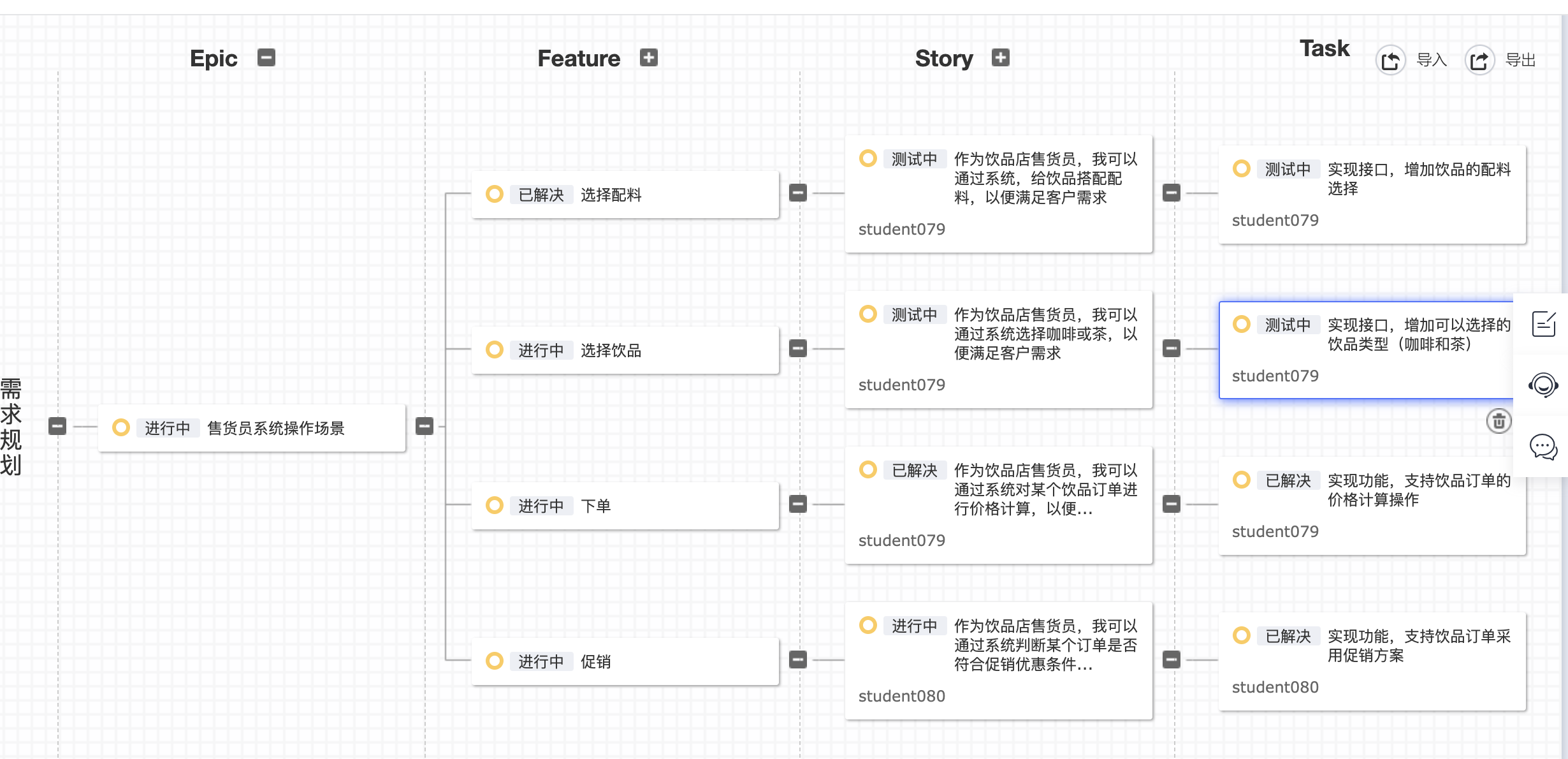
**软件工程 Lab4**

石睿欣 张逸涵 胡宵宵 宋怡景

一、阐述本次实验中工作项规划的设计思想



仔细研读文档需求，在讨论后我们达成了统一，先确认此次需求从业务角度来看都属于售货员系统操作的场景，同时按照需求的种类划分任务，确定新增的需求分属于选择配料，选择饮品，下单，以及促销这几个功能，所以我们将任务分成了实现增加饮品配料，增加可以选择的饮品类型，支持饮品订单价格计算操作，以及支持饮品订单采用促销方案的功能并对他们进行测试。

