ALTARIX

Тестовое задание

Реализация игрового процесса карточной игры “Преферанс”

СОДЕРЖАНИЕ

1. Введение

Необходимо реализовать игровой процесс карточной игры “Преферанс” – инициализацию игры с участием 3-х игроков и сдающего карты, процесс торговли, открытие прикупа, сброс карт, заявление игры (6 пик, 7 треф, мизер, распасы и т.п.), розыгрыш, фиксирование результатов в пульке (пуля, гора, висты).

Реализация игрового процесса не предполагает наличие графического интерфейса, интерактивного взаимодействия с пользователями и хранения результатов в базе данных.

В рамках реализации игрового процесса необходимо спроектировать все нужные сущности предметной области и их API. Главными участниками игрового процесса являются игроки-боты, которые в автоматическом режиме осуществляют игровой процесс после инициализации игры.

В процессе игры боты должны безусловно следовать правилам, которые могут быть, как жестко предопределены, так и задаваться в процессе инициализации игры. Безусловно, базовые правила должны быть жестко заданы – старшинство мастей, выкладывание карты в масть или, если таковой нет, то козыря, порядок ходов. Вариации могут касаться стоимости игр, правил розыгрыша распасов, определенных правил торговли – см. конвенции (набор правил) “Ростов”, “Сочи”, “Ленинград”.

Все действия в процессе игры должны логироваться, как минимум, в консоль (дополнительное логирование в файл будет плюсом). Под действиями понимаются:

- Инициализация игры и условия, на каких она играется (10 раздач, до 10 в пуле, и т.п.), имена игроков-ботов
- Номер раздачи и результат раздачи карт (у кого какие карты и прикуп)
- Процесс торговли
- Прикуп и сброс,
- Финальная заявка игрока, действия других игроков (вист/пас – игра в открытую/закрытую)
- Розыгрыш (порядок ходов)
- Фиксирование результатов розыгрыша

Помимо логирования действий, все указанные выше данные должны сохраняться в специально спроектированных для этого структурах данных внутри приложения и храниться в оперативной памяти во время работы приложения.

После окончания игры приложение должно быть способно выдать результаты для определенных запросов, например:

- Сохранить в файл/распечатать результаты третьей раздачи карт

- Сохранить в файл/распечатать процесс торговли в пятом розыгрыше
- Сохранить в файл/распечатать процесс 7-го розыгрыша
- Сохранить в файл/распечатать результаты второго розыгрыша
- Сохранить в файл/распечатать полный процесс 6-розыгрыша (раздача, торговля, заявка, игра, результаты).
- Сохранить в файл/распечатать результаты всех розыгрышей.
- Сохранить в файл/распечатать финальные результаты игры.

Вариации этих запросов должны быть прописаны в коде в виде вызовов соответствующих методов, после окончания игры.

Качество игры ботов не так важно – главное, чтобы они играли по правилам, но наличие определенной стратегии игры/торговли/заявки игры бота будет плюсом. Параметризация стратегии при инициализации бота – еще большим плюсом.

Игра может быть написана на любом языке программирования, но реализация на Java предпочтительнее.

Исходный код проекта должен быть размещен на GitHub.

Вместе с исходным кодом должна быть предоставлена UML-диаграмма классов предметной области, используемых в игре.

Если по каким-либо причинам нет возможности полностью реализовать требования к приложению в срок, то можно предоставить приложение в незаконченном виде, с указанием причин, по которым приложение не удалось реализовать полностью.

Для ознакомления с правилами игры можно использовать, как общедоступные источники в интернете, так и бесплатные варианты игры “Преферанс” для персональных компьютеров или мобильных устройств.

P.S. При проектировании и разработке игры могут пригодиться знания таких паттернов проектирования, как Builder и Strategy. При распределении обязанностей между классами (в каком классе какие методы реализовать) могут пригодиