HW 2

1.

Exercise 4

a. GroceryStore는 식료품 이름, 가격을 저장하는 1차원 리스트들과 총액을 가진다.

b.

Transformers: MakeEmpty(리스트 안을 초기화), SaveItem(새로운 아이템 저장), DeleteItem(아이템 삭제), CheckOut(물건을 계산하면 물품, 가격 리스트에서 해당 위치의 아이템 차감), Total(총액을 저장)

Observers: IsFull(리스트 안이 꽉 찼는지 확인), LengthIs(리스트 안에 저장된 값의 개수 반환), RetriveItem(리스트 안의 아이템 가져오기)

Iterators: ResetList(iteration 전 리스트의 제일 앞으로 이동), GetNextItem(다음 아이템 가져오기)

c. void CheckOut(string buy);

d.

void CheckOut(string buy) {

int idx = RetrieveItem(this->stuff, buy); //int RetriveItem(string\* stuff, string buy)

if (idx != -1) //RetrieveItem: buy 찾지 못하면 -1 반환

DeleteItem(this->stuff, , this->price, idx); //void DeleteItem(string\* stuff, int\* price, int idx)

else

cout << "there's no ' " << buy << " ' in our grocery store.\n";

e. c는 logical level에서의 표현으로 함수의 리턴 타입, 입력 파라미터만을 표현하고 있고 d는 implementation level에서의 표현으로 함수의 리턴 타입, 입력 파라미터뿐만 아니라 함수 내부의 동작 구현부까지 나타내고있다.

Exercise 17

a. string타입 2차원 배열

b. 단일 제품의 정보를 담고 있는 struct

c. string타입 1차원 배열

d. unsigned int타입 2차원 배열

e. unsigned int타입 2차원 배열 (년도, 사망자 수) / 각 주의 정보(2차원 배열)의 포인터를 저장하는 1차원 배열

f. unsigned int타입 3차원 배열 (년도, 월, 사망자 수) / 각 주의 정보(3차원 배열)의 포인터를 저장하는 1차원 배열

g. 이름, 주소 및 모든 친구의 전화 정보를 가지는 struct

h. int타입 1차원 배열

Exercise 21

a. true

b. false

Exercise 22

(class의 사용자와 만든 사람이 다르다고 가정할 때,) User level에 있는 사용자는 logical level의 domain, operation 선언만 보고 사용하고, implementation level에서의 자세한 구현부는 신경 쓰지 않아도 된다. 🡪 시스템 뷰어의 입장에서 간단하고 필요한 것만 볼 수 있다.

Exercise 23

Inheritance, friend relationship, polymorphism