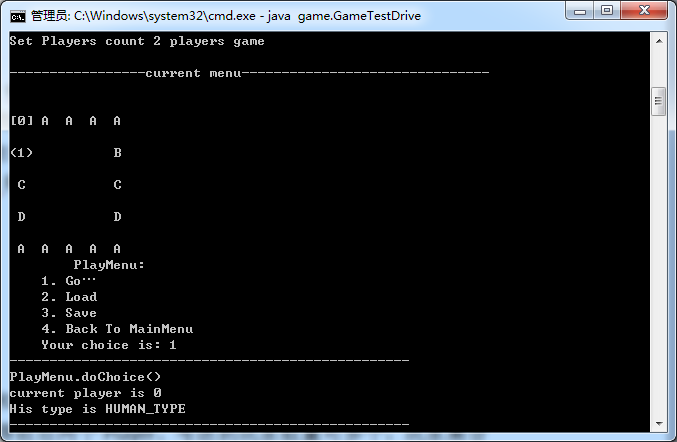
作业05 行为型+其它(以step0\_src中代码为基础)

1. 将玩家改为只有两个Player，一个HumanPlayer，一个AutoPlayer。初始时，各玩家都站在map中的第一个Block上，初始的移动方向任意。每轮次HumanPlayer玩家前进3格,AutoPlayer前进2格。前进方向任意，但若前方有格子(前进、左转、右转)，必定前进，而不能往回走。HumanPlayer所在的格子用[X]表示，AutoPlayer的格子用(X)表示。请实现代码。

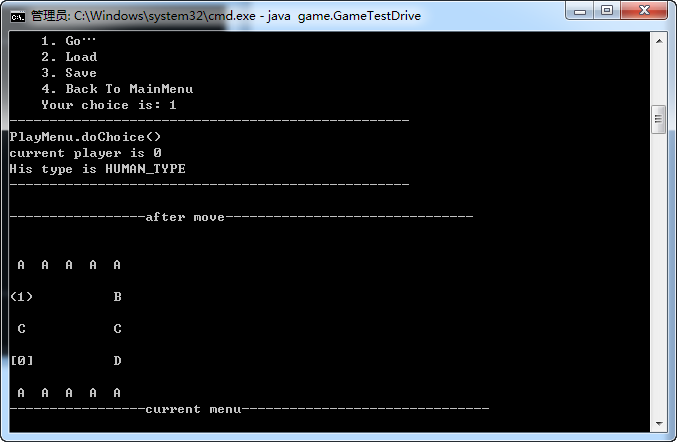
**-----------------------------------------------------答案分割线-----------------------------------------------------------**

