하위 설계서

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | 1. 목적 |  * 고급객체지향 프로그래밍의 실습 과제이다. 클래스, 상속, 자료구조, 패턴 등을 사용한 프로그래밍을 통하여 실력을 향상시킨다. * 실전 프로그래밍에서의 예외 처리, 다양한 선택변수 등을 접하고 대비하는 능력을 향상시킨다.  |  | | --- | | 1. 사용자 Targeting |  * 커피숍 매니지먼트 프로그램을 이용하고자 하는 점원 및 관리자를 대상으로 하며, 직관적인 인터페이스로 누구나 쉽게 커피숍 매니지먼트 프로그램을 이용할 수 있어야 한다.  |  | | --- | | 1. 설계 원칙 |   - 커피숍 매니지먼트 프로그램을 구현하기 위한 객체는 App, CounterFacade, ManagerFacade, ProductPriceCalculator, DiscountAlgorithm, CoffeeFactory, Coffee, OrderTransactionManager, Transaction, StockManager, SaleManager 이다.  - DiscountAlgorithm은 추상 클래스로써, DiscountDrinkInStore, DiscountTakeOut, DiscountCoupon 클래스의 부모 클래스이다. 순수가상함수인 discount()를 가지고 있다.  - DiscountDrinkInStore, DiscountTakeOut, DiscountCoupon은 discount() 함수를 오버라이딩하여 매장에서 마시는 경우, 테이크아웃 하는 경우, 쿠폰을 사용하는 경우에 대한 할인된 음료의 가격을 계산한다.  - Coffee는 추상 클래스로써, BaseCoffee, IngerdientDecorator의 부모 클래스이다. 순수가상함수인 getPrice()와 getIngredientsID() 함수를 가지고 있다.  - IngredientDecorator는 Coffee 클래스에 데코레이터 패턴들 적용하기 위한 MilkDecorator, MochaDecorator, ShotDecorator, WhipDecorator의 부모클래스이다.  - MilkDecorator, MochaDecorator, ShotDecorator, WhipDecorator 클래스는 getPrice()와 getIngredientsID() 함수를 오버라이딩하여 자신의 가격 및 아이디를 반환한다.  - Transaction 클래스는 고객이 주문한 내역을 저장하기 위한 클래스로써, 음료의 아이디와 추가 재료들의 아이디를 저장하는 int형 vector 변수와, 음료의 가격을 저장하는 정수형 변수를 가지고 있다.  - ProductPriceCalculator는 DiscountAlgorithm 클래스를 활용하여 음료의 할인된 최종 가격을 계산한다..  - CoffeeFactory 클래스는 음료의 아이디를 입력 받아 아이디에 해당되는 음료를 생성하여 반환한다.  - OrderTransactionManager는 고객이 주문한 음료들의 내역을 먼저 주문한 고객이 먼저 받도록 관리한다.  - StockManager는 현재 커피숍이 가지고 있는 재료들의 재고를 관리한다.  - CounterFacade는 점원이 사용하는 기능을 구현한 클래스로써, 주문 받기, 커피 제공, 주문 취소 기능을 가지고 있다.  - ManagerFacade는 점장이 사용하는 기능을 구현한 클래스로써, 재료 발주, 재고 확인, 주문 내역 확인 기능을 가지고 있다.  - 메인에서는 오직 App의 run()만을 호출한다.   |  | | --- | | 1. 개발 도구 |   Microsoft Visual Studio C++ 2013   |  | | --- | | 1. 적용 범위 |  * 장소적 범위 : 커피숍 매니지먼트 프로그램이 사용될 수 있는 장소 * 프로그램적 범위 : 카페, 커피숍 관리 프로그램 등  |  | | --- | | 1. 시나리오 흐름도 |   시나리오 흐름도     |  | | --- | | 1. 인터페이스 정의 |  * 문장은 사용자가 해당 기능에 대한 사용법을 명확히 알 수 있도록 명시한다. * 글자가 지저분하지 않도록 띄어쓰기와 줄 바꿈을 해준다. * 입력은 키보드로만 받는다. * 한국어와 영어 외의 언어는 사용하지 않는다.  |  | | --- | | 1. 헤더와 cpp, 클래스정의 |  * 클래스 선언은 헤더파일을 만들고 메인 cpp에 불러온다. * 헤더에서 선언한 클래스는 헤더와 같은 이름의 cpp에 구현한다.  |  |  | | --- | --- | | **헤더와 cpp** | | | 기능 | 이름 | | 클래스 선언 | App.h  BaseCoffee.h  Coffee.h  CoffeFactory.h  CounterFacade.h  DiscountAlgorithm.h  DiscountCoupon.h  DiscountDrinkInStore.h  DiscountTakeOut.h  IngredientsDecorator.h  ManagerFacade.h  MilkDecorator.h  MochaDecorator.h  OrderTransactionManager.h  ProductPriceCalculator.h  SaleManager.h  ShotDecorator.h  Transaction.h  WhipDecorator.h | | 클래스 구현 | App.cpp  BaseCoffee.cpp  Coffee.cpp  CoffeFactory.cpp  CounterFacade.cpp  DiscountAlgorithm.cpp  DiscountCoupon.cpp  DiscountDrinkInStore.cpp  DiscountTakeOut.cpp  IngredientsDecorator.cpp  ManagerFacade.cpp  MilkDecorator.cpp  MochaDecorator.cpp  OrderTransactionManager.cpp  ProductPriceCalculator.cpp  SaleManager.cpp  ShotDecorator.cpp  Transaction.cpp  WhipDecorator.cpp | | 메인 구현 | Main.cpp |  |  |  | | --- | --- | | **클래스** | | | 클래스 이름 | 기능 | | App.h 프로그램 기능 총괄  BaseCoffee.h 커피 베이스  Coffee.h 커피 객체 클래스  CoffeFactory.h 음료를 생산하는 팩토리 클래스  CounterFacade.h 점원 기능  DiscountAlgorithm.h 할인된 음료의 가격을 계산  DiscountCoupon.h 쿠폰 사용 할 때  DiscountDrinkInStore.h 가게 안에서 마시고 갈 때  DiscountTakeOut.h 테이크 아웃 할 때  