_dic)

        #--3)
        # ------------------------------------------------------------------------
-------
        # addr_list = [1, 2 , 3 ,'AA', {"name":"홍길동", "tel":123}]
        # # addr_list.pop(0)        #index삭제, 그냥은 맨뒤삭제
        # # addr_list.remove( {"name":"홍길동", "tel":123} )  #AA라는값삭제
        # # del addr_list[0]
        #
```

```
        # for i, val in enumerate(addr_list): #enumerate() : 값이 몇번재 값인지 알
려줌
                                              # for 몇번째인지 알려줄 변수, 변수(값)
in enumerate(list)
        #     print(i, val)
        #     if val == "AA":
        #         del addr_list[i]
        # print(addr_list)
        # ----------------------------------------------------------------------
-------
        # for i, addr_dic in enumerate(addr_list):  # enumerate() : 값이 몇번재 값
인지 알려줌
        #     if addr_dic["tel"] == search_tel:
        #         addr_list.pop(i)

        #--4)
        # for i in range(len(addr_list)):
        #     if addr_list[i]["tel"] == del_tel:
        #         addr_list.pop[i]["tel"]

        # --5)
        # isdel = False
        # search_tel = input("삭제 전화번호:")
        # for i, addr_dic in enumerate(addr_list):
        #     if addr_dic["tel"] == search_tel:
        #         yn = input("정말 삭제하시겠습니까?(Y/N)")
        #         if yn.upper() == "Y":
        #             # del addr_list[i]
        #             addr_list.pop(i)
        #             isdel = True
        #
        # if isdel == True:
        #     print("삭제되었습니다")
        # elif isdel == False:
        #     print("검색 결과가 없습니다")

    elif cmd == "5":
            isud = False
            search_tel=input("수정할 대상의 전화번호 : ")
            for addr_dic  in addr_list:
                if addr_dic["tel"] == search_tel:
                    ud_tel = input("수정 전화번호 : ")
                    addr_dic["tel"] = ud_tel
                    print(addr_dic)
                    isud = True

            if isud == True:
                print("수정되었습니다")
            elif isud == False:
                print("검색 결과가 없습니다")
```

## 주소록 함수

```
addr_list = [ {"name":"홍길동",  "tel":"010"},
              {"name":"홍길동",  "tel":"555"},
              {"name":"아무개",  "tel":"111"}
            ]
#-------------------------------------------------------------------------------
-----------------
#입력, 삭제, 수정시 저장하기
def addr_file_write():
    with
open(file="C:\\AI\\pythonProject\\PycharmProjects\\pythonProject\\venv\\test\\ad
dress.txt", mode='w', encoding = "utf-8") as f:
        f.write("name   tel\n")
        for addr_dic in addr_list:
            temp =addr_dic["name"]+"\t"+addr_dic["tel"]+"\n"
            f.write(temp)
#-------------------------------------------------------------------------------
-----------------
def menu_print():
    print("====================================")
    print("1.입력", end="\t")
    print("2.조회", end="\t")
    print("3.전체", end="\t")
    print("4.삭제", end="\t")
    print("5.수정", end="\t")
    print("6.종료")
    print("====================================")

def add_input():
    name = input("이름 : ")
    tel = input("전화번호 : ")
    addr_list.append({"name": name, "tel": tel})

def add_search_all():
    print("총", len(addr_list), "건")
    for addr_dic in addr_list:
        print(addr_dic["name"], "\t", addr_dic["tel"])

def add_search() :
    fd_name = input("찾을 이름 : ")
    search_list = []
    for fd in addr_list:
        if fd["name"] == fd_name:
            search_list.append(fd["tel"])
    if len(search_list) <= 0:
        print("찾는 이름이 없습니다")
    else:
        for search_tel in search_list:
            print(search_tel)

def add_del():
    isdel = False
    search_tel = input("삭제 전화번호:")
    for i, addr_dic in enumerate(addr_list):
        if addr_dic["tel"] == search_tel:
            yn = input("정말 삭제하시겠습니까?(Y/N)")
```