1. 初始化 HumanPlayer 为 (0, 0)处，而 AutoPlayer 为 (1, 0)处。逆时针方向行走。

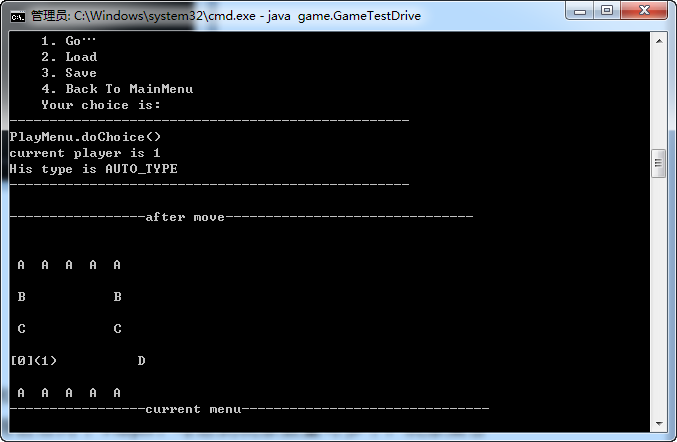
将 [X] 中的 X 改为 Player 的 ID。



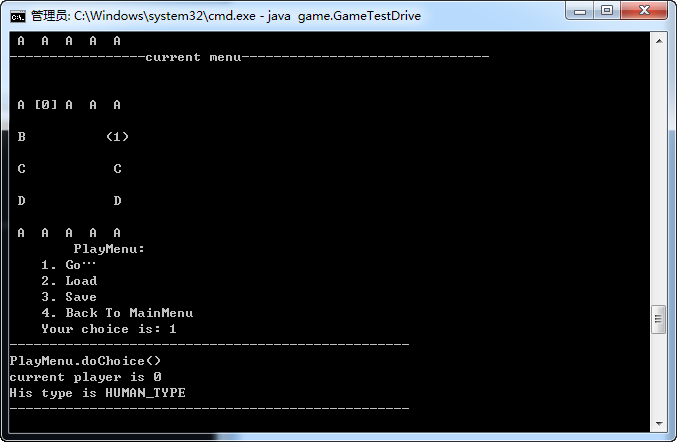
（2）HumanPlayer 行走 3 格



1. AutoPlayer 行走两格



1. 循环

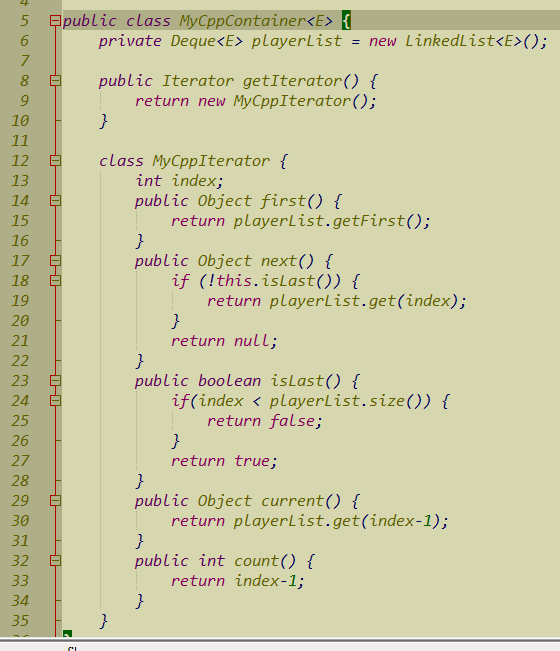


1. 在Players的代码中,通过getFirst和getLast取得两个玩家，但这两个成员函数与Player用deque表示紧密相关,且只能取得前后两个Player。考虑到玩家数量可多个，玩家集合的数据结构可多样，请用迭代器模式重新实现遍历所有玩家的功能，要求分别实现C++风格的外部迭代器和Java风格的外部迭代器。迭代器接口如下：

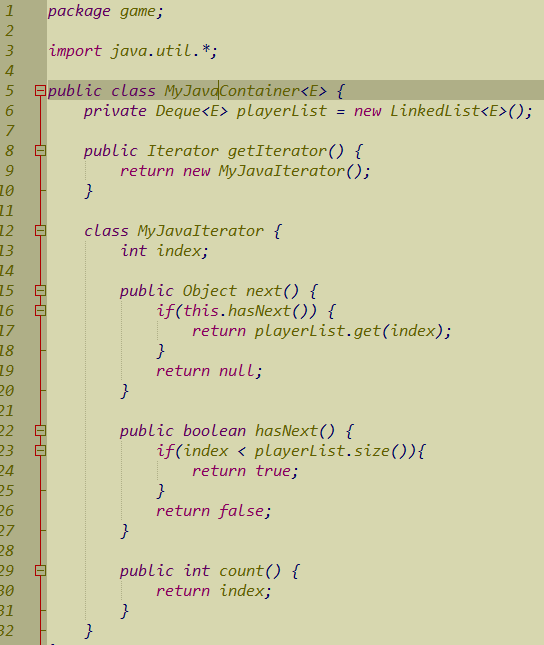
|  |  |
| --- | --- |
| class CppIterator {  public:  CppIterator( ) {}  virtual ~CppIterator( ) {}  virtual void first() = 0;  virtual void next() = 0;  virtual bool isLast() const = 0;  virtual void \* current() = 0;  virtual int count() const=0;  }; | class JavaIterator {  public:  JavaIterator() {}  virtual ~JavaIterator() {}  virtual void\* next() = 0;  virtual bool hasNext() const = 0;  virtual int count() const=0;  }; |

