미니 프로젝트 1차 결과 보고서

-고객을 위한 임플란트 데이터 관리 프로젝트-

오스템 임플란트 취업연계형 과정 수강생 : 정재현

※목적 :

본 프로젝트는 임플란트의 종류에 따른 가격을 나열하고 고객에게 받은 접수 데이터를 활용하여 상품과 고객의 데이터를 조회, 삭제, 변경 기능을 넣어 활용할 수 있고, 상품과 고객의 데이터 리스트를 상호작용 하여 구매 리스트에 저장하고, 구매 리스트에서도 조회, 삭제, 변경, 저장 기능 프로그램을 제작하는 것이 목적.

※프로젝트 내용 : - 프로젝트의 구현 내용

C와 C++를 활용하여 고객, 상품, 구매 정보(데이터) 를 클래스로 만들고, STL 을 활용하여 각 클래스 마다 리스트들을 제공하는 데이터 관리 구조와 디자인 패턴을 최소로 고려한 책임중심원칙을 지키며, 고객의 데이터와 상품(임플란트)의 데이터를 활용하여 조회, 삭제, 변경 기능과 .txt 파일과 .csv 파일로 데이터를 저장하는 프로그램.

```
Client 파일 불러오기 완료
Product 파일 불러오기 완료
Shopping 파일 불러오기 완료

안녕하십니까? 임플란트 전용 치과 전문 센터 입니다.

아래에 해당하는 번호를 입력하여 원하시는 정보를 활용하십시오.
1. 고객 정보 관리, 2. 상품 정보 관리, 3.구매 정보 관리, 4.종료
번호를 입력하여 주세요 :
```

위의 내용처럼 폴더에 위치한 .csv.파일을 찾고 불러오고 난 뒤 정보관리 번호 나열

1. 고객의 데이터와 prime key를 저장하여 조회, 삭제, 변경 기능으로 고객의 데이터를 활용함.

가. 고객 데이터 입력

1) 고객 데이터 콘솔화면

```
아래에 해당하는 번호를 입력하여 원하시는 정보를 활용하십시오.
1. 고객 정보 관리, 2. 상품 정보 관리, 3.구매 정보 관리, 4.종료
번호를 입력하여 주세요 : 1

1.고객 정보 관리
1.입력, 2.조회, 3.삭제, 4.모두삭제, 5.변경
해당하는 번호를 입력해주세요 : 1

고객 정보 입력
고객 등록 ID : Client_004
고객 등록 이름 : 정다현
고객 등록 전화번호 : 010-9156-2739
고객 등록 이메일 : ekgus10@naver.com
Client Prime Key : Client_004

고객 정보 입력 완료
```

고객 정보 번호 중 1번을 입력하여 고객의 데이터(ID, 이름 등)를 입력하여 "고객 정보 입력 완료" 문 출력

2) C, Cpp 코드

ClientManager.cpp

```
void ClientManager::Client_Input(string _word,
    string _name, string _phone, string _email)
{
    clientList.push_back(new Client(_word, _name, _phone, _email));
    ClientManager::C_Count += 1;
    Client_PK(_word);
    cout << "\n고객 정보 입력 완료" << endl;
}
```

Main.cpp

```
switch (S_number)
{
case 1:
    cout << "\n고객 정보 입력" << endl;
    cout << "고객 등록 ID : "; cin >> c_word;
    cout << "고객 등록 이름 : "; cin >> c_name;
    cout << "고객 등록 전화번호 : "; cin >> c_phone;
    cout << "고객 등록 이메일 : "; cin >> c_email;
    cm.Client_Input(c_word, c_name, c_phone, c_email);
    break;
```

나. 고객 데이터 조회

1) 고객 조회 콘솔 화면

```
1.고객 정보 관리
1.입력, 2.조회, 3.삭제, 4.모두삭제, 5.변경
해당하는 번호를 입력해주세요 : 2
ClientCount : 4
++++++++++++++++++++++++++++++고객 정보 리스트++++++++++++++++++++++++++++++
--------------------------------------------------------------------------------
    고객ID |  고객 성함 |   고객 전화번호 |         고객 이메일 |    고객 등급
--------------------------------------------------------------------------------
 Client_001 |     정재현 |   010-2464-2739 |     0306jh@naver.com |         VIP
--------------------------------------------------------------------------------
 Client_002 |     정해영 |   010-5637-2739 |                not |        VVIP
--------------------------------------------------------------------------------
 Client_003 |     정민희 |   010-3421-2739 |                not |      Normal
--------------------------------------------------------------------------------
 Client_004 |     정다현 |   010-9156-2739 |    ekgus10@naver.com |      Normal
--------------------------------------------------------------------------------
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

고객의 정보를 전체로 조회하기 위해 고객 정보 관리 번호 중 2번을 눌러 정보리스트를 나열

2) C, Cpp 코드

ClientManager.cpp

```cpp
//고객 정보 리스트 공개 함수
void ClientManager::Display()
{
    cout << "ClientCount : " << C_Count << endl;
    cout << "++++++++++++++++++++++++++++++++고객 정보 리스트++++++++++++++++++++++++++++++++" << endl;
    cout << "--------------------------------------------------------------------------------" << endl;
    cout << setw(11) << "고객ID" << " | " << setw(10) << "고객 성함" << " | " << setw(15) << "고객 전화번호" << " | "
        << setw(20) << "고객 이메일" << " | " << setw(12) << "고객 등급" << endl;
    cout << "--------------------------------------------------------------------------------" << endl;
    for_each(clientList.begin(), clientList.end(), [](Client* c)
        {
            cout << setw(11) << c->getCWord() << " | " << setw(10) << c->getCName() << " | "
                << setw(15) << c->getCPhone() << " | " << setw(20) << c->getCEmail() << " | "

                //등급 확인과 구매한 총금액을 확인하는 함수
                << setw(12) << c->getCGrade() << endl;
            cout << "--------------------------------------------------------------------------------" << endl;
        });
    cout << "++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++" << endl << endl;
}
```

Main.cpp

```cpp
    case 2:
        cm.Display();
        break;
```

다. 고객 데이터 삭제

1) 고객 데이터 콘솔 화면

```
ID와 성함을 입력하면 정보가 삭제됩니다.
삭제할 고객 ID 입력 : Client_004
삭제할 고객 성함을 입력 : 정다현

고객 정보 삭제 완료


아래에 해당하는 번호를 입력하여 원하시는 정보를 활용하십시오.
1. 고객 정보 관리, 2. 상품 정보 관리, 3.구매 정보 관리, 4.종료
번호를 입력하여 주세요 : 1

1.고객 정보 관리
1.입력, 2.조회, 3.삭제, 4.모두삭제, 5.변경
해당하는 번호를 입력해주세요 : 2
ClientCount : 3
++++++++++++++++++++++++++++++++고객 정보 리스트++++++++++++++++++++++++++++++++
--------------------------------------------------------------------------------
     고객ID |  고객 성함 |   고객 전화번호 |          고객 이메일 |    고객 등급
--------------------------------------------------------------------------------
 Client_001 |     정재현 |   010-2464-2739 |     0306jh@naver.com |         VIP
--------------------------------------------------------------------------------
 Client_002 |     정해영 |   010-5637-2739 |                not |        VVIP
--------------------------------------------------------------------------------
 Client_003 |     정민희 |   010-3421-2739 |                not |      Normal
--------------------------------------------------------------------------------
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

위의 콘솔화면 처럼 고객의 ID와 이름을 입력하면 해당 열만 삭제할 수 있도록 기능 구현

2) C, Cpp코드

ClientManager.cpp

```cpp
//고객 정보 제거 함수
void ClientManager::Client_Remove(string _word, string _name)
{
    for (int i = 0; i < C_Count; i++)
    {
        if ((clientList.at(i)->getCWord().compare(_word) == 0) &&
            (clientList.at(i)->getCName().compare(_name) == 0))
        {
            clientList.erase(clientList.begin() + i);
            ClientManager::C_Count -= 1;
            cout << "\n고객 정보 삭제 완료" << endl;
        }
    }
}
```

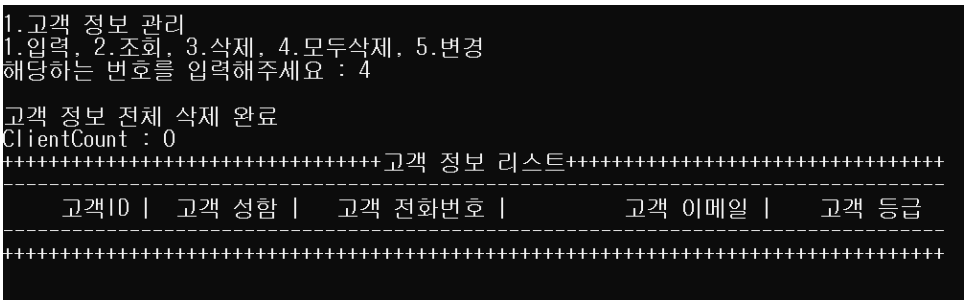
Vector<Client*>로 저장된 clientList의 ID와 이름을 비교하여 입력한 값에 맞으면 삭제하는 함수 구현

Main.cpp

```
case 3:
    cm.Display();
    cout << "\nID와 성함을 입력하면 정보가 삭제됩니다." << endl;
    cout << "삭제할 고객 ID 입력 : "; cin >> c_word;
    cout << "삭제할 고객 성함을 입력 : "; cin >> c_phone;
    cm.Client_Remove(c_word, c_phone); cout << endl;
    break;
```

라. 고객 데이터 전체삭제

1) 고객 데이터 전체삭제 콘솔화면



해당번호 4번을 입력하면 고객의 정보 리스트 들이 모두 삭제됨

2) C, Cpp

ClientManager.cpp

```
//고객 정보 전체 제거 함수
void ClientManager::RemoveAll()
{
    while (!clientList.empty())
    {
        clientList.erase(clientList.begin(), clientList.end());
    }
    ClientManager::C_Count = 0;
    cout << "\n고객 정보 전체 삭제 완료" << endl;
}
```

Main.cpp

```
case 4:
    cm.RemoveAll();
    cm.Display();
    break;
```

마. 고객 데이터 변경

1) 고객 데이터 변경 콘솔화면

```
1.고객 정보 관리
1.입력, 2.조회, 3.삭제, 4.모두삭제, 5.변경
해당하는 번호를 입력해주세요 : 5
ClientCount : 3
++++++++++++++++++++++++++++++++고객 정보 리스트++++++++++++++++++++++++++++++++
--------------------------------------------------------------------------------
     고객ID |  고객 성함 |   고객 전화번호 |          고객 이메일 |    고객 등급
--------------------------------------------------------------------------------
 Client_001 |     정재현 |   010-2464-2739 |     0306jh@naver.com |         VIP
--------------------------------------------------------------------------------
 Client_002 |     정해영 |   010-5637-2739 |                not |        VVIP
--------------------------------------------------------------------------------
 Client_003 |     정민희 |   010-3421-2739 |                not |      Normal
--------------------------------------------------------------------------------
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++


ID와 성함을 입력하면 정보가 변경됩니다.
변경될 고객 ID를 입력 : Client_001
변경될 고객 성함을 입력 : 정재현

변경할 고객 ID : Client_001
변경할 고객 성함 : 정재현
변경할 고객 전화번호 : 010-9154-2739
변경할 고객 이메일 : defija@gmail.com
고객 정보 변경 완료!!
```

```
1.고객 정보 관리
1.입력, 2.조회, 3.삭제, 4.모두삭제, 5.변경
해당하는 번호를 입력해주세요 : 2
ClientCount : 3
++++++++++++++++++++++++++++++++고객 정보 리스트++++++++++++++++++++++++++++++++
--------------------------------------------------------------------------------
     고객ID |  고객 성함 |   고객 전화번호 |          고객 이메일 |    고객 등급
--------------------------------------------------------------------------------
 Client_001 |     정재현 |   010-9154-2739 |     defija@gmail.com |         VIP
--------------------------------------------------------------------------------
 Client_002 |     정해영 |   010-5637-2739 |                not |        VVIP
--------------------------------------------------------------------------------
 Client_003 |     정민희 |   010-3421-2739 |                not |      Normal
--------------------------------------------------------------------------------
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

고객변경 번호 입력 후 해당 ID와 이름을 입력후 변경하여 조회한 화면

2) C, Cpp

ClientManager.cpp

```cpp
//고객 정보 변경 함수
void ClientManager::Client_Change(string _word, string _name)
{
    for (int i = 0; i < ClientManager::C_Count; i++)
    {
        if ((clientList.at(i)->getCWord().compare(_word) == 0) &&
            (clientList.at(i)->getCName().compare(_name) == 0))
        {
            cout << "\n변경할 고객 ID : "; cin >> ClientManager::CM_Word;
            clientList.at(i)->setCWord(ClientManager::CM_Word);
            cout << "변경할 고객 성함 : "; cin >> ClientManager::CM_Name;
            clientList.at(i)->setCName(ClientManager::CM_Name);
            cout << "변경할 고객 전화번호 : "; cin >> ClientManager::CM_Phone;
            clientList.at(i)->setCPhone(ClientManager::CM_Phone);
            cout << "변경할 고객 이메일 : "; cin >> ClientManager::CM_Email;
            clientList.at(i)->setCEmail(ClientManager::CM_Email);
            cout << "고객 정보 변경 완료!!" << endl;
        }
    }
}
```

Main.cpp

```
case 5:
    cm.Display();
    cout << "\nID와 성함을 입력하면 정보가 변경됩니다." << endl;
    cout << "변경될 고객 ID를 입력 : "; cin >> c_word;
    cout << "변경될 고객 성함을 입력 : "; cin >> c_name;
    cm.Client_Change(c_word, c_name); cout << endl;
    break;
```

2. 상품(임플란트)의 데이터를 prime key 를 저장하여 조회, 삭제, 변경 기능으로 고객의 데이터를 활용함.

가. 상품 데이터 입력

1) 콘솔화면

```
아래에 해당하는 번호를 입력하여 원하시는 정보를 활용하십시오.
1. 고객 정보 관리, 2. 상품 정보 관리, 3.구매 정보 관리, 4.종료
번호를 입력하여 주세요 : 2

2.상품 정보 관리
1.입력, 2.조회, 3.삭제, 4.모두삭제, 5.변경
해당하는 번호를 입력해주세요 : 1

상품 정보 입력
상품 등록 ID : CARE_ELSE
상품 등록 이름 : 샘플
상품 등록 가격 : 10000
Product Prime Key : CARE_ELSE

상품 정보 등록 완료!
```

상품 정보 관리에서 해당번호르 입력 후 상품의 ID, 이름, 가격 데이터를 입력 후 "상품 정보 등록 완료" 문 확인

2) C, Cpp

ProductManager.cpp

```
//상품 데이터 입력 함수
void ProductManager::Product_Input(string _id, string _name, int _price)
{
    productList.push_back(new Product(_id, _name, _price));
    P_Count += 1;
    Product_PK(_id);
    cout << "\n상품 정보 등록 완료!" << endl << endl;
}
```

Main.cpp

```
case 1:

    cout << "\n상품 정보 입력" << endl;
    cout << "상품 등록 ID : "; cin >> p_id;
    cout << "상품 등록 이름 : "; cin >> p_name;
product_1:;
    cout << "상품 등록 가격 : "; cin >> p_price;
    if (!cin)//정수형 인자 콘솔 입력부분 경계 검사
    {
        cout << "\n등록 가격에 정수형 숫자를 입력하지 않았습니다." << endl;
        cin.clear();
        cin.ignore(INT_MAX, '\n');
        goto product_1;
    }
    pm.Product_Input(p_id, p_name, p_price);
    break;
```

나. 상품 데이터 조회

1) 콘솔화면

```
2.상품 정보 관리
1.입력, 2.조회, 3.삭제, 4.모두삭제, 5.변경
해당하는 번호를 입력해주세요 : 2
ProductCount : 7
++++++++++++상품 정보 리스트++++++++++++
----------------------------------------
   상품 ID |   상품 이름 |    상품 가격
----------------------------------------
  CARE_001 |      오스템 |     100000원
----------------------------------------
  CARE_002 |        니어 |     200000원
----------------------------------------
  CARE_003 |      원데이 |      50000원
----------------------------------------
  CARE_004 |      상악동 |      70000원
----------------------------------------
  CARE_005 |      무치악 |     150000원
----------------------------------------
  CARE_006 |      뼈이식 |      80000원
----------------------------------------
 CARE_ELSE |        샘플 |      10000원
----------------------------------------
++++++++++++++++++++++++++++++++++++++++
```

상품 정보 관리에서 해당 버튼을 입력하면 새로운 항목이 추가 되는걸 볼 수 있음.

2) C, Cpp

ProductManager.cpp

```cpp
//상품 리스트 공개 함수
void ProductManager::Display()
{
    cout << "ProductCount : " << P_Count << endl;
    cout << "++++++++++++상품 정보 리스트++++++++++++" << endl;
    cout << "----------------------------------------" << endl;
    cout << setw(11) << "상품 ID" << " | " << setw(11) << "상품 이름" << " | " << setw(12) << "상품 가격" << endl;
    cout << "----------------------------------------" << endl;
    for_each(productList.begin(), productList.end(),
        [](Product* p) {
            cout << setw(11) << p->getPId() << " | " << setw(11) << p->getPName() << " | "
                << setw(10) << p->getPPrice() << "원" << endl;
            cout << "----------------------------------------" << endl;
        });
    cout << "++++++++++++++++++++++++++++++++++++++++" << endl << endl;
}
```

Main.cpp

```cpp
case 2:
    pm.Display();
    break;
```

다. 상품 데이터 삭제

1) 콘솔화면

```
ID를 입력하면 정보가 삭제됩니다.
삭제할 상품 ID 입력 : CARE_ELSE

상품 정보 삭제 완료!


아래에 해당하는 번호를 입력하여 원하시는 정보를 활용하십시오.
1. 고객 정보 관리, 2. 상품 정보 관리, 3.구매 정보 관리, 4.종료
번호를 입력하여 주세요 : 2

2.상품 정보 관리
1.입력, 2.조회, 3.삭제, 4.모두삭제, 5.변경
해당하는 번호를 입력해주세요 : 2
ProductCount : 6
++++++++++++상품 정보 리스트++++++++++++
----------------------------------------
    상품 ID |   상품 이름 |    상품 가격
----------------------------------------
   CARE_001 |      오스템 |     100000원
----------------------------------------
   CARE_002 |        니어 |     200000원
----------------------------------------
   CARE_003 |      원데이 |      50000원
----------------------------------------
   CARE_004 |      상악동 |      70000원
----------------------------------------
   CARE_005 |      무치악 |     150000원
----------------------------------------
   CARE_006 |      뼈이식 |      80000원
----------------------------------------
++++++++++++++++++++++++++++++++++++++++
```

해당 상품 ID입력 후 상품정보 조회시 해당 리스트가 삭제 되는 것을 확인

2) C, Cpp

ProductManager.cpp

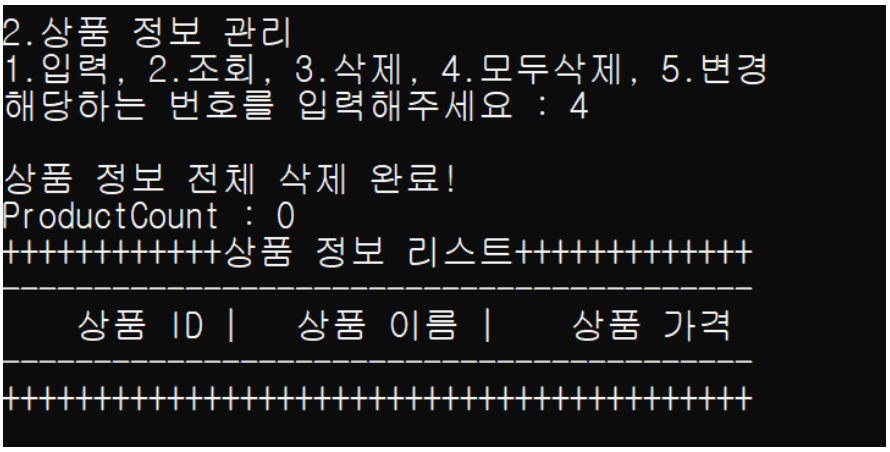
```cpp
//상품 정보 삭제 함수
void ProductManager::Product_Remove(string _id)
{
    for (int i = 0; i < P_Count; i++)
    {
        if (productList.at(i)->getPId().compare(_id) == 0)
        {
            productList.erase(productList.begin() + i);
            P_Count -= 1;
            cout << "\n상품 정보 삭제 완료!" << endl;
        }
    }
}
```

Main.cpp

```cpp
case 3:
    cm.Display();
    cout << "\nID와 성함을 입력하면 정보가 삭제됩니다." << endl;
    cout << "삭제할 고객 ID 입력 : "; cin >> c_word;
    cout << "삭제할 고객 성함을 입력 : "; cin >> c_phone;
    cm.Client_Remove(c_word, c_phone); cout << endl;
    break;
```

라. 상품 데이터 전체삭제

1) 콘솔화면



해당 번호를 입력하면 상품 ID 전부 삭제

2) C, Cpp

ProductManager.cpp

```cpp
//상품 정보 전체 제거 함수
void ProductManager::RemoveAll()
{
    while (!productList.empty())
    {
        productList.erase(productList.begin(), productList.end());
    }
    P_Count = 0;
    cout << "\n상품 정보 전체 삭제 완료!" << endl;
}
```

Main.cpp

```
case 4:
    cm.RemoveAll();
    cm.Display();
    break;
```

마. 상품 데이터 변경

1) 콘솔화면

```
2.상품 정보 관리
1.입력, 2.조회, 3.삭제, 4.모두삭제, 5.변경
해당하는 번호를 입력해주세요 : 2
ProductCount : 6
++++++++++++상품 정보 리스트++++++++++++
----------------------------------------
    상품 ID |   상품 이름 |    상품 가격
----------------------------------------
   CARE_001 |     오스템P |     150000원
----------------------------------------
   CARE_002 |        니어 |     200000원
----------------------------------------
   CARE_003 |      원데이 |      50000원
----------------------------------------
   CARE_004 |      상악동 |      70000원
----------------------------------------
   CARE_005 |      무치악 |     150000원
----------------------------------------
   CARE_006 |      뼈이식 |      80000원
----------------------------------------
++++++++++++++++++++++++++++++++++++++++
```

```
ID를 입력하면 정보가 변경됩니다.
변경될 상품 ID를 입력 : CARE_001

변경할 상품 ID : CARE_001
변경할 상품 이름 : 오스템P
변경할 상품 가격 : 150000
상품 정보 변경 완료!
```

해당하는 변경 버튼을 입력후 상품의 ID를 입력하면 변경할 데이터 나열, 변경 이후 조회를 하면 상품정보 변경

2) C, Cpp

ProductManager.cpp

```
//상품 정보 변경 함수
void ProductManager::Product_Change(string _id)
{
    for (int i = 0; i < P_Count; i++)
    {
        if (productList.at(i)->getPId().compare(_id) == 0)
        {
            cout << "\n변경할 상품 ID : "; cin >> ProductManager::PM_ID;
            productList.at(i)->setPId(ProductManager::PM_ID);
            cout << "변경할 상품 이름 : "; cin >> ProductManager::PM_Name;
            productList.at(i)->setPName(ProductManager::PM_Name);
        product_price:;
            cout << "변경할 상품 가격 : "; cin >> ProductManager::PM_Price;
            if (!cin)//정수형 인자 콘솔 입력부분 경계 검사
            {
                cout << "\n등록 가격에 정수형 숫자를 입력하지 않았습니다." << endl;
                cin.clear();
                cin.ignore(INT_MAX, '\n');
                goto product_price;
            }
            productList.at(i)->setPPrice(ProductManager::PM_Price);
            cout << "상품 정보 변경 완료!" << endl;
        }
    }
}
```

Main.cpp

```
case 5:
    cm.Display();
    cout << "\nID와 성함을 입력하면 정보가 변경됩니다." << endl;
    cout << "변경될 고객 ID를 입력 : "; cin >> c_word;
    cout << "변경될 고객 성함을 입력 : "; cin >> c_name;
    cm.Client_Change(c_word, c_name); cout << endl;
    break;
```

3. 고객과 상품의 데이터와 prime key를 저장하여 구매 리스트를 만들고 고객의 prime key와 상품의 prime key를 검색하여 목적에 맞는 리스트만 출력.

가. 구매 리스트 데이터 입력

1) 콘솔화면

구매 리스트 정보에서 주문 입력후 해당하는 ID를 입력후 날짜와 수량 데이터를 입력하면 구매 정보 추가 완료

2) C,Cpp화면

```
//구매 정보 데이터 추가 함수
void ShoppingManager::Shopping_Input(ClientManager& C_ref, ProductManager& P_ref,
    int _num, string _clpk, string _prpk, int _date, int _quantatiy)
{
    //고객 등급을 string변수로 내부에서 따로 저장하는 이유는 compare함수에 "VVIP"로 작성하면 const char*로 NULL문자까지 포함시켜 버린다.
    string VVIP = "VVIP";
    string VIP = "VIP";
    string Normal = "Normal";


    for (int i = 0; i < C_ref.Count(); i++)
    {
        if (C_ref.clientList.at(i)->getCWord().compare(_clpk) == 0) //클라이언트 PK와 입력한 클라이언트 PK값 비교 참이면 바디 계산
        {
            string C_Word = C_ref.clientList.at(i)->getCWord();
            auto it1 = find_if(C_ref.clientList.begin(), C_ref.clientList.end(),
                [=](Client* c) {return *c == C_Word; });
            if (it1 != C_ref.clientList.end())
            {
                Client* c = *it1;
                cout << "ADD Client Prime Key : " << c->getCWord() << endl;
                for (int j = 0; j < P_ref.Count(); j++)
                {
                    if (P_ref.productList.at(j)->getPId().compare(_prpk) == 0) //프로덕트 PK와 입력한 프로덕트 PK값 비교 참이면 바디 계산
                    {
                        string P_Id = P_ref.productList.at(j)->getPId();
                        auto it2 = find_if(P_ref.productList.begin(),
                            P_ref.productList.end(),
                            [=](Product* p) {return *p == P_Id; });
                        if (it2 != P_ref.productList.end())
                        {
                            Product* p = *it2;
                            cout << "ADD Prodcut Prime Key : " << p->getPId() << endl;
                            double price;
```

```
                    //등급별 가격 할인
                    if (C_ref.clientList.at(i)->getCGrade().compare(VVIP) == 0)
                    {
                        price = (P_ref.productList.at(j)->getPPrice() * _quantatiy) * 0.90; //VVIP면 10퍼센트 할인
                        shoppingList.push_back(new Shopping(_num,
                            _clpk, _prpk, _date, _quantatiy, price)); // 위의 PK값 일치가 충족 된다면 쇼핑 리스트에 데이터 추가
                        Snumber += 1;
                        S_Count += 1;
                        cout << "\n구매 정보 추가 완료" << endl;
                    }
                    if (C_ref.clientList.at(i)->getCGrade().compare(VIP) == 0)
                    {
                        price = (P_ref.productList.at(j)->getPPrice() * _quantatiy) * 0.95; //VIP면 5퍼센트 할인
                        shoppingList.push_back(new Shopping(_num,
                            _clpk, _prpk, _date, _quantatiy, price)); // 위의 PK값 일치가 충족 된다면 쇼핑 리스트에 데이터 추가
                        Snumber += 1;
                        S_Count += 1;
                        cout << "\n구매 정보 추가 완료" << endl;
                    }
                    if (C_ref.clientList.at(i)->getCGrade().compare(Normal) == 0)
                    {
                        price = P_ref.productList.at(j)->getPPrice() * _quantatiy; //Normal이면 아무것도 없음
                        shoppingList.push_back(new Shopping(_num,
                            _clpk, _prpk, _date, _quantatiy, price)); // 위의 PK값 일치가 충족 된다면 쇼핑 리스트에 데이터 추가
                        Snumber += 1;
                        S_Count += 1;
                        cout << "\n구매 정보 추가 완료" << endl;
                    }
                }
            }
        }
    }
}
```

구매 입력 부분 코드는 고객의 PK와 상품의 PK를 입력하여 해당 리스트들의 PK와 입력한 PK의 값이 일치하는지를 보고 추가하는 방식으로 코드를 작성

나. 구매 리스트 데이터 조회

가) 콘솔 화면

```
3.구매 정보 관리
1.주문, 2.조회, 3.삭제, 4.전체삭제, 5.변경
번호를 입력해주세요 : 2
++++++++++++++++++++++++++++++구매 정보 리스트++++++++++++++++++++++++++++++
-------------------------------------------------------------------------------
 번호 |    고객 ID |     상품 ID |   구매 날짜 |  수량 |           총 금액
-------------------------------------------------------------------------------
    0 | Client_001 |    CARE_001 |           0 |     0 |                0원
-------------------------------------------------------------------------------
    1 | Client_002 |    CARE_002 |      220905 |     5 |          1000000원
-------------------------------------------------------------------------------
    2 | Client_001 |    CARE_001 |      220905 |     5 |           500000원
-------------------------------------------------------------------------------
    3 | Client_001 |    CARE_002 |      220905 |     1 |           190000원
-------------------------------------------------------------------------------
    4 | Client_002 |    CARE_001 |      220905 |     1 |            90000원
-------------------------------------------------------------------------------
    5 | Client_003 |    CARE_003 |      220906 |     2 |           100000원
-------------------------------------------------------------------------------
    6 | Client_003 |    CARE_001 |      220906 |     2 |           200000원
-------------------------------------------------------------------------------
    7 | Client_003 |    CARE_003 |      220906 |     1 |            50000원
-------------------------------------------------------------------------------
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

조회를 누르면 추가한 구매정보를 볼 수 있음

```
구매 정보 리스트에서 검색할 키워드가 있으십니까?
1. ClientID, 2. ProductID, 3. 날짜별 구매 정보, 4.종료 : 1

ClientID의 정보중 어떤 정보를 원하십니까?
1. ClientID 정보, 2. 해당 ClientID 구매 목록 : 1
ClientID 입력 : Client_001

해당 ClientID(Client_001)의 정보는 :
고객 성함 : 정재현,
고객 등급 : VIP,
고객 전화번호 : 010-2464-2739,
고객 이메일 : 0306jh@naver.com


구매 정보 리스트에서 검색할 키워드가 있으십니까?
1. ClientID, 2. ProductID, 3. 날짜별 구매 정보, 4.종료 : 1

ClientID의 정보중 어떤 정보를 원하십니까?
1. ClientID 정보, 2. 해당 ClientID 구매 목록 : 2
ClientID 입력 :
Client_001

++++++++++++++++++++++Client_001의 구매 정보 리스트++++++++++++++++++++++++
-------------------------------------------------------------------------------
 번호 |     고객 ID |     상품 ID |   구매 날짜 |  수량 |              총 금액
-------------------------------------------------------------------------------
    0 | Client_001 |    CARE_001 |           0 |     0 |                    0
-------------------------------------------------------------------------------
    2 | Client_001 |    CARE_001 |      220905 |     5 |               500000
-------------------------------------------------------------------------------
    3 | Client_001 |    CARE_002 |      220905 |     1 |               190000
-------------------------------------------------------------------------------
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

조회 함수안에 ClientID 정보와 해당 ClientID의 구매 목록을 볼 수 있음

나) C, Cpp

```cpp
//구매 정보 리스트 공개 함수
void ShoppingManager::Shopping_Display(ClientManager& _cm, ProductManager& _pm)
{
    int num = 0;
    string PKCL, PKPR;
    int sm_date, flash = 1;
    char find_c;

    cout << "++++++++++++++++++++++++++++++구매 정보 리스트++++++++++++++++++++++++++++++" << endl;
    cout << "-------------------------------------------------------------------------------" << endl;
    cout << setw(5) << "번호" << " | " << setw(10) << "고객 ID" << " | " << setw(11) << "상품 ID" << " | " << setw(11) << "구매 날짜" << " | "
        << setw(5) << "수량" << " | " << setw(18) << "총 금액" << endl;
    cout << "-------------------------------------------------------------------------------" << endl;
    for_each(shoppingList.begin(), shoppingList.end(), [](Shopping* s)
        {
            cout << setw(5) << s->getSNumber() << " | " << setw(10) << s->getSPKClient() << " | "
                << setw(11) << s->getSPKProduct() << " | " << setw(11) << s->getSDate() <<
                " | " << setw(5) << s->getSQuan() << " | " << setw(16) << s->getSAllprice() << "원" << endl;
            cout << "-------------------------------------------------------------------------------" << endl;
        });
    cout << "+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++" << endl << endl;
```

조회 코드에서는 구매 정보리스트를 나열하는 코드 작성

```cpp
//구매리스트 조회후 client, product, 날짜별 정보 검색
while (num != 4)
{
back_search:;
    cout << "\n구매 정보 리스트에서 검색할 키워드가 있으십니까?" << endl;
    cout << "1. ClientID, 2. ProductID, 3. 날짜별 구매 정보, 4.종료 : "; cin >> num;

    if (!cin)
    {
        cout << "\n정수형 숫자를 입력해주시기 바랍니다." << endl;
        cin.clear();
        cin.ignore(INT_MAX, '\n');
        goto back_search;
    }
    switch (num)
    {
        //ClientID의 정보의 종류를 물어보는 case문
    case 1:
        cout << "\nClientID의 정보중 어떤 정보를 원하십니까?" << endl;
        cout << "1. ClientID 정보, 2. 해당 ClientID 구매 목록 : ";

        scanf_s("%c", &find_c);

        do
        {
            find_c = _getch();
            if (find_c >= '1' && find_c <= '2')
            {
                printf("%c", find_c);
                break;
            }
        } while (true);
        //ClientID 정보 공개 함수
```

```cpp
//ClientID 정보 공개 함수
if (find_c == '1')
{
    cout << "\nClientID 입력 : "; cin >> pk_c1;
    cout << "\n해당 ClientID(" << pk_c1 << ")의 정보는 :";
    PKCL = pk_c1;
    auto it = find_if(_cm.clientList.begin(), _cm.clientList.end(),
        [=](Client* c) {return *c == PKCL; });
    if (it != _cm.clientList.end())
    {
        Client* c = *it;
        cout << "\n고객 성함 : " << c->getCName() <<
            ", \n고객 등급 : " << c->getCGrade() <<
            ", \n고객 전화번호 : " << c->getCPhone() <<
            ", \n고객 이메일 : " << c->getCEmail() << endl << endl;
    }
}
//해당 ClientID 구매 정보 리스트
else if (find_c == '2')
{
    cout << "\nClientID 입력 : "; cin >> pk_c1;
    PKCL = pk_c1;
    cout << endl;
    auto it = find_if(_cm.clientList.begin(), _cm.clientList.end(),
        [=](Client* c) {return *c == PKCL; });
    if (it != _cm.clientList.end())
    {
        Client* c = *it;
        cout << "++++++++++++++++++++++++" << pk_c1 <<"의 구매 정보 리스트++++++++++++++++++++++++" << endl;
        cout << "-------------------------------------------------------------------------------" << endl;
        cout << setw(5) << "번호" << " | " << setw(10) << "고객 ID" << " | " << setw(11) << "상품 ID" << " | " << setw(11) << "구매 날짜" << " | "
            << setw(5) << "수량" << " | " << setw(18) << "총 금액" << endl;
        for (int i = 0; i < S_Count; i++)
        {
            if (c->getCWord().compare(shoppingList.at(i)->getSPKClient()) == 0)
            {
                cout << "-------------------------------------------------------------------------------" << endl;
                cout << setw(5) << shoppingList.at(i)->getSNumber() << " | " << setw(10);
                textcolor(LIGHTRED, BLACK);
                cout << shoppingList.at(i)->getSPKClient();
                textcolor(WHITE, BLACK);
                cout << " | " << setw(11) << shoppingList.at(i)->getSPKProduct() << " | " << setw(11) << shoppingList.at(i)->getSDate() << " | "
                    << setw(5) << shoppingList.at(i)->getSQuan() << " | " << setw(18) << shoppingList.at(i)->getSAllprice() << endl;
            }
        }
        cout << "-------------------------------------------------------------------------------" << endl;
        cout << "+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++" << endl << endl;
    }
}
```

이윽고 조회 화면 아래에서 구매 정보 리스트에서 PK값을 입력하면 해당 리스트를 비교해 찾아가면서 입력한 PK와 맞으면 출력하는 기능을 구현

ClientID조회 부분에서 정수 값을 입력해야 하는데 문자를 입력하면 루프백을 돌게하는 함수도 구현

```
    //해당없는 정수형 숫자는 조회 질문으로 돌아감
    else
    {
        cout << "\n조회 질문으로 돌아가겠습니다." << endl;
        break;
    }
    break;
    //ClientID의 정보의 종류를 물어보는 case문

    //ProdcutID의 정보의 종류를 물어보는 case문
case 2:
    cout << "\nProductID의 정보중 어떤 정보를 원하십니까?" << endl;
    cout << "1. ProductID 정보, 2. 해당 ProductID 구매 목록 : ";
    scanf_s("%c", &find_c);

    do
    {
        find_c = _getch();
        if (find_c >= '1' && find_c <= '2')
        {
            printf("%c", find_c);
            break;
        }
    } while (true);
    //해당 ProdcutID의 정보 검색
    if (find_c == '1')
    {
        cout << "\nProductID 입력 :"; cin >> pk_pr;
        cout << "\n해당 ProductID(" << pk_pr << ")의 정보는 :";
        PKPR = pk_pr;
        auto it = find_if(_pm.productList.begin(), _pm.productList.end(),
            [=](Product* p) {return *p == PKPR; });
        if (it != _pm.productList.end())
        {
            Product* p = *it;
            cout << "\n상품 이름 : " << p->getPName() <<
                ", \n상품 가격 : " << p->getPPrice() << endl << endl;
        }
    }
```

```
    //해당 ProductID의 구매정보 리스트 나열
    else if (find_c == '2')
    {
        cout << "\nProductID 입력 : "; cin >> pk_pr;
        PKPR = pk_pr;
        cout << endl;
        auto it = find_if(_pm.productList.begin(), _pm.productList.end(),
            [=](Product* p) {return *p == PKPR; });
        if (it != _pm.productList.end())
        {
            Product* p = *it;
            cout << "++++++++++++++++++++++++" << pk_pr << "의 구매 정보 리스트++++++++++++++++++++++++" << endl;
            cout << "-------------------------------------------------------------------------------" << endl;
            cout << setw(5) << "번호" << " | " << setw(10) << "고객 ID" << " | " << setw(11) << "상품 ID" << " | " << setw(11) << "구매 날짜" << " | "
                << setw(5) << "수량" << " | " << setw(18) << "총 금액" << endl;

            for (int i = 0; i < S_Count; i++)
            {
                if (p->getPId().compare(shoppingList.at(i)->getSPKProduct()) == 0)
                {
                    cout << "-------------------------------------------------------------------------------" << endl;
                    cout << setw(5) << shoppingList.at(i)->getSNumber() << " | " << setw(10) << shoppingList.at(i)->getSPKClient() << " | ";
                    textcolor(LIGHTRED, BLACK);
                        cout << setw(11) << shoppingList.at(i)->getSPKProduct();
                        textcolor(WHITE, BLACK);
                        cout << " | " << setw(11) << shoppingList.at(i)->getSDate() << " | "
                        << setw(5) << shoppingList.at(i)->getSQuan() << " | " << setw(18) << shoppingList.at(i)->getSAllprice() << endl;
                }

            }
            cout << "-------------------------------------------------------------------------------" << endl;
            cout << "++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++" << endl << endl;
        }
    }

    else
    {
        cout << "조회 질문으로 돌아가겠습니다." << endl;
        break;
    }
    break;
    //ProdcutID의 정보의 종류를 물어보는 case문
```

```
    //shopping list의 날짜별 구매정보 리스트 탐/검색
case 3:
back_date:;
    cout << "\nShopping Date입력 : "; cin >> date;
    if (!cin)
    {
        cout << "현재 입력한 정보가 정수형 데이터가 아닙니다." << endl;
        cin.ignore(INT_MAX, '\n');
        goto back_date;
    }
    sm_date = date;
    cout << endl;
    if (flash == 1)
    {
        auto it = find_if(shoppingList.begin(), shoppingList.end(), [=](Shopping* s) {return *s == sm_date; });
        if (it != shoppingList.end())
        {
            Shopping* s = *it;
            cout << "++++++++++++++++++++++++++" << date << " 별 구매 리스트++++++++++++++++++++++++++" << endl;
            cout << "-------------------------------------------------------------------------------" << endl;
            cout << setw(5) << "번호" << " | " << setw(10) << "고객 ID" << " | " << setw(11) << "상품 ID" << " | " << setw(11) << "구매 날짜" << " | "
                << setw(5) << "수량" << " | " << setw(18) << "총 금액" << endl;
            for (int i = 0; i < S_Count; i++)
            {
                if (s->getSDate() == shoppingList.at(i)->getSDate())
                {
                    cout << "-------------------------------------------------------------------------------" << endl;
                    cout << setw(5) << shoppingList.at(i)->getSNumber() << " | " << setw(10) << shoppingList.at(i)->getSPKClient() << " | "
                        << setw(11) << shoppingList.at(i)->getSPKProduct() << " | ";
                    textcolor(LIGHTRED, BLACK);
                    cout << setw(11) << shoppingList.at(i)->getSDate();
                    textcolor(WHITE, BLACK);
                    cout << " | "  << setw(5) << shoppingList.at(i)->getSQuan();
                        cout << " | " << setw(18) << shoppingList.at(i)->getSAllprice() << endl;
                }

            }
            cout << "-------------------------------------------------------------------------------" << endl;
            cout << "++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++" << endl << endl;
        }
    }
    break;
    //shopping list의 날짜별 구매정보 리스트 탐/검색


case 4:
    cout << "검색 프로그램 종료" << endl;
    break;
default:
    cout << "해당하는 검색 키워드가 없습니다." << endl;
    break;
}
}
```

ClientID뿐만 아니하 ProductID, 날짜 정보에 따라 해당하는 정보를 불러올 수 있음.

조회 창에서 나가기 위해서는 구매 정보 리스트에서 검색할 키워드 질문에 4번을 입력하여 나가기만 하면 됨.

다. 구매 리스트 데이터 삭제

1) 콘솔화면

```
삭제할 구매 번호를 입력해주세요 : 7

구매 정보 삭제 완료!

아래에 해당하는 번호를 입력하여 원하시는 정보를 활용하십시오.
1. 고객 정보 관리, 2. 상품 정보 관리, 3.구매 정보 관리, 4.종료
번호를 입력하여 주세요 : 3

3.구매 정보 관리
1.주문, 2.조회, 3.삭제, 4.전체삭제, 5.변경
번호를 입력해주세요 : 2
++++++++++++++++++++++++++++구매 정보 리스트++++++++++++++++++++++++++++
-------------------------------------------------------------------------
 번호 |    고객 ID |     상품 ID |   구매 날짜 |  수량 |           총 금액
-------------------------------------------------------------------------
    0 | Client_001 |    CARE_001 |          0 |     0 |              0원
-------------------------------------------------------------------------
    1 | Client_002 |    CARE_002 |     220905 |     5 |        1000000원
-------------------------------------------------------------------------
    2 | Client_001 |    CARE_001 |     220905 |     5 |         500000원
-------------------------------------------------------------------------
    3 | Client_001 |    CARE_002 |     220905 |     1 |         190000원
-------------------------------------------------------------------------
    4 | Client_002 |    CARE_001 |     220905 |     1 |          90000원
-------------------------------------------------------------------------
    5 | Client_003 |    CARE_003 |     220906 |     2 |         100000원
-------------------------------------------------------------------------
    6 | Client_003 |    CARE_001 |     220906 |     2 |         200000원
-------------------------------------------------------------------------
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

구매 정보 관리 창에서 3번을 입력후 삭제할 번호를 입력하면 해당 구매 정보 리스트의 행이 삭제되는 모습

2) C,Cpp화면

ShoppingManager.cpp

```cpp
//구매 정보 삭제 함수
void ShoppingManager::Shopping_Remove(int _num)
{
    for (int i = 0; i < S_Count; i++)
    {
        if (shoppingList.at(i)->getSNumber() == _num)
        {
            shoppingList.erase(shoppingList.begin() + i);
            S_Count--;
            cout << "\n구매 정보 삭제 완료!" << endl;
        }
    }
}
```

구매 정보 리스트 번호를 입력하면 입력 번호와 리스트 번호를 탐색 비교하여 해당 행을 삭제

Main.cpp

```cpp
case 3:
    sm.Display();
shopping_3:;
    cout << "\n삭제할 구매 번호를 입력해주세요 : "; cin >> s_num;
    if (!cin || s_num == 0)//정수형 인자 콘솔 입력부분 경계 검사
    {
        cout << "\n번호에 정수형 숫자를 입력하지 않았거나 0번째는 삭제할 수 없습니다." << endl;
        cin.clear();
        cin.ignore(INT_MAX, '\n');
        goto shopping_3;
    }
    sm.Shopping_Remove(s_num);
    //sm.FindCPrice(cm);
    break;
```

라. 구매 리스트 데이터 전체 삭제

1) 콘솔화면

```
3.구매 정보 관리
1.주문, 2.조회, 3.삭제, 4.전체삭제, 5.변경
번호를 입력해주세요 : 4

구매 정보 전체 삭제 완료!
++++++++++++++++++++++++++++++구매 정보 리스트++++++++++++++++++++++++++++++
------------------------------------------------------------------------------
 번호 |     고객 ID |     상품 ID |   구매 날짜 |  수량 |            총 금액
------------------------------------------------------------------------------
    0 | Client_001 |    CARE_001 |           0 |     0 |                 0원
------------------------------------------------------------------------------
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

구매 정보 관리 창에서 4번 입력 시 0번을 제외하고 나머지 데이터는 삭제완료 이유는 폴더에 저장되어 있는 Csv파일에 내용이 없으면 프로그램이 죽어 버리는 경우가 생기기 때문에 더미값을 부여

2) C,Cpp화면

```cpp
//구매 정보 전체 삭제 함수
void ShoppingManager::Shopping_Remove_All(ClientManager& _cm)
{
    while (!shoppingList.empty())
    {
        shoppingList.erase(shoppingList.begin() + 1, shoppingList.end());
        if (shoppingList.at(0)) break; //시작데이터는 남기고 나머지 데이터는 지워줌
    }
    S_Count = 1; //Count 1개는 남길 수 있게 함.
    Snumber = 0; //Shoppinglist count 초기화

    cout << "\n구매 정보 전체 삭제 완료!" << endl;
}
```

마. 구매 리스트 데이터 변경(날짜와 수량만)

1) 콘솔화면

```
변경할 구매 번호를 입력해주세요 : 6

변경할 날짜 : 220907
변경할 수량 : 1
고객 정보 변경 완료!!
가격 변경 완료
```

```
3.구매 정보 관리
1.주문, 2.조회, 3.삭제, 4.전체삭제, 5.변경
번호를 입력해주세요 : 2
++++++++++++++++++++++++++++++구매 정보 리스트++++++++++++++++++++++++++++++
------------------------------------------------------------------------------
 번호 |     고객 ID |     상품 ID |   구매 날짜 |  수량 |            총 금액
------------------------------------------------------------------------------
    0 | Client_001 |    CARE_001 |           0 |     0 |                 0원
------------------------------------------------------------------------------
    1 | Client_002 |    CARE_002 |      220905 |     5 |           1000000원
------------------------------------------------------------------------------
    2 | Client_001 |    CARE_001 |      220905 |     5 |            500000원
------------------------------------------------------------------------------
    3 | Client_001 |    CARE_002 |      220905 |     1 |            190000원
------------------------------------------------------------------------------
    4 | Client_002 |    CARE_001 |      220905 |     1 |             90000원
------------------------------------------------------------------------------
    5 | Client_003 |    CARE_003 |      220906 |     2 |            100000원
------------------------------------------------------------------------------
    6 | Client_003 |    CARE_001 |      220907 |     1 |            100000원
------------------------------------------------------------------------------
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

변경할 구매정보 번호를 입력하여 해당 날짜와 수량 데이터를 입력후 변경되었는지를 확인

2) C,Cpp화면

ShoppingManager.cpp

```
//구매 정보 변경 함수
void ShoppingManager::Shopping_Change(int _num, ClientManager& sh_cm, ProductManager& sh_pm)
{

    for (int i = 0; i < S_Count; i++)
    {
        if (shoppingList.at(i)->getSNumber() == _num)
        {
        shopping_date:;
            cout << "\n변경할 날짜 : "; cin >> date;
            if (!cin)//정수형 인자 콘솔 입력부분 경계 검사
            {
                cout << "\n등록 가격에 정수형 숫자를 입력하지 않았습니다." << endl;
                cin.clear();
                cin.ignore(INT_MAX, '\n');
                goto shopping_date;
            }
            shoppingList.at(i)->setSDate(date);
        shopping_quan:;
            cout << "변경할 수량 : "; cin >> quan;
            if (!cin)//정수형 인자 콘솔 입력부분 경계 검사
            {
                cout << "\n등록 가격에 정수형 숫자를 입력하지 않았습니다." << endl;
                cin.clear();
                cin.ignore(INT_MAX, '\n');
                goto shopping_quan;
            }
            shoppingList.at(i)->setSQuan(quan);
            cout << "고객 정보 변경 완료!!" << endl;

            for (int j = 0; j < sh_pm.Count(); j++)
            {
                if (sh_pm.productList.at(j)->getPId().compare(shoppingList.at(i)->getSPKProduct()) == 0)
                {
                    int price = sh_pm.productList.at(j)->getPPrice() * quan;
                    shoppingList.at(i)->setSAllprice(price);
                    cout << "가격 변경 완료" << endl;
                    FindCPrice(cm);
                }
            }
        }
    }
}
```

구매 정보 리스트의 번호를 비교하여 변경될 구매 정보 행의 날짜와 수량을 변경

```
//변경하는 구매 정보 리스트에서도 등급 변경
for (int j = 0; j < sh_pm.Count(); j++)
{
    if (sh_pm.productList.at(j)->getPId().compare(shoppingList.at(i)->getSPKProduct()) == 0)
    {
        for (int k = 0; k < sh_cm.Count(); k++)
        {
            if (sh_cm.clientList.at(k)->getCWord().compare(shoppingList.at(i)->getSPKClient()) == 0)
            {
                if (sh_cm.clientList.at(k)->getCGrade().compare(VVIP) == 0)
                {
                    int price = (sh_pm.productList.at(j)->getPPrice() * quan) * 0.90;
                    shoppingList.at(i)->setSAllprice(price);
                    cout << "가격 변경 완료" << endl;
                    break;
                }
                if (sh_cm.clientList.at(k)->getCGrade().compare(VIP) == 0)
                {
                    int price = (sh_pm.productList.at(j)->getPPrice() * quan) * 0.95;
                    shoppingList.at(i)->setSAllprice(price);
                    cout << "가격 변경 완료" << endl;
                    break;
                }
                if(sh_cm.clientList.at(k)->getCGrade().compare(Normal) == 0)
                {
                    int price = sh_pm.productList.at(j)->getPPrice() * quan;
                    shoppingList.at(i)->setSAllprice(price);
                    cout << "가격 변경 완료" << endl;
                    break;
                }
            }
        }
    }
}
```

변경하는 구매 수량에 따라 변경금액도 달라질 수 있도록 코드개선

Main.cpp

```
case 5:
shopping_4:;
    sm.Display();
    cout << "\n변경할 구매 번호를 입력해주세요 : "; cin >> s_num;
    if (!cin)//정수형 인자 콘솔 입력부분 경계 검사
    {
        cout << "\n번호에 정수형 숫자를 입력하지 않았습니다." << endl;
        cin.clear();
        cin.ignore(INT_MAX, '\n');
        goto shopping_4;
    }
    sm.Shopping_Change(s_num, cm, pm);
    sm.Display();
    //sm.FindCPrice(cm);
    break;
```

4. 구매리스트에서 해당 고객이 구매한 금액만큼 고객의 등급을 부여하여 고객 리스트에 자동 할당

가. 콘솔화면

```
3.구매 정보 관리
1.주문, 2.조회, 3.삭제, 4.전체삭제, 5.변경
번호를 입력해주세요 : 1
ClientCount : 3
++++++++++++++++++++++++++++++++고객 정보 리스트++++++++++++++++++++++++++++++++
--------------------------------------------------------------------------------
     고객ID |  고객 성함 |   고객 전화번호 |          고객 이메일 |    고객 등급
--------------------------------------------------------------------------------
 Client_001 |     정재현 |   010-2464-2739 |     0306jh@naver.com |         VIP
--------------------------------------------------------------------------------
 Client_002 |     정해영 |   010-5637-2739 |                not |        VVIP
--------------------------------------------------------------------------------
 Client_003 |     정민희 |   010-3421-2739 |                not |      Normal
--------------------------------------------------------------------------------
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Client의 등급은 아래의 구매정보 관리에 총 금액의 누적금액에 따라 등급이 결정

100만원 이상은 VVIP, 50만원 이상은 VIP, 이외에 경우는 Normal로 설정