```
            if yn.upper() == "Y":
                addr_list.pop(i)
                isdel = True

    if isdel == True:
        print("삭제되었습니다")
    elif isdel == False:
        print("검색 결과가 없습니다")

def add_update():
    isud = False
    search_tel = input("수정할 대상의 전화번호 : ")
    for addr_dic in addr_list:
        if addr_dic["tel"] == search_tel:
            ud_tel = input("수정 전화번호 : ")
            addr_dic["tel"] = ud_tel
            print(addr_dic)
            isud = True

    if isud == True:
        print("수정되었습니다")
    elif isud == False:
        print("검색 결과가 없습니다")


#addr_list = []

while(1):
    menu_print()

    cmd = input("명령어입력 : ")

    if cmd=="1" :
        add_input()

    elif cmd == "2":
        add_search()

    elif cmd=="3":
        add_search_all()

    elif cmd =="4":
        add_del()

    elif cmd == "5":
        add_update()

    elif cmd == "6":
        break

#-----------------------------------------------------
def run():
    while (1):
        menu_print()
```

```
        cmd = input("명령어입력 : ")

        if cmd == "1":
            add_input()
            addr_file_write()

        elif cmd == "2":
            add_search()

        elif cmd == "3":
            add_search_all()

        elif cmd == "4":
            add_del()
            addr_file_write()

        elif cmd == "5":
            add_update()
            addr_file_write()

        elif cmd == "6":
            break


#__ : 특수기능 내부 함수
#현재 파일에서 직접 돌렸다, 다른 모듈에서도 호출될 경우 어디에서 실행되는 것인지 알고자 할때 사용
if __name__ == "__main__":
    print("직접돌려보기")
    run()
```

## 주소록 함수 호출