**-----------------------------------------------------答案分割线-----------------------------------------------------------**

Cpp Iterator 实现



Java Iterator 实现



两者实现可以在 Monopoly-Q2特别版\src\game\ 目录下找到

1. 若玩家前进时，各玩家的前进策略多种多样，如一种策略是各方向是等概率的（例如1中的代码），另一种策略是各方向有不同几率：inDir未确定时，向各方向前进的概率相等；inDir确定时，各方向的概率不一定相同。具体可见下表。而且以后可能会添加新的前进策略，如具有AI的策略等；玩家还可能需要动态改变前进策略。请使用策略模式实现。

|  |  |  |
| --- | --- | --- |
|  | inDir==NONE时 | inDir!=NONE时 |
| 连通四个方向 | 各方向各25% | 直行50%,左转25%，右转25% |
| 连通往三方向 | 各方向各33% | 可直行时，直行60%，左转(或右转)40%；  不可直行时，左转和右转各50% |
| 连通两个方向 | 各方向各50% | 与inDir不同的方向为100% |
| 连通一个方向 | 各方向各100% | 100%地转向往回走 |

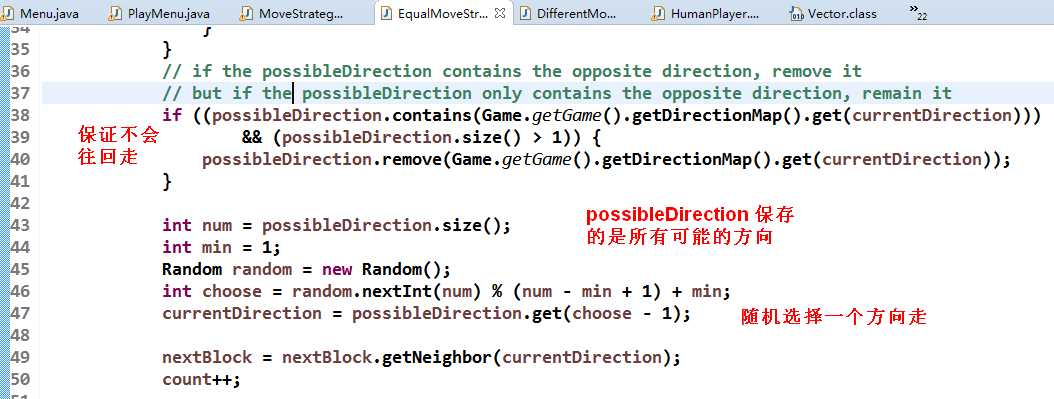
**-----------------------------------------------------答案分割线-----------------------------------------------------------**

以下 [0] 代表 HumanPlayer，[1] 代表 AutoPlayer

由于我不太懂 indir 代表的含义，故只是分开实现两个策略。

1. 对于 inDir == none 时，使用 EqualMoveStrategy；
2. 对于 inDir != none 使用 DifferentMoveStrategy

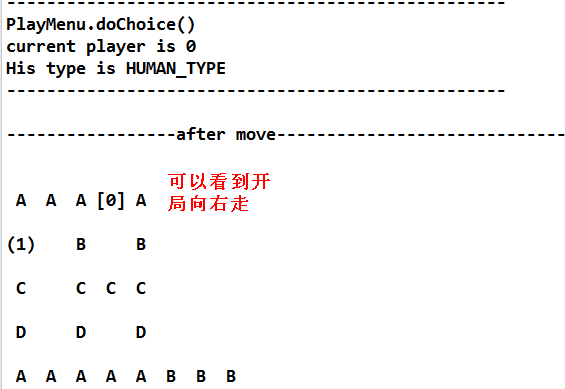
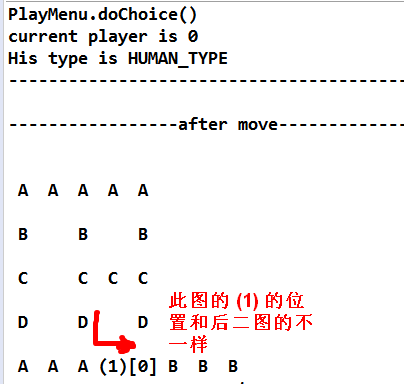
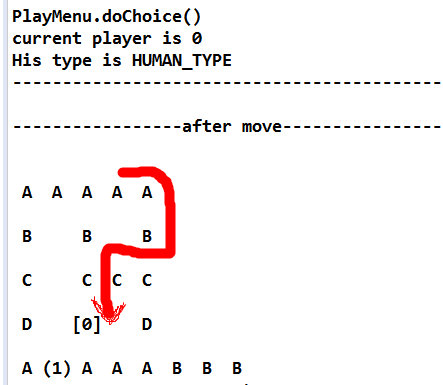
EqualMoveStrategy 代码核心部分为：

