ЧИТАЕМ:

**2 СЛАЙД:**

Группа: III – ATM терминал банка.

1) Отобразить список операций. (внесение, снятие, оплата)

2) Можно после внесения выбрать оплата и отобразить возможность оплатить сотовую связь и коммунальные услуги.

3) Если денег недостаточно, то отображаем список операций с доступной суммой.

4) При операции отображаем список действий того как осуществляется операции.

**3 СЛАЙД:**

* **Создать отображение списка операций. (внесение, снятие, оплата)**
* **Реализовать возможность оплаты сотовой связи и коммунальных услуг.**
* **Реализовать отображение возможных действия, учитывая количество денег на счету клиента.**
* **Реализовать отображение списка действий того, как осуществляется операции.**

**4 СЛАЙД:**

**Используемые паттерны:**

* **Фасад**
* **Стратегия**

**5 СЛАЙД:**

**Класс BaseClient реализует клиента банкомата, который имеет такие поля и методы как:**

* **NAME - Имя клиента**
* **CurrentTerminal - банк к которому он относится**
* **Cash - количество наличных**
* **SumOfMoneyFromAccount - Количество средств на счёте**
* **interact - взаимодействие клиента с ATM терминалом**
* **DepositFunds - клиент хочет внести деньги**
* **WithdrawFunds - клиент хочет снять наличные**
* **ToPay - клиент хочет оплатить услугу**

Клиентов может быть несколько и каждому клиенту можно задать определённое число наличных с собой и количество денег на карте. Клиент может пользоваться только тем интерфейсом, который предоставляет ему ATM терминал, напрямую взаимодействовать с компонентами, например, с кассетой или диспансером, клиент не может. Класс клиента так же реализует интерфейс IClient, в котором есть ограниченный список операций взаимодействия с каким-либо ATM терминалом: снятие наличных, внесение наличных, оплата услуг и вывод списка возможных операций.

**6 СЛАЙД:**

**Класс банковского терминала имеет такие поля и методы как:**

* **Кассета – количество купюр разного номинала в банкомате**
* **Контроллер – содержит пользовательский интерфейс для использования банкомата**
* **Принтер – печать чеков об операции**
* **Диспенсер – процесс внесения или выдачи купюр**
* **Конструктор, принимающий 4 компонента: кассета, принтер, контроллер, диспансер**
* **PrintSetOperation - вывод списка операций**
* **DepositFunds – внесение средств**
* **WithdrawFunds - снятие средств со счета**
* **ToPay – оплата различных услуг**

В нашем случае есть три различных ATM терминала: Сбербанк, ВТБ и Тинькофф. Все они реализую интерфейс IATMTerminal: внести наличные в кассету и зачислить денежные средства на счёт клиента, снять наличные из кассеты и выдать их клиенту или вызывать контроллер для оплаты услуги по выбору клиента. Каждый терминал имеет свои запчасти, из которых он состоит и у каждого класса они свои – у терминала Сбербанка: CassetteSberbank, ControllerSberbank, DispensarySberbank и PrinterSberbank. У ВТБ и Тинькофф по аналогии, но с другими названиями. Каждый компонент, в зависимости от того к какому терминалу он принадлежит, реализует общий интерфейс, но имеет различную логику внутри класса.

**7 СЛАЙД:**

**Стратегия - это поведенческий паттерн проектирования, который позволяет определить семейство алгоритмов, инкапсулировать каждый из них и сделать их взаимозаменяемыми. При этом позволяется изменять алгоритмы независимо от клиентского кода, который использует эти алгоритмы.**

**8 СЛАЙД:**

Наша реализация паттерна стратегия заключается в определении алгоритма нескольких родственных классов ATM терминалов, которые отличаются поведением и содержимыми компонентами. В зависимости от полученных во время создания экземпляра класса IATMTerminal, компонентов, паттерн стратегия определяет конкретный класс, который в дальнейшем предоставляет интерфейс взаимодействия для пользователя. В дальнейшем покажем эту реализацию в коде.

**9** **СЛАЙД:**

**Фасад - это структурный паттерн проектирования, который предоставляет простой интерфейс для сложной подсистемы, состоящей из множества классов. Фасад позволяет скрыть сложность взаимодействия между классами подсистемы и предоставить клиенту только необходимый функционал, что упрощает работу с системой и уменьшает зависимости между классами.**

**10 СЛАЙД:**

Наша реализация паттерна фасад заключается в уменьшении количества зависимостей между клиентом и сложной системой ATM терминала. Мы реализовали отдельный класс, который помогает изолировать компоненты системы от клиента и позволяет работать с ними независимо. В нашем случае эти компоненты это: Диспенсер, принтер, контроллер и кассета. Клиент не должен иметь напрямую к ним доступ, он должен реализовываться через взаимодействие клиента с классом ATMTerminal, который в свою очередь предоставляет удобный для клиента интерфейс и в котором скрыта вся сложная логика работы все компонентов вместе взятых.

**11 СЛАЙД:**

**Теперь о программе в целом**

**12 СЛАЙД:**

**Рассказываем про код…**