```
from PycharmProjects.pythonProject.venv.test.lec06_주소록함수 import run
run()
```

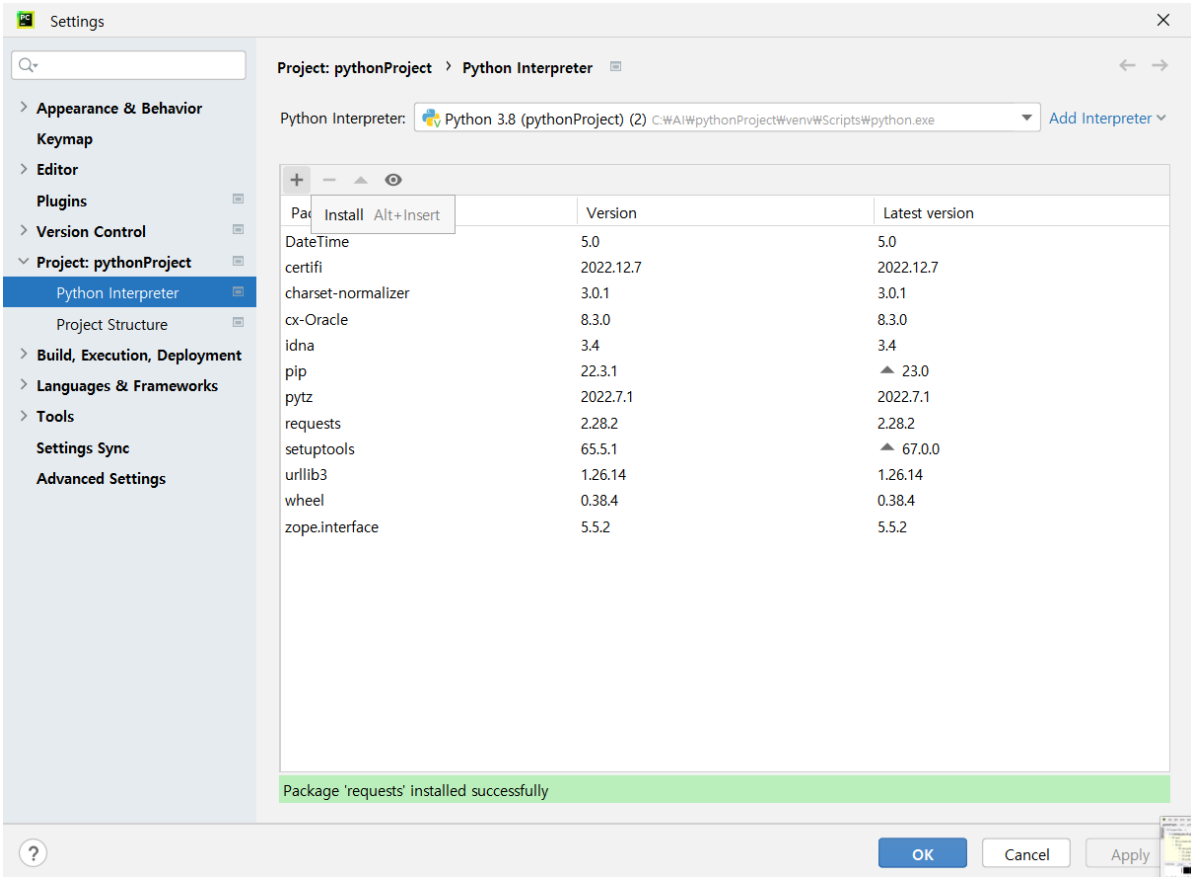
# 오라클 연동(DB)

whl : 파이썬 압축 확장자 / jar : 자바 / web : war / gip : window

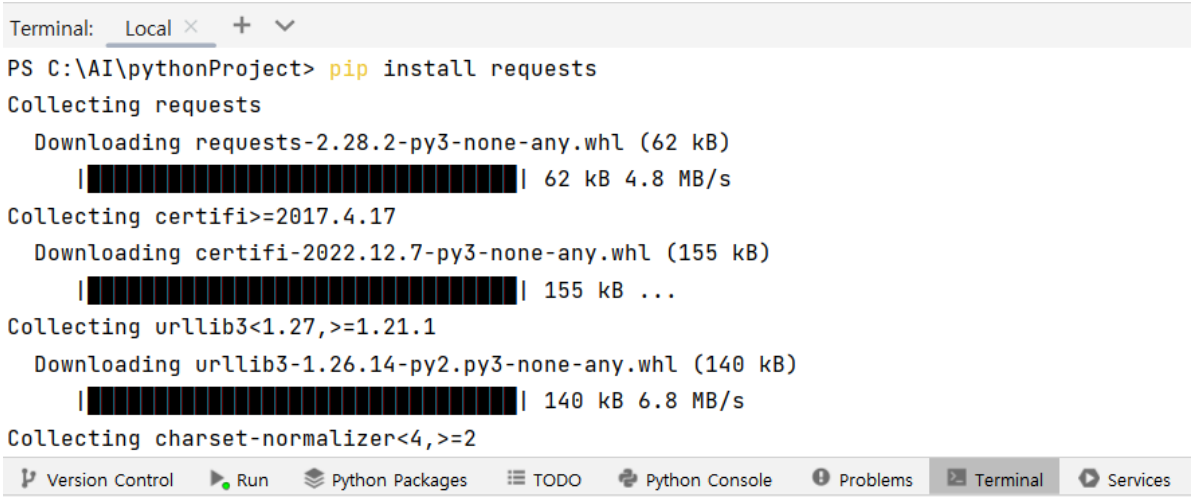
## setting

cx-Oracle , requests 설치

1. [File] - [Settings] - [Project:pythonProject] - [Python Interpreter] - [+]




2. 파이참 하단 [Terminal] : pip install 파일이름

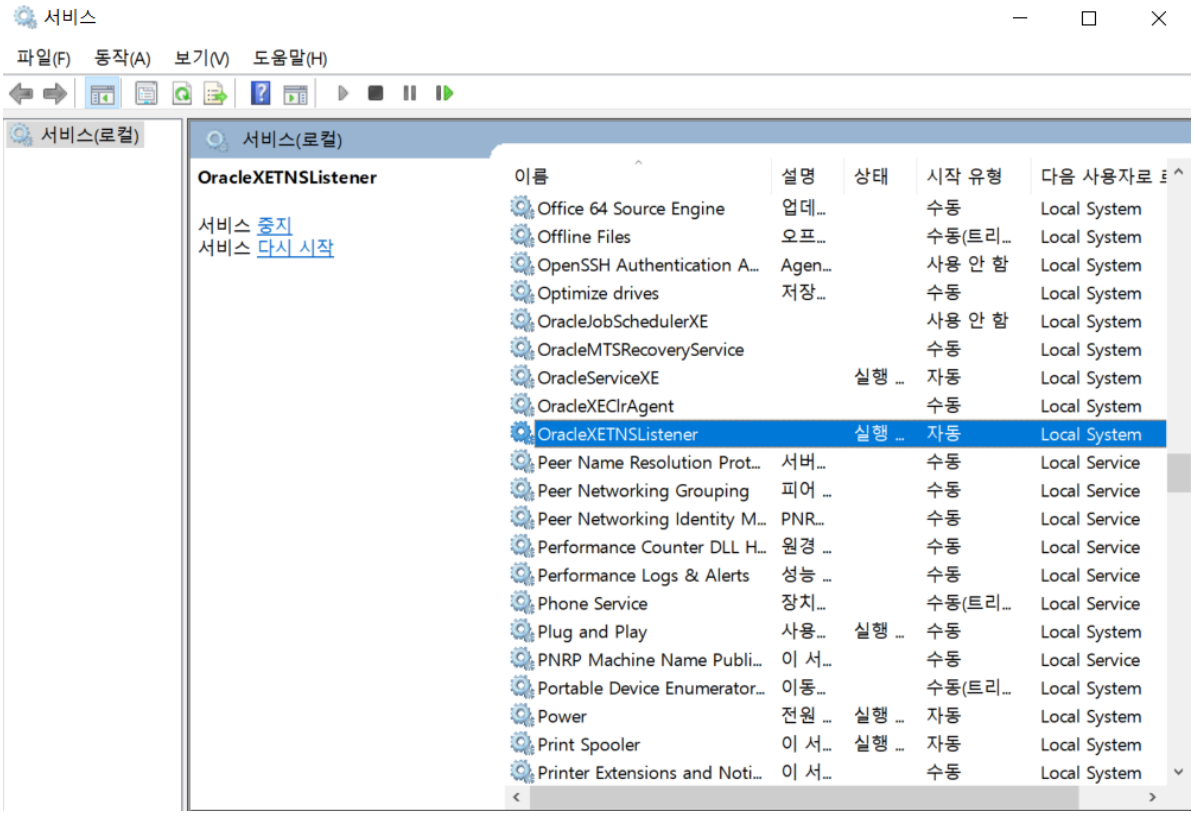



# 오라클 연동