根据任务量，分工为一人实现增加饮品配料，增加可以选择的饮品类型，支持饮品订单价格计算操作，一人实现支持饮品订单采用促销方案的功能的架构设计，一人在架构下完成功能的实现，另一人测试本次代码。

二、阐述本次实验中代码实现的设计思想

**（1）新增饮料、配料与订单原价计算设计：**

1.增加饮品配料，增加可以选择的饮品类型：

为了使代码风格统一同时考虑到不同饮品类已经有了不同的特点，比如tea和coffee对应size不同，以后还可能具体饮品特点不同，所以仿照Lab3的结构，新增抽象类tea，及其继承子类RedTea和GreenTea，实现抽象函数cost并填配料类于entry包中，在repository里新增构造对应类的接口和实现，在data包中增添RedTea和GreenTea的csv配置文件，因为暂时而言，配料之间仅仅是名字不同，并无其他区别，所以并未将配料每个种类单独拿出来，而是当成一个属性，根据不同的名字得到取相应的配置文件，构建不同属性的对象。

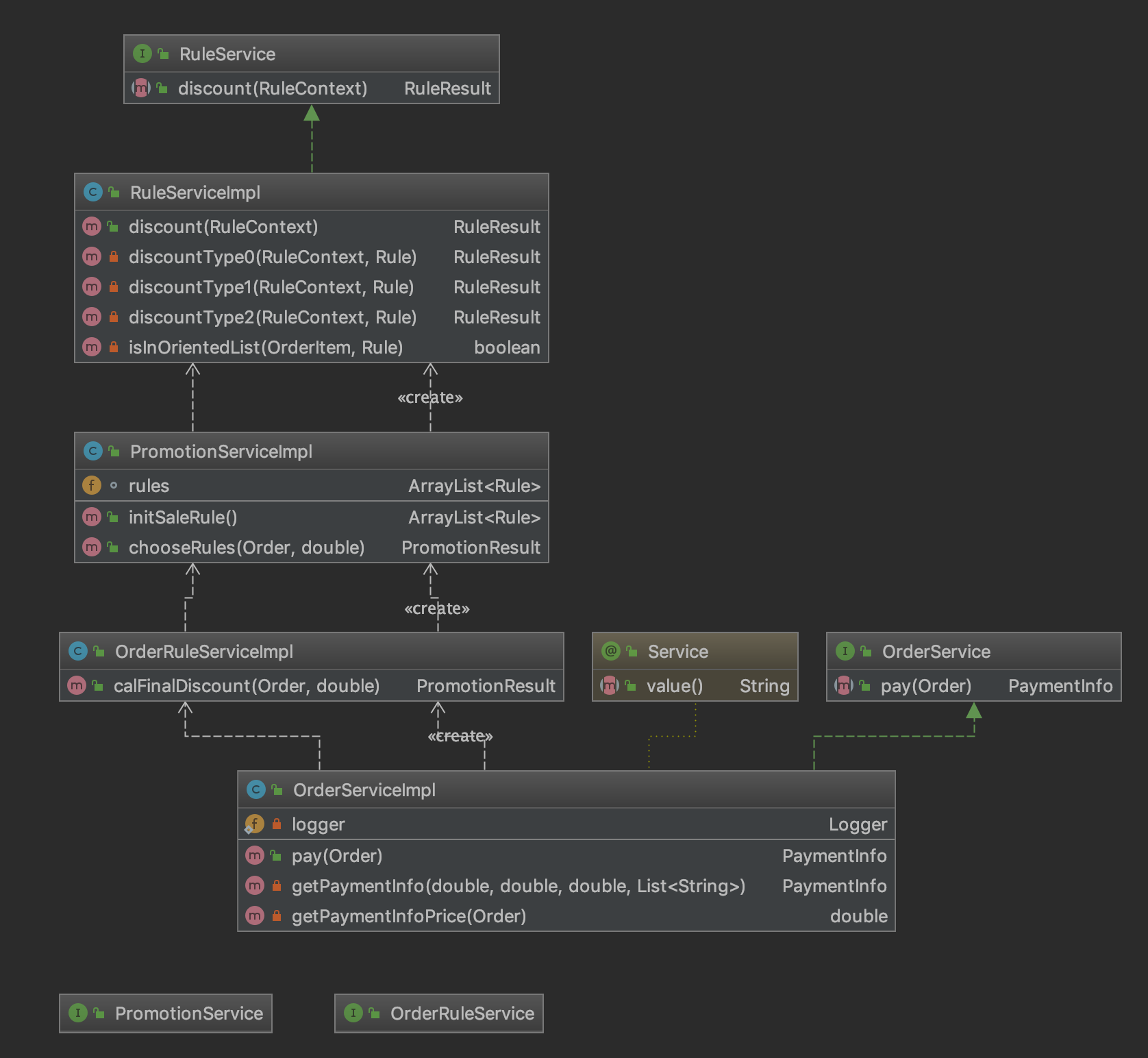
同时，Tea和Coffee又都属于饮品，有共同点，有共同的属性和行为，为了更好的可扩展性，将其抽象出drinks的抽象父类，有抽象函数cost。在DrinkUtil里面有函数getDrinks可以根据drink的名称构造相应的具体饮品对象。