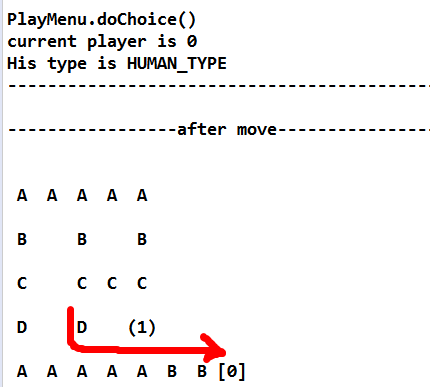
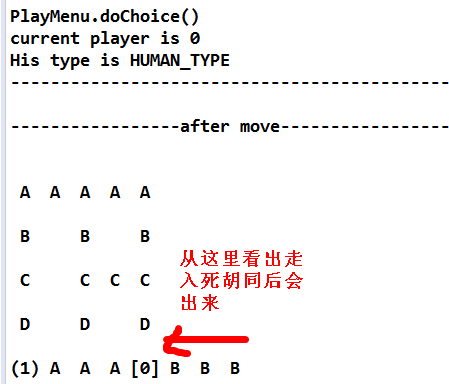
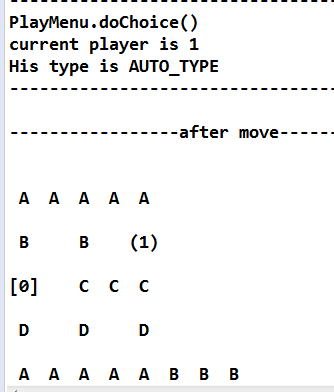


DifferentMoveStrategy 核心代码如下



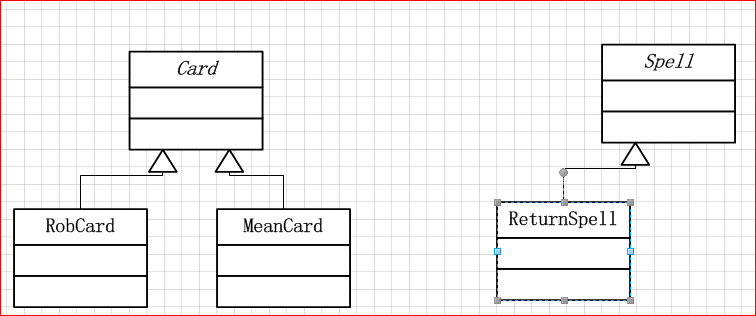
EqualMoveStrategy 测试如下，我的判断标准是能遍历地图上的任意一个地方：

走入死胡同会出来是在代码中的 possibleDirection.size()>1 这句体现出来的

1. 为提高趣味性,增加了卡片子系统和魔法子系统, Player可以使用卡片、魔法等，作用于各玩家或系统中的其它对象上。部分类图如下:



这里使用RobCard会抢走指定玩家的一半金钱；使用MeanCard会平均所有玩家的金钱；使用ReturnSpell会使指定玩家直接返回到开始的Block，inDir不定。

1. 修改PlayMenu为：

PlayMenu

1. Go…
2. ApplyRobCard
3. ApplyMeanCard
4. ApplyReturnSpell
5. Load
6. Save
7. Back To MainMenu

Your selected:

并给Player类添加行为ApplyRobCard(Player \* target);、ApplyMeanCard()；和ApplyReturnSpell(Player \* target)，然后实现。

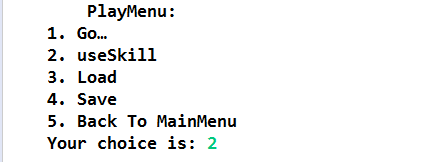
1. 以后肯定要添加其它Card和Spell，还有可能增加道具子系统等。由于Player已有子类AutoPlayer和HumanPlayer，而且未来扩展时Card、Spell和道具等时，作用目标target有多种可能，如Player、Block、Player集合，甚至Card类等，所以希望在未来扩展时，避免通过继承Player类或其子类进行扩展。请使用命令模式实现此目地。

**-----------------------------------------------------答案分割线-----------------------------------------------------------**

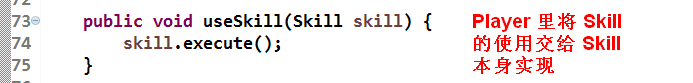
将 Spell 改成 通俗的 Magic

同时使用 Skill 来作为 Card 和 Magic 的基类，方便管理

（1）



（2）



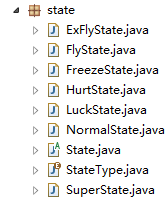
（3）

每个 Card 可以重复使用，因此使用 CardMgr 方便管理，避免重复构造相同 Card

Magic 同理

1. 若需要增加FlyCard和FreezeSpell。其作用是：当应用FlyCard到某个Player上时，该player将每轮次前进6格，并持续2个轮次；当施加FreezeSpell到某个Player上时，该player将每轮次只前进1格，并持续3个轮次。 考虑到以后还会添加其它类似功能的Card和Spell，但不希望直接从Player类派生子类进行扩展。请使用状态模式实现此目的。

**-----------------------------------------------------答案分割线-----------------------------------------------------------**



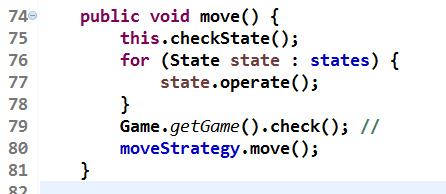


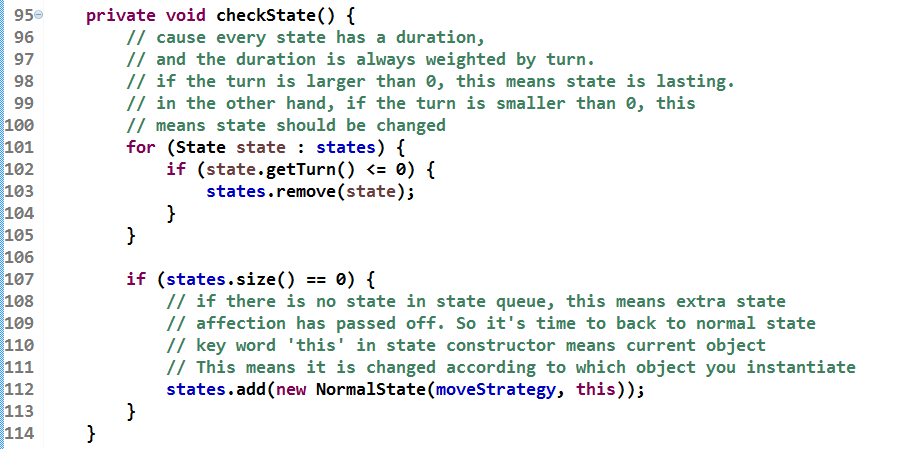
1. 若还需要增加HurtSpell,该Spell会导致被作用的Player,每轮次减少Money的数量5,并持续2个轮次;增加LuckCard, 该Card会导致被作用的Player,保持Luck状态3轮次,在Luck状态时,每轮次还额外增加200金钱;增加ExFlyCard, 除了FlyCard的效果 ,同时在前进时,不受任何Block的阻挡(如BarBlock); 增加SuperCard,其作用为同时有FlyCard和LuckCard的效果,并持续1回合。多个Card或Spell的效果也可能会同时存在，如LuckCard和HurtSpell。请实现.