```
# pip install cx_Oracle

import cx_Oracle

# with cx_Oracle.connect("ai","0000", "localhost:1521/XE") as conn:
#     if bool(conn):
#         print("연결성공")
#     else:
#         print("연결실패")
#
#     with conn.cursor() as cur:
#         cur.execute("select * from emp")
```

```
#         for row in cur:
#             print(list(row))
            # cur.close()
#conn.close()

conn= cx_Oracle.connect("ai","0000", "localhost:1521/XE")
    #오라클 연동(id,pw,"localhost:서비스번호/XE")
    #XE: 무료 버전/ ocl : 유료 버전
if bool(conn):
    print("연결성공")
else:
    print("연결실패")
```

- 최대 connection 15 -->만약 15가 다 차면 오라클 중지 시켜야 함
    - 탐색창 "서비스" -> Oracle
    - 끌 때 : OracleXETNSListener -> OracleServiceXE 순서로 끄기
    - 켤 때 : OracleServiceXE ->  OracleXETNSListener 순서로 켜기




1. SELECT

```
#-------------------------------------------------------------
# SELECT
#-------------------------------------------------------------
--1)
cur=conn.cursor() #커서 호출
cur.execute("select * from emp") #커서를 사용하여 실행
for row in cur: #한줄 씩 테이블 읽어오기
    print(list(row))
cur.close()
conn.close()

--2)
```

```
conn = cx_Oracle.connect("ai", "0000", "localhost:1521/XE")
sql = "select * from addr where seq = :1 or seq = :2"
cur = conn.cursor()
cur.execute(sql, [5, 1])
for row in cur:
    print( list(row)  )
cur.close()
conn.close()
```

2. UPDATE

```
--오라클
CREATE TABLE ADDR (
    SEQ number PRIMARY KEY ,
    NAME VARCHAR2(10),
    TEL VARCHAR2(15)
);
--------------------------------------------------------------------------
drop sequence addr_seq ;
create sequence addr_seq
start with 1
increment by 1
nocache;

insert into addr values(addr_seq.nextval, '홍길동', '000');
select * from addr;
commit;
```

```
#-------------------------------------------------
# update
# update addr set name='홍길순', tel='999' where seq=1
#-------------------------------------------------
conn = cx_Oracle.connect("ai", "0000", "localhost:1521/XE")
sql = "update addr set name=:1, tel=:2 where seq=:3"
cur = conn.cursor()
cur.execute(sql, ['홍길순', '999', 1])
conn.commit()
cur.close()
conn.close()
```

3. INSERT

```
#-------------------------------------------------------------
# INSERT : 1 row
# INSERT INTO ADDR VALUES(ADDR_SEQ.NEXTVAL,'홍길동', '000')
#-------------------------------------------------------------
#--1)
conn= cx_Oracle.connect("ai","0000", "localhost:1521/XE")
sql ="INSERT INTO ADDR VALUES(ADDR_SEQ.NEXTVAL, :1, :2)"
cur= conn.cursor()
```

```
cur.execute(sql, ['아무개','555'])
cur.execute(sql, ['함소영','2525'])
conn.commit()
cur.close()
conn.close()

#--2)
conn= cx_Oracle.connect("ai","0000", "localhost:1521/XE")
#sql ="INSERT INTO ADDR VALUES(ADDR_SEQ.NEXTVAL, :1, :2)" # :1,:2
sql ="INSERT INTO ADDR VALUES(ADDR_SEQ.NEXTVAL, :vnm, :vtel)" #--> 변수로 받아도 됨
cur= conn.cursor()
vnm="나변수"
vtel="999"
cur.execute(sql, [vnm,vtel])
conn.commit()
cur.close()
conn.close()

#--3)
conn= cx_Oracle.connect("ai","0000", "localhost:1521/XE")
sql ="INSERT INTO ADDR VALUES(ADDR_SEQ.NEXTVAL, :vnm, :vtel)"
cur= conn.cursor()
cur.execute(sql, {"vnm":"나변수2","vtel":"8989"}) #딕셔너리로 삽입 가능
conn.commit()
cur.close()
conn.close()


#-------------------------------------------------------------
# INSERT : multi rows
#-------------------------------------------------------------
#--1)
datas = [{"vnm":"나이름1","vtel":"111"},
         {"vnm":"나이름2","vtel":"222"},
         {"vnm":"나이름3","vtel":"333"}
         ]

conn= cx_Oracle.connect("ai","0000", "localhost:1521/XE")
sql ="INSERT INTO ADDR VALUES(ADDR_SEQ.NEXTVAL, :vnm, :vtel)"
cur= conn.cursor()
cur.executemany(sql, datas) #딕셔너리로 삽입 가능
conn.commit()
cur.close()
conn.close()

#--2)
datas = [["리스트1","6666"],
         ["리스트2","8888"],
         ["리스트3","9999"]
         ]

conn= cx_Oracle.connect("ai","0000", "localhost:1521/XE")
sql ="INSERT INTO ADDR VALUES(ADDR_SEQ.NEXTVAL, :1, :2)"
cur= conn.cursor()
cur.executemany(sql, datas)
```

```
conn.commit()
cur.close()
conn.close()
```

4. DELETE

```
#-------------------------------------------------------------
# DELETE
# delete from addr where name like '%나이름%' or name like '%리스트%';
#-------------------------------------------------------------
conn= cx_Oracle.connect("ai","0000", "localhost:1521/XE")
sql ="delete from addr where name like :1 or name like :2"
cur= conn.cursor()
cur.execute(sql, ['%나이름%','%리스트%'])
conn.commit()
cur.close()
conn.close()
```

**오라클과 파이썬 동시 작업 시 반드시 한쪽에서 COMMIT 완료 후 작업 진행하기