**Design Project A**

Project #3

\* 부리또 매장 관리 프로그램을 작성하시오. 자세한 요구사항은 아래와 같다.

**1. System Requirements**

- 부리또 매장 관리 프로그램을 구현하기 위한 객체는 App, BuritoFactory, TransactionManager, Burito, BaseBurito, Decorator, Jumbo, Sausage, Drink, Fries, ComboDiscount, SetDiscount이다.

- App은 BuritoFactory\* bf, TransactionManager\*를 가지고 있다.

- BuritoFactory는 팩토리 패턴을 적용하여, 주문 받은 메뉴의 아이디를 파라메터로 받아 부리또를 생성하여 반환한다. 생성할 수 있는 메뉴는 단품, 콤보, 세트 이다.

- TransactionManager는 주문 내역을 관리하는 클래스로써, 받은 주문 내역 리스트를 가지고 있다.

- 부리또는 데코레이터 패턴을 적용하여 생성한다. 데코레이터 클래스들은 Burito, BaseBurito, Jumbo, Sausage, Drink, Fries, ComboDiscount, SetDiscount가 있다.

- Burito는 데코레이터 패턴을 구현하기 위한 최상위 클래스로써, Decorator 클래스와 BaseBurito클래스의 부모 클래스이다. 순수 가상함수 getCost와 getDescription을 가지고 있다.

- Decorator는 추가 재료 및 할인 정보 클래스를 부리또에 적용하기 위한 Jumbo, Sausage, Drink, Fries, ComboDiscount, SetDiscount의 부모클래스이다.

- Jumbo, Sausage, Drink, Fries, ComboDiscount, SetDiscount 클래스는 getPrice()와 getDescription() 함수를 오버라이딩하여 자신의 가격 및 구성품을 반환한다.

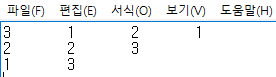
- 메인에서는 오직 App의 run()만을 호출한다.

**2. 공통 기능**

* 데이터 적재 기능 : 프로그램 시작 시, Order.txt으로부터 데이터를 읽어  
  TransactionManager의 orderTransactions에 저장시킨다.
  + Order DB

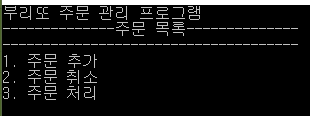
하나의 Order에 대한 정보는 리스트 크기, 메뉴의 아이디, 추가 재료의 아이디 로 구성되며, 추가 재료의 아이디는 여러 개 쓰일 수도 있다, 탭을 이용하여 필드를 구분한다.

개행문자를 통해 각각의 Order를 구분한다.



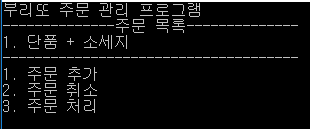
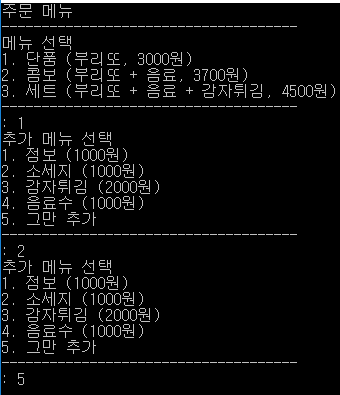
**[사용자 기능]**

* 관리 프로그램 사용자에게 현재 받은 주문들의 목록과 수행할 수 있는 기능들을 출력하여 보여준다..



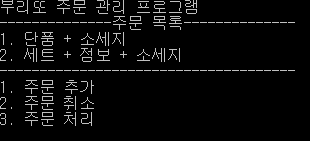
**[주문 추가]**

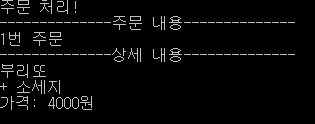
* 고객으로부터 주문을 전달받아 추가한다.



**[주문 처리]**

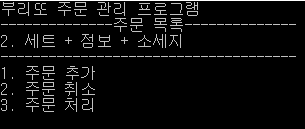
* 주문 목록의 가장 앞의 주문을 처리하여 처리 내용을 출력한다.

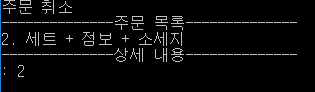




**[주문 취소]**

* 사용자가 취소한 주문을 주문 목록에서 삭제한다.





**[특수 상황 조건]**

* 실제 프로그램과 약간의 차이로 발생하는 특수한 상황의 결정은 본인이 직접 규칙을 정하고, 발표 시 설명한다.

**3. 예외 처리**

- 주어진 범위 외의 값 또는 잘못된 타입의 입력 값 입력 시

**4. Program Requirements**

- 모든 리스트는 STL을 활용하여 구현

- Factory, Decorator 패턴을 system requirement에 맞게 적용시켜 구현할 것

- 소스코드에 함수, 중요 변수, 중요 기능, 중요 알고리즘 등은 반드시 주석을 단다.

**5. Interface Example**

- 자신이 편리한 인터페이스를 자유롭게 디자인한다.

**6. 추가기능 구현**

- 없음