**-----------------------------------------------------答案分割线-----------------------------------------------------------**

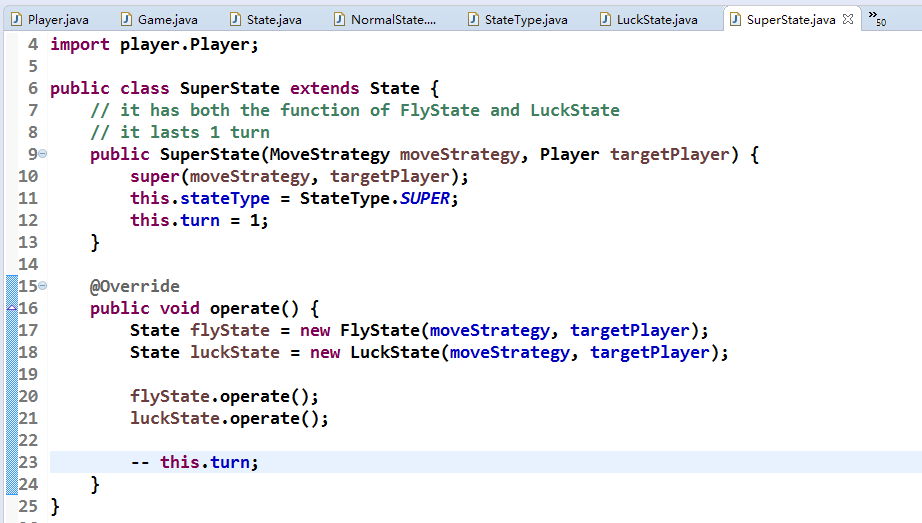
1. 多个 Card 或 Spell 同时存在是通过 State 队列来实现的。

在 Player.move() 中，Player 移动之前判断自身状态，移除已经执行完毕的状态。然后执行未完成的状态。最后才执行移动操作





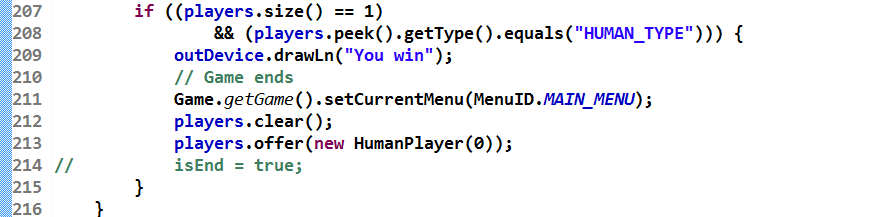
1. 对于一个 Card 可能拥有多个 Skill 的效果，实现就是构造多个 Skill 对应的状态，并执行对应状态的 operate() 函数



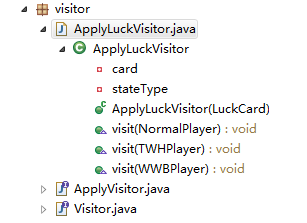
（3）对于 ExFlyCard 这种涉及到 Block 的状态，目前是给 State 添加了一个 boolean 属性 isInvinsible 表示受不受其他东西的影响

1. 若某个Player的money小于等于0，则表示该player破产了，需退出游戏过程，其它玩家继续；此时，所有继续游戏的玩家Money会减半，所有TripBlock的惩罚会变成200；若HumanPlayer都已破产，游戏返回mainmenu；若AutoPlayer都破产，且只剩一个HumanPlayer，显示胜利，返回mainmenu。

**-----------------------------------------------------答案分割线-----------------------------------------------------------**

1. 若各Player又分两种高富帅和白富美等有限几种。应用不同Card、Spell等到不同类型的Player时，效果可有不同效果, 如LuckCard应用到高富帅上，保持Luck状态3轮次；应用到白富美上，保持Luck状态4轮次等。已知Card、Spell等未来会增加和扩展。请以访问者模式设计并实现。



ApplyLuckVisitor 实现如下