```
3.구매 정보 관리
1.주문, 2.조회, 3.삭제, 4.전체삭제, 5.변경
번호를 입력해주세요 :
2
++++++++++++++++++++++++++++++구매 정보 리스트++++++++++++++++++++++++++++++
--------------------------------------------------------------------------------
 번호 |    고객 ID |     상품 ID |   구매 날짜 |  수량 |            총 금액
--------------------------------------------------------------------------------
    0 | Client_001 |    CARE_001 |          0 |     0 |               0원
--------------------------------------------------------------------------------
    1 | Client_002 |    CARE_002 |     220905 |     5 |         1000000원
--------------------------------------------------------------------------------
    2 | Client_001 |    CARE_001 |     220905 |     5 |          500000원
--------------------------------------------------------------------------------
    3 | Client_001 |    CARE_002 |     220905 |     1 |          190000원
--------------------------------------------------------------------------------
    4 | Client_002 |    CARE_001 |     220905 |     1 |           90000원
--------------------------------------------------------------------------------
    5 | Client_003 |    CARE_003 |     220906 |     2 |          100000원
--------------------------------------------------------------------------------
    6 | Client_003 |    CARE_001 |     220906 |     2 |          200000원
--------------------------------------------------------------------------------
    7 | Client_001 |    CARE_003 |     220907 |     1 |           47500원
--------------------------------------------------------------------------------
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

현재 추가된 Client_001은 50000원인 CARE_003 1개 상품을 구입하여 VIP등급으로

5%할인을 받아 47500원의 할인가를 제공 받음, 변경을 하면 변경 수량에 따른 금액도 할인가 적용

나. C, Cpp

```cpp
//쇼핑 리스트에 따른 고객 총금액 탐지 함수 client->cprice->getCPrice(), shopping->getAllPrice()
    //이 함수는 각각 추가 제거 변경 기능에서 기능에 따른 고객 등급이 최신화 될 수 있도록 적용
void ShoppingManager::FindCPrice(ClientManager& _cm)
{
    //초기화할 변수값 생성 등급, 가격
    string grade;
    //int cprice = 0;

    //먼저 client가 가지고 있는 사용 금액 함수 초기화
    for (int i = 0; i < _cm.Count(); i++)
    {
        _cm.clientList.at(i)->setCPrice(0); //0으로 초기화
    }

    for (int j = 0; j < _cm.Count(); j++) //현재 설정된 고객 리스트 개수 만큼 for문으로 탐색 고객은 i
    {
        for (int i = 0; i < S_Count; i++) //현재 설정된 구매 리스트의 개수 만큼 for문으로 탐색 구매정보는 j
        {
            if (_cm.clientList.at(j)->getCWord().compare(shoppingList.at(i)->getSPKClient()) == 0)
                //탐색한 구매 리스트 와 고객 리스트의 PK가 일치하는 지 확인후 구매 리스트의 총금액을
                //더하여 고객등급을 초기화
            {
                //구매 정보 1행에 해당하는 값 불러오고 cprice에 누적
                //cprice += shoppingList.at(i)->getSAllprice();

                //해당하는 고객 행에 사용한 금액 누적
                _cm.clientList.at(j)->setAddCPrice(shoppingList.at(i)->getSAllprice());
            }
        }
    }

    //누적된 금액 누적 값에 따른 등급 지정
    for (int i = 0; i < _cm.Count(); i++)
    {
        if (_cm.clientList.at(i)->getCPrice() >= 1000000) //해당하는 고객 누적값이 1,000,000 이상인 경우
        {
            grade = "VVIP";
            _cm.clientList.at(i)->setCGrade(grade); //VVIP로 등급 설정
            //cout << "VVIP로 설정" << endl;
        }
        else if (_cm.clientList.at(i)->getCPrice() >= 500000 &&
            _cm.clientList.at(i)->getCPrice() < 1000000) //해당하는 고객 누적값이 500000 이상 1000000 미만인 경우
        {
            grade = "VIP";
            _cm.clientList.at(i)->setCGrade(grade);//VIP로 등급 설정
            //cout << "VIP로 설정" << endl;
        }
        else if(_cm.clientList.at(i)->getCPrice() >= 0 &&
            _cm.clientList.at(i)->getCPrice() < 500000)
        {
            grade = "Normal"; //500000미만인 경우
            _cm.clientList.at(i)->setCGrade(grade);//Normal로 등급 설정
            //cout << "Normal로 설정" << endl;
        }
    }
}
```

누적된 금액에 따라 등급을 결정하는 함수 이후 누적 가격에 따라 고객의 등급도 부여가 됨

```
//등급별 가격 할인
if (C_ref.clientList.at(i)->getCGrade().compare(VVIP) == 0)
{
    price = (P_ref.productList.at(j)->getPPrice() *  _quantatiy) * 0.90; //VVIP면 10퍼센트 할인
    shoppingList.push_back(new Shopping(_num,
        _clpk, _prpk, _date, _quantatiy, price)); // 위의 PK값 일치가 충족 된다면 쇼핑 리스트에 데이터 추가
    Snumber += 1;
    S_Count += 1;
    cout << "\n구매 정보 추가 완료" << endl;
}
if (C_ref.clientList.at(i)->getCGrade().compare(VIP) == 0)
{
    price = (P_ref.productList.at(j)->getPPrice() *  _quantatiy) * 0.95; //VIP면 5퍼센트 할인
    shoppingList.push_back(new Shopping(_num,
        _clpk, _prpk, _date, _quantatiy, price)); // 위의 PK값 일치가 충족 된다면 쇼핑 리스트에 데이터 추가
    Snumber += 1;
    S_Count += 1;
    cout << "\n구매 정보 추가 완료" << endl;
}
if (C_ref.clientList.at(i)->getCGrade().compare(Normal) == 0)
{
    price = P_ref.productList.at(j)->getPPrice() *  _quantatiy; //Normal이면 아무것도 없음
    shoppingList.push_back(new Shopping(_num,
        _clpk, _prpk, _date, _quantatiy, price)); // 위의 PK값 일치가 충족 된다면 쇼핑 리스트에 데이터 추가
    Snumber += 1;
    S_Count += 1;
    cout << "\n구매 정보 추가 완료" << endl;
}
```

고객 리스트에 저장된 등급에 따라 할인가 적용 함수 입력, 변경 함수안에서 구현

5. 모든 데이터 파일 또는 csv로 저장

```
//고객 정보 파일로 저장 함수
void ClientManager::Save()
{
    ofstream file;
    file.open("clientlist.csv");
    if (!file.fail()) {
        for (const auto& c : clientList) {
            file << c->getCWord() << ',';
            file << c->getCName() << ',';
            file << c->getCPhone() << ',';
            file << c->getCEmail() << ',';

            //파일로 등급과 구매한 금액 저장
            file << c->getCGrade() << ',';
            file << c->getCPrice() << ',';

            //Client의 머릿수
            file << C_Count << endl;
        }
        file << endl;
    }
    file.close();
    cout << "Client 파일 저장 완료" << endl;
}
```

만일 파일 찾으면 파일을 저장하기 위해 해당 List안의 데이터를 "A,B,.....Z"형태로 .txt 나 .csv형태로 저장

```cpp
//고객 정보 파일로 불러오기 함수
void ClientManager::Load()
{
    //vector<Client*> vecList;
    ifstream file; //input
    file.open("clientlist.csv");
    if (!file.fail()) {
        while (!file.eof()) {
            vector<string> row = parseCSV(file, ',');
            if (row.size()) {
                //해당 고객이 사용한 금액 int 형으로 변환
                int cprice = atoi(row[5].c_str());
                //고객의 인원수 저장
                int count = atoi(row[6].c_str());

                //row[4]와 cprice에서 client생성자와 비교하여 파일에서 불러드림
                Client* c = new Client(row[0], row[1],
                    row[2], row[3], row[4], cprice);
                clientList.push_back(c);
                C_Count = count;
            }
        }
    }
    file.close();
    cout << "Client 파일 불러오기 완료" << endl;
}
```