IngredientsDecorator.h 음료에 추가하는 재료들  ManagerFacade.h 점장 기능  MilkDecorator.h 재료로 쓰일 우유  MochaDecorator.h 재료로 쓰일 모카  OrderTransactionManager.h 주문 내역 관리  ProductPriceCalculator.h 최종 가격 계산  SaleManager.h 재고 관리  ShotDecorator.h 재료로 쓰일 샷  Transaction.h 주문 내역  WhipDecorator.h 재료로 쓰일 휘핑크림 | |   ▣ 클래스 별 요소와 기능 ▣   |  | | --- | | **App.h** | | #pragma once  #include "CounterFacade.h"  #include "ManagerFacade.h"  class App  {  public:  App();  ~App();  void run();  private:  CounterFacade\* cf; // 점원 Facade  ManagerFacade\* mf; // 점장 Facade  }; |  |  |  | | --- | --- | | **CounterFacade.h** | | | #pragma once  #include "CoffeeFactory.h"  #include "OrderTransactionManager.h"  #include "StockManager.h"  #include "SaleManager.h"  #include "ProductPriceCalculator.h"  class CounterFacade  {  public:  CounterFacade();  CounterFacade(OrderTransactionManager\* \_otm, StockManager\* \_sm, SaleManager\* \_slm);  ~CounterFacade();  void order();  void serveCoffee();  void cancle();  private:  CoffeeFactory cf; //Coffee 팩토리  ProductPriceCalculator ppc; // 가격 계산  OrderTransactionManager\* otm; // 주문 관리  StockManager\* sm; // 재고 관리  SaleManager\* slm; // 가격 관리  }; |  |  | | --- | | **ManagerFacade.h** | | #pragma once  #include "StockManager.h"  #include "OrderTransactionManager.h"  #include "SaleManager.h"  class ManagerFacade  {  public:  ManagerFacade();  ManagerFacade(OrderTransactionManager \* \_otm, StockManager\* \_sm, SaleManager\* \_slm);  ~ManagerFacade();  void orderIngredients();  void printStockStatus();  void printTransactionStatus();  private:  OrderTransactionManager \* otm; // 주문 관리  StockManager\* sm; // 재고 관리  SaleManager\* slm; // 가격 관리  }; |  |  | | --- | | **CoffeeFactory.h** | | #pragma once  //#include "Coffee.h"  #include "BaseCoffee.h"  #include "MilkDecorator.h"  #include "MochaDecorator.h"  #include "ShotDecorator.h"  #include "WhipDecorator.h"  class CoffeeFactory  {  public:  CoffeeFactory();  ~CoffeeFactory();  Coffee\* makeCoffee(int coffeeID); // 커피 생성  }; |  |  | | --- | | **IngredientsDecorator.h** | |  | | #pragma once  #include "Coffee.h"  class IngredientsDecorator :  public Coffee  {  public:  IngredientsDecorator();  IngredientsDecorator(Coffee\* coffee);  ~IngredientsDecorator();  protected:  Coffee\* pCoffee;  }; | |  |  |  | | --- | | **OrderTransactionManager.h** | | #pragma once  #include "Transaction.h"  #include <vector>  #include <iostream>  using namespace std;  class OrderTransactionManager  {  public:  OrderTransactionManager();  ~OrderTransactionManager();  void addTransaction(Transaction t);  Transaction getFrontTransaction();  void deleteTransactionFromFront();  void deleteTransactionByIndex(int idx);  void printTransactions();  private:  vector<Transaction> orderTransactions; // 주문 순서  }; | |  |  |  | | --- | | 1. Main의 변수와 cpp정의 |   App : App 클래스 선언   |  | | --- | | 1. 프로젝트 추진 일정 |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  | **추진일정** | | | | | | | | | | | | **1주** | | **2주** | | **3주** | | **4주** | | **5주** | | | | **요구사항 분석** |  |  |  |  |  |  |  |  |  |  |  | | **헤더,  cpp작성** |  |  |  |  |  |  |  |  |  |  |  | | **main작성** |  |  |  |  |  |  |  |  |  |  |  | | **1차 완성** |  |  |  |  |  |  |  |  |  |  |  | | **옵션 추가** |  |  |  |  |  |  |  |  |  |  |  | | **버그 수정** |  |  |  |  |  |  |  |  |  |  |  | | **검토 및  제출** |  |  |  |  |  |  |  |  |  |  |  | |