2.支持饮品订单价格计算操作：

由助教给出的说明和计算公式，可以知道此价格计算也就是完成一个函数，遍历传入的order里面的orderItem，计算每杯的价格和，每杯的价格是饮品本身价格+饮品杯型+配料价格和，其中得到饮品本身价格和杯型价格是通过getDrinks用特定子类构造Drinks的类，利用多态，重载Drinks中的cost函数，而配料价格是遍历Ingredient，根据名字找到对应配置，得到Ingredient对象得到相应价格。返回结果。

**（2）促销方案的设计架构：**

在一个咖啡店里，促销政策可能有很多种，也可能随时间变化。为了应对促销规则的多变性，我们将促销设计成一个单独的模块。



1.使用门面模式，OrderRuleService是促销模块的唯一暴露接口，其余模块仅调用该接口来实现促销功能。

2.OrderRuleService去调用PromotionService来处理当前情况下促销方案的最优选择。之所以还要在PromotionService外面套一层门面，是为了可扩展性。比如除了促销活动，还可能有优惠券这种促销形式，之后扩展这部分的功能时，我们直接在OrderRuleService添加对优惠券处理接口的调用，之后在OrderRuleService里判断是选促销活动还是选优惠券即可。

3.无论是促销活动，还是优惠券，我们都可以剥离出它们共同的抽象本质：一条条促销规则。促销规则类型很多，如满减、满赠、打折等。一条规则就对应于文档中的一条语句，如“卡布奇诺第二杯半价”。因为本次lab不设计数据库中的表设计，我们先设计了rule对象来抽象促销规则的共性。

4.促销规则之间的关系：优先级、兼容、互斥。在此，我们对促销规则引入分组概念。就像lab4文档中的第一类优惠、第二类优惠一样，我们给促销规则标上组别，两个规则属于不同组代表它们是不同类优惠。组别的概念隐含着关系：同组优惠兼容，不同组优惠互斥。其实还有可能某一类优惠不与其他任何类优惠互斥，我们可以在后续需要时引入特殊组概念来解决这个需求。对于同组优惠兼容，我们引入优先级概念来解决分层的问题。比如满100减10，满200减25，满300减50这种分层结构，我们可以设置优先级，先去判断满300减50这条规则，再去判断满200减25这条规则。优先级和组别将被设计成rule对象的属性。

5.整体的调用逻辑：

OrderRuleService是门面，调用PromotionService里的chooseRules方法返回PromotionResult，PromotionResult里封装了对于当前订单选择的最优“优惠组合”方案信息。

RuleService是用来处理每一条促销规则的。RuleService里的discount方法，传入RuleContext上下文对象，并返回RuleResult对象。

PromotionService里对于数据库中（本次lab中内设好了rule对象）获得的每条促销规则调用一次RuleService,然后对于同组兼容的促销规则整合，不同组的优惠结果比较，选出最优的方案，封装成PromotionResult。（注：RuleContext对象封装了一些上下文信息：商品列表、原始价格、当前优惠规则等。RuleResult对象主要封装“使用该条规则，当前订单能获得的最佳优惠结果”信息。）