폴더에 저장되어 있는 파일을 찾은 경우 해당 리스트에 데이터 값들을 ','를 구분해서 불러오기

```cpp
//파일에서 불러오는 정보를 파싱하여 ','문은 구분할 수 있는 함수를 제작
vector<string> ClientManager::parseCSV(istream& file,
    char delimiter)
{
    stringstream ss; //문자행렬을 받는 stringstream
    vector<string> row;
    string t = " \n\r\t"; // 파일 내의 \n\r\t에서 공백값을 지우는 변수 설정

    while (!file.eof()) { //file의 끝까지
        char c = file.get();
        if (c == delimiter || c == '\r' || c == '\n') {//c가 받는 ','와 \r, \n인 경우
            if (file.peek() == '\n') file.get();
            string s = ss.str();
            s.erase(0, s.find_first_not_of(t)); // string인 t변수에서 해당 \n\r\t는 지우겠다는 함수
            s.erase(s.find_last_not_of(t) + 1); // 마지막까지
            row.push_back(s); //vector로 저장된 값들을 넣어줌
            ss.str("");
            if (c != delimiter) break; //','을 만나면 함수종료
        }
        else { //c가 받는  ','와 \r, \n이 아닌 경우
            ss << c;
        }
    }
    return row;
}
```

','와 '₩n', '₩r'을 구분하기 위한 파싱 함수 제작

로드 세이브 기능은 메인 함수 내에서 구현, Load()호출 이후 프로그램 종료시 저장하겠다는 문구로 저장

```
Client 파일 불러오기 완료
Product 파일 불러오기 완료
Shopping 파일 불러오기 완료

안녕하십니까? 임플란트 전용 치과 전문 센터 입니다.

아래에 해당하는 번호를 입력하여 원하시는 정보를 활용하십시오.
1. 고객 정보 관리, 2. 상품 정보 관리, 3.구매 정보 관리, 4.종료
번호를 입력하여 주세요 :
```

```
저장후 프로그램을 종료하시겠습니까? (y/n) : y
Client 파일 저장 완료
Product 파일 저장 완료
Shopping 파일 저장 완료
프로그램 종료!!
```

6. 고객, 상품 클래스내의 가상함수를 불러드리는 Abstract 클래스

```
#ifndef _ABSTRACT_MANAGER_H_
#define _ABSTRACT_MANAGER_H_

class AbstractManager
{
public:
    virtual void Display() = 0;
    virtual void Save() = 0;
    virtual void Load() = 0;
    virtual int Count() = 0;
    virtual void RemoveAll() = 0;
};

#endif
```

```
class ClientManager : public AbstractManager //ClientManager클래스는 AbstractManager클래스 상속
{
public:

    //AbstractManager클래스의 가상함수를 오버라이드
    virtual int Count() override { return C_Count; };
    virtual void RemoveAll() override;
    virtual void Display() override;
    virtual void Save() override;
    virtual void Load() override;
```

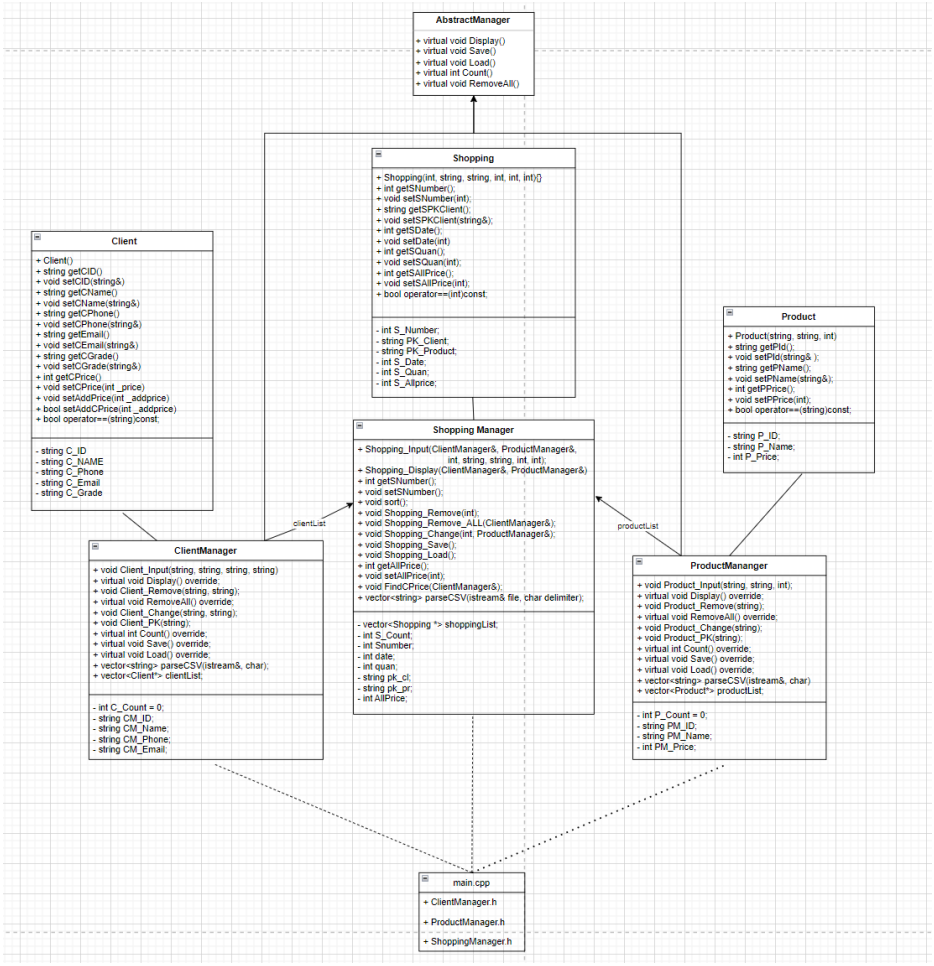
```
class ProductManager : public AbstractManager
{
public:
    void Product_Input(string _id, string _name, int _price);
    void Product_Remove(string _id);
    void Product_Change(string _id);
    void Product_PK(string _id);

    //AbstracuManager 클래스의 가상함수 오버라이드
    virtual void RemoveAll() override;
    virtual void Display() override;
    virtual int Count() override { return P_Count; };
    virtual void Save() override;
    virtual void Load() override;
```

고객과 상품의 매니저 클래스에서 공통으로 사용되는 함수들만 순수 가상함수로 묶어둠 이유는 가상함수를 쓰면 불필요한 함수 중복 사용을 줄일 수 있기 때문

※내부 구조 : - 클래스 다이어그램을 이용해서 구조 설정

- 클래스 관계도(UML)



- 데이터 구조도(CSV)

1. ClientList구조

	A	B	C	D	E	F	G
1	Client_001	정재현	010-2464-	0306jh@n	VIP	690000	3
2	Client_002	정해영	010-5637-	not	VVIP	1090000	3
3	Client_003	정민희	010-3421-	not	Normal	300000	3

A : ClientID, B, : ClientName, C : ClientPhone, E : ClientGrade, F : ClientAllPrice, G : ClientCount;

ClientList에서는 A가 PK값으로 고객, 구매 정보 리스트의 기능들을 활용할 수 있는 키워드이다.

2. ProductList구조

	A	B	C	D
1	CARE_001	오스템	100000	6
2	CARE_002	니어	200000	6
3	CARE_003	원데이	50000	6
4	CARE_004	상악동	70000	6
5	CARE_005	무치악	150000	6
6	CARE_006	뼈이식	80000	6

A : ProductID, B : ProductName, C : ProductPrice, D : ProductCount

ProductList에서도 A가 PK값으로 상품, 구매 정보 리스트의 기능들을 활용할 수 있는 키워드이다.

3. ShoppingList구조

	A	B	C	D	E	F	G
1	0	Client_001	CARE_001	0	0	0	7
2	1	Client_002	CARE_002	220905	5	1000000	7
3	2	Client_001	CARE_001	220905	5	500000	7
4	3	Client_001	CARE_002	220905	1	190000	7
5	4	Client_002	CARE_001	220905	1	90000	7
6	5	Client_003	CARE_003	220906	2	100000	7
7	6	Client_003	CARE_001	220906	2	200000	7

A : ShoppingListNumber, B : ClientPK, C : ProductPK, D : 구매날짜, E : 구매수량, F : 구매총액, G : ShoppingCount

ClinetPK는 B, ProductPK는 C에 위치하여 구매 정보 리스트에서 입력, 조회, 부분에 입력값과 일치하면 코드내의 연산을 처리하게 되어 원하는 정보를 불러올 수 있음.

단, 삭제, 변경 기능은 단순히 구매정보 리스트 내에서 이루어 지기 때문에 A를 키값으로 삭제 변경 기능을 구현

※발전 방향 :

- 미진행 내용들
- 더 많은 책임중심 프로그램으로 변경하지 못한점
- 파일 내에 csv는 있는데 내용이 없으면 프로그램이 죽는 현상 미개선
- 해당 정보를 알아보고 질문에 나가고 싶을 때 뒤로가기 버튼이 없어서 불편함이 있음
- 누적된 구매 리스트 금액이 구매가와 할인가를 따로 만들지 않아서 할인가 그대로 누적하여 VVIP의 등급 상승에 컴플레인이 들어올 수 있음(업계비밀)
- AbstractManager클래스에서 ShoppingManager클래스까지는 상속하지 못함