**Информация для проверяющего**.

Этот текст содержит краткое описание решения.

Поскольку задание сформулировано довольно расплывчато (что, может быть, и хорошо, потому что заставляет не только писать код, но и формулировать задачу, и проектировать решение), придется немного пофантазировать. Фактически нужно сформулировать TЗ. Итак, попробую поставить задачу так, как я ее вижу.

Вендинг может содержать несколько слотов, в которые помещаются продукты. Машина заполняется слотами, как стек. С машиной имеет дело персонал обслуживания и пользователь.

Интерфейс персонала.

Действия персонала с машиной.

* Получить машину new\_Vending
* Получить число слотов vending\_Capacity
* Получить число свободных позиций автомата free\_Positions
* Вставить слот в машину add\_Slot
* Получить доступ к слоту с заданным номером get\_Slot
* Получить информацию о заполении продуктами product\_Availability

Действия персонала со слотом.

* Получить слот new\_Slot
* Получить число позиций слота slot\_Capacity
* Поместить продукт в слот add\_Product
* Получить число свободных позиций слота free\_Positions

Действия персонала с продуктом.

* Получить продукт new\_Product

Интерфейс пользователя включает только следующие действия пользователя с машиной.

* Получить информацию о имеющихся продуктах – product\_Availability.
* Купить продукт - buy\_Product.

Интерфейсы пользователя и персонала предоставляются отдельно (в виде заголовочных файлов) и никак не зависят от реализации (структуры классов и алгоритмов).

Сценарии работы с автоматом такие.

Сначала пресонал обслуживания инсталлирует машину, используя функци new\_Vending, new\_Slot, add\_Slot, заполнят ее продуктами, используя функции new\_Product, get\_Slot, add\_Product.

Далее пользователи совершают покупки, используя функции product\_Availavility и buy\_Product.

Функция get\_Availability позвоялет получить информацию о заполнении автомата и отправить робота для заправки автомата, который использует функции free\_Positions, get\_Slot, new\_Product, add\_Product.

Продукт обладает следующими характеристиками.

* Название (строка).
* Идентификационный код (тип перечисления).
* Стоимость.

Я считаю, что каждому продукту резонно назначить идентификационный код, номенклатуру продуктов резонно опеределять отдельной таблицей Assortment. Вообще-то по-хорошему ассортимент и цены также должны быть изменяемой информацией (это уже действия администрации), но пока оставим так.

Я решил, что стоимость не может выражаться типом float (потому, что не может быть, например, равна 3,33333). Стоимость я выражаю в наменьших денежных единицах (таких как копейки, центы, евроценты). Для класса Price определен оператор <<, который позволяет значение 150 выводить, как 1.50.

В плане развития функционала я могу предложить следующее. Для оптимизации бизнес-процессе можно добавить функции, позволяющие получать данные о статистике продаж (включая распределение по времени), эти данные собираются и анализируются какой-то Business Support System, где какой-нибудь ИИ принимает решения о том, когда, и куда отпралять робота для пополнения автомата, какие продукты и сколько нужно загружать.

use\_case.cpp

vending\_user.h

vending\_mgt.h

vеnding.cpp

vending.h

Схема взамосвязей между модулями.

Проект решения таков. Код состоит из следующих файлов.

vending\_mgt.h описание интерфейса персонала обслуживания

vending\_user.h описание интерфейса пользователя

vending.h описание классов

vending.cpp реализация классов, и интерфейсов персонала и пользователя

use\_case.cpp тестовый пример использования интерфейса персонала и интерфейса пользователя

Модуль use\_case, в котором реализована функция main, моделирует процесс инсталляции и первоначального заполнения машины а также возможную организацию интерфейса машины и процесс продаж.

Схема взаимосвязей между модулями кода показана на рисунке. На схеме стрелками показано включение заголовочных файлов, заголовочные файлы показаны белым цветом.

Не реализоваты следующие вещи.

1. Обработка ошибок условная. Хотя алгоритмы таковы, что ошибок не должно возникать, по-хорошему нужно выбрасывать исключения в некоторых ситуациях.

* Методы Slot::add\_Product и Vending::add\_Slot должны выбрасывать исключения в тех случах, когда невозможно добавить продукт или слот.
* Методы Slot::get\_Product и Vending::get\_Product должны выбрасывать исключения в тех случаях, когда продукт невозможно получить, а не заставлять вызывающего анализировать возращаемое значение на nullptr.
* Метод Vending::get\_Slot также должен выбрасывать ислючение при ошибочном задании номера слота.

2. Еще один недостаток, вообще говоря, необходимо исправить. Поскольку функция get\_Slot возвращает указатель на объект класса Slot, такой объект можно по ошибке удалить. Поэтому, если слот уже находится в машине, деструктор класса Slot должен предпринимать соответствующие действия.

Но пока я решил оставить все, как есть, в качестве заготовки для дальнейшего развития.