以下是一些对象的设计：（对象属性均为private,选择性地暴露他们的get、set接口。）

① Rule对象的设计：

int groupId:分组信息，当前规则在哪个组。

int scope:资格范围，该属性的设置方便扩展，因为促销规则也有一定的范围，如只有高级会员才能享受满100减10的优惠等。

ArrayList<Drinks> oriented:对象范围，因为促销规则有一定的适用对象。如“卡布奇诺第二杯半价”这个促销规则的适用对象是卡布奇诺。这一部分可以扩展为对象范围和对象属性。如对于一些sku属性（销售属性）进行设置：如只有白色的iPhone8才打折。

int profitType:利益类型，如0是满减，1是满赠，2是打折。

int condition:促销中需要满足的额度，如“满200减10”中200是额度。

int discountRange:参与优惠的范围。如“第二杯半价”和“两杯五折”是不一样的概念，前者是第二杯参与优惠，后者是两杯都参与优惠。

double profit：优惠，八折就是0.8，送一杯就是1。

ArrayList<Drinks> freeDrinks:如果送饮料，送的类型。如果不送，优惠为null。

boolean canAdd:该促销规则是否支持累加。如“满100减10”支持累加就说明“满200的时候可以减20”。

② RuleContext对象的设计：

Order order:可能用到促销规则的订单。

Rule rule：当前促销规则。

double purePrice：原始价格。

③ RuleResult对象的设计：

Rule rule：当前规则。

String ruleDescription：当前规则的描述，方便构建msg。

double discount：规则带来的优惠。

④ PromotionResult对象的设计：

List<String> promotionType:选择的促销优惠的类型。如我们把第一类优惠当作促销规则的第一组。如果我们的计算结果是第一类优惠的组合最优，这里返回的就是1.

double discount：最终的促销优惠。

**（3）测试设计：**

测试主要分为两大类：第一类是对订单本身的测试，第二类是对促销方案策略的选择的测试 。

1.第一类包括对订单本身、订单中饮品列表、饮品标识符、饮品杯型、配料列表、配料标识符和配料份数以及不同饮品组合的测试，具体测试如下：

订单为null：

|  |  |  |
| --- | --- | --- |
| 输入的订单信息 | 期待的输出 | 期待的log信息 |
| 订单为null | 运行时异常 | The order has something wrong. |

对订单中饮品列表的长度的测试：

|  |  |  |
| --- | --- | --- |
| 输入的订单信息 | 期待的输出 | 期待的log信息 |
| 饮品列表为null | 运行时异常 | The order has something wrong. |
| 饮品列表中的长度为0 | 运行时异常 | The order has something wrong. |
| 饮品列表中只有一小杯espresso | 原价=22，优惠=0，最终价格=22，优惠信息列表中无内容 | 无 |
| 饮品列表中有三小杯espresso | 原价=66，优惠=0，最终价格=66，优惠信息列表中无内容 | 无 |

对订单中饮品的name的测试：

|  |  |  |
| --- | --- | --- |
| 输入的订单信息 | 期待的输出 | 期待的log信息 |
| 饮品name为null | 运行时异常 | Failed to get drinks. |
| 饮品name错误 | 运行时异常 | Failed to get drinks. |
| 一杯饮品，name为espresso，无配料，小杯 | 原价=22，优惠=0，最终价格=22，优惠信息列表：[] | 无 |
| 一杯饮品，name为cappuccino，无配料，小杯 | 原价=24，优惠=0，最终价格=24，优惠信息列表：[] | 无 |
| 一杯饮品，name为redTea，无配料，小杯 | 原价=20，优惠=0，最终价格=20，优惠信息列表：[] | 无 |
| 一杯饮品，name为greenTea，无配料，小杯 | 原价=20，优惠=0，最终价格=20，优惠信息列表：[] | 无 |

对订单中饮品的杯型的测试：

|  |  |  |
| --- | --- | --- |
| 输入的订单信息 | 期待的输出 | 期待的log信息 |
| 饮品杯型小于1 | 运行时异常 | The size of coffee is invalid. |
| 饮品杯型大于3 | 运行时异常 | The size of coffee is invalid. |
| 一杯espresso，无配料，size为1 | 原价=22，优惠=0，最终价格=22，优惠信息列表：[] | 无 |
| 一杯espresso，无配料，size为2 | 原价=24，优惠=0，最终价格=24，优惠信息列表：[] | 无 |
| 一杯espresso，无配料，size为3 | 原价=26，优惠=0，最终价格=26，优惠信息列表：[] | 无 |
| 一杯redTea，无配料，size为1 | 原价=20，优惠=0，最终价格=20，优惠信息列表：[] | 无 |
| 一杯redTea，无配料，size为2 | 原价=22，优惠=0，最终价格=22，优惠信息列表：[] | 无 |
| 一杯redTea，无配料，size为3 | 原价=23，优惠=0，最终价格=23，优惠信息列表：[] | 无 |

对订单列表中不同饮品组合的测试：

|  |  |  |
| --- | --- | --- |
| 输入的订单信息 | 期待的输出 | 期待的log信息 |
| 一小杯cappuccino+一小杯espresso | 原价=46，优惠=0，最终价格=46，优惠信息列表：[] | 无 |
| 一小杯redTea+一小杯greenTea | 原价=38，优惠=0，最终价格=38，优惠信息列表：[] | 无 |
| 一小杯espresso+一小杯redTea | 原价=42，优惠=0，最终价格=42，优惠信息列表：[] | 无 |

对订单中配料列表的长度的测试：

|  |  |  |
| --- | --- | --- |
| 输入的订单信息 | 期待的输出 | 期待的log信息 |
| 配料列表为null | 运行时异常 | Ingredient is invalid. |
| 配料列表的内容为空，一小杯espresso | 原价=22，优惠=0，最终价格=22，优惠信息列表：[] | 无 |
| 配料列表中有一份份数为1的cream，一小杯espresso | 原价=23，优惠=0，最终价格=23，优惠信息列表：[] | 无 |
| 配料列表有三份份数为1的cream，一小杯espresso | 原价=25，优惠=0，最终价格=25，优惠信息列表：[] | 无 |

对配料name的测试：

|  |  |  |
| --- | --- | --- |
| 输入的订单信息 | 期待的输出 | 期待的log信息 |
| 配料name为null | 运行时异常 | Ingredient is invalid. |
| 配料name错误 | 运行时异常 | Ingredient is invalid. |
| 配料name为cream，一小杯加了一份cream的espresso | 原价=23，优惠=0，最终价格=23，优惠信息列表：[] | 无 |

对配料份数的测试：

|  |  |  |
| --- | --- | --- |
| 输入的订单信息 | 期待的输出 | 期待的log信息 |
| 配料cream份数为-1，一小杯espresso | 运行时异常 | Ingredient is invalid. |
| 配料cream份数为0，一小杯espresso | 原价=22，优惠=0，最终价格=22，优惠信息列表：[] | 无 |
| 配料cream份数为1，一小杯espresso | 原价=23，优惠=0，最终价格=23，优惠信息列表：[] | 无 |
| 配料cream份数为3，一小杯espresso | 原价=25，优惠=0，最终价格=25，优惠信息列表：[] | 无 |

对不同配料组合的测试：

|  |  |  |
| --- | --- | --- |
| 输入的订单信息 | 期待的输出 | 期待的log信息 |
| 配料列表为一份milk+一份cream，一小杯espresso | 原价=24.2，优惠=0，最终价格=24.2，优惠信息列表：[] | 无 |

2.第二类是对优惠策略选择的测试：

对组合优惠中espresso的优惠方式的测试：

|  |  |
| --- | --- |
| 输入的订单信息 | 期待的输出 |
| 只有一大杯espresso | 原价=26，优惠=0，最终价格=26，优惠信息列表：[] |
| 一大杯espresso+一小杯espresso | 原价=48，优惠=0，最终价格=48，优惠信息列表：[] |
| 两大杯espresso，其中一杯加入一份cream | 原价=53，优惠=8，最终价格=45，优惠信息列表：[“espresso: 每2杯,2杯8.0折”] |
| 三大杯espresso | 原价=79，优惠=8，最终价格=71，优惠信息列表：[“espresso: 每2杯,2杯8.0折”] |

对组合优惠中cappuccino的优惠方式的测试：

|  |  |
| --- | --- |
| 输入的订单信息 | 期待的输出 |
| 只有一小杯cappuccino | 原价=24，优惠=0，最终价格=24，优惠信息列表中无内容 |
| 一大杯cappuccino+一小杯cappuccino | 原价=52，优惠=11，最终价格=41，优惠信息列表：[“cappuccino: 每2杯,1杯5.0折”] |
| 两小杯cappuccino，其中一杯加入一份cream | 原价=49，优惠=11，最终价格=38，优惠信息列表：[“cappuccino: 每2杯,1杯5.0折”] |
| 三杯cappuccino，分别是小杯，中杯，大杯 | 原价=78，优惠=11，最终价格=67，优惠信息列表：[“cappuccino: 每2杯,1杯5.0折”] |

对组合优惠中红茶的优惠方式的测试：

|  |  |
| --- | --- |
| 输入的订单信息 | 期待的输出 |
| 三小杯红茶 | 原价=60，优惠=0，最终价格=60，优惠信息列表：[] |
| 四小杯红茶 | 原价=80，优惠=18，最终价格=62，优惠信息列表：[“redTea :买3杯送1杯”] |
| 九小杯红茶（此处测试九杯是因为5、6、7杯都会选择满100减30的优惠方式） | 原价=180，优惠=36，最终价格=144，优惠信息列表：[“redTea :买3杯送1杯”] |
| 四杯红茶，三杯是小杯，一杯为大杯 | 原价=83，优惠=18，最终价格=65，优惠信息列表：[“redTea :买3杯送1杯”] |
| 四小杯，其中一杯加入一份cream | 原价=81，优惠=18，最终价格=63，优惠信息列表：[“redTea :买3杯送1杯”] |

对组合优惠中绿茶的优惠方式的测试：

|  |  |
| --- | --- |
| 输入的订单信息 | 期待的输出 |
| 四小杯greenTea | 原价=72，优惠=16，最终价格=56，优惠信息列表：[“greenTea :买3杯送1杯”] |

对组合优惠中两种茶都有时的优惠方式的测试：

|  |  |
| --- | --- |
| 输入的订单信息 | 期待的输出 |
| 八小杯茶，其中一杯为红茶 | 原价=146，优惠=34，最终价格=112，优惠信息列表：[“redTea greenTea :买3杯送1杯”] |
| 八小杯茶，其中两杯为红茶 | 原价=148，优惠=36，最终价格=112，优惠信息列表：[“redTea greenTea :买3杯送1杯”] |
| 九小杯茶，其中一杯为红茶 | 原价=164，优惠=34，最终价格=130，优惠信息列表：[“redTea greenTea :买3杯送1杯”] |

对组合优惠中不同饮品优惠累加的测试

|  |  |
| --- | --- |
| 输入的订单信息 | 期待的输出 |
| 两大杯espresso+六小杯cappuccino | 原价=196，优惠=41，最终价格=155，优惠信息列表：[“espresso: 每2杯,2杯8.0折”，“cappuccino: 每2杯,1杯5.0折”] |
| 八小杯greenTea+两小杯cappuccino | 原价=192，优惠=43，最终价格=149，优惠信息列表：[ “greenTea :买3杯送1杯”，“cappuccino: 每2杯,1杯5.0折”] |
| 八小杯greenTea+两大杯espresso | 原价=196，优惠=40，最终价格=156，优惠信息列表：[“espresso: 每2杯,2杯8.0折”，“greenTea :买3杯送1杯”] |
| 两大杯espresso+4小杯greenTea+2小杯cappuccino | 原价=172，优惠=35，最终价格=137，优惠信息列表：[“espresso: 每2杯,2杯8.0折”，“greenTea :买3杯送1杯”，“cappuccino: 每2杯,1杯5.0折”] |

对满100减30的优惠方式的测试：

|  |  |
| --- | --- |
| 输入的订单信息 | 期待的输出 |
| 一小杯espresso | 原价=22，优惠=0，最终价格=22，优惠信息列表：[] |
| 一小杯espresso+一中杯espresso+三小杯greenTea | 原价=100，优惠=30，最终价格=70，优惠信息列表：[“满100减30.0”] |
| 五小杯espresso | 原价=110，优惠=30，最终价格=80，优惠信息列表：[“满100减30.0”] |

对满100减30和组合优惠两种优惠方式都有时最优优惠策略选择的测试：

|  |  |
| --- | --- |
| 输入的订单信息 | 期待的输出 |
| 五小杯redTea | 原价=100，优惠=30，最终价格=70，优惠信息列表：[“满100减30.0”] |
| 七小杯greenTea+一小杯redTea | 原价=146，优惠=34，最终价格=112，优惠信息列表：[“redTea greenTea :买3杯送1杯”] |
| 四小杯cappuccino+两大杯espresso | 原价=148，优惠=30，最终价格=118，优惠信息列表：[“espresso: 每2杯,2杯8.0折”，“cappuccino: 每2杯,1杯5.0折”] |

三、阐述本次实验中对需求变更和项目管理（项目规划、缺陷管理等）的认识

本次lab的需求增加了饮品类型，加入了配料选择和优惠政策，对比前几次的lab，这几个需求需要代码的结构具有更高的拓展型和易改性。并且，本次lab的实现的自由度更加高了，助教并没有给我们提供实现需求的具体架构，所以在完成本次lab时，我们在写代码之前对于自己代码的架构设计也变得重要起来。为了增加可拓展性，使得以后可以方便的新加别的配料，修改优惠方式，我们非常努力的设计了一个目前我们非常满意的自认为具有很高拓展性的架构设计，具体也在文档的第二点中详细阐述了。

就项目管理而言，本次的项目规划也是由我们自己设计的，让我们在写代码前更加了解了整个lab的需求，具体表现为规划好了要如何写，谁来写，具体会有哪些逻辑等。比起前几次lab中助教直接给出项目规划，我们会就task的理解不清多次询问助教，这个状况现在也得到了避免。

而缺陷管理，使得我们在分组合作中的交流更加明确，在相互的代码审计以及测试中，如果发现了逻辑问题或者bug，直接写在缺陷管理中，就完全不会因为群内微信消息太多而遗漏，并且也给了写代码的人更加清晰的修改目标，提高了合作的效率。

四、阐述实验中遇到的问题与解决方案

1. espresso和cappuccino的csv文件中的价格问题：在lab3中，cappuccino的基础价格为$20，espresso的基础价格为$22，而在本次lab中刚好反过来，这属于是外部数据库的错误，通过修改csv文件解决。
2. 日志机制的补充与完善：在第一次的简单版本的开发中，本组成员只对极少部分的异常情况做了日志记录，根据上课所讲内容，对所有的异常信息都做了详细的日志记录。
3. 对配料份数为负数的情况：本组成员在开发中，没有考虑到配料份数小于0的情况，因此，会出现加入-1份的配料时，总价小于饮品本身原价的异常情况，后来通过加入对传入的参数是否合法的检测而解决。
4. discount问题：本组成员开发中，将discount算反了，对于打八折的espresso，其discount应当为基础价格的0.2倍，本组成员却算成了0.8倍。
5. 饮品name问题：因为该lab是小组成员合作完成，所以开发中出现了使用饮品name不统一的情况，在通过饮品name得到对应饮品实例的工具类中，使用的是“cappuccino”、“espresso”、“greenTea” 和 “redTea”，但是在一个计算优惠方案的sevice类中，使用的却是首字母大写的饮品name，这种不统一导致在计算优惠条件的代码中有一个逻辑无法正确返回，该问题由修改name使其统一解决。

6. 因为我们项目的架构设计具有较高的拓展性，为了使组员在之后的lab中能够继续在此架构上进行拓展操作而不产生遗忘和不理解，再加上考虑到随着lab的增加每人所完成过的功能也变得更多，我们为项目撰写了wiki文档，便于全体组员对项目的理解。

Wiki链接（可以从代码仓库页直接看见）<https://devcloud.huaweicloud.com/wiki/project/3369792d00704d829c147847cf4b7a1c/wiki/view/doc/314954>

五、小组分工

小组分工

|  |  |  |  |
| --- | --- | --- | --- |
| achillessanger | 石睿欣 | 17302010065 | 优惠方案计算代码实现 |
| HXX5656 | 胡宵宵 | 17302010077 | 优惠方案代码架构设计 |
| song | 宋怡景 | 17302010079 | 测试 |
| zyhlx | 张逸涵 | 17302010076 | 增加饮品、配料、计算优惠前价格代码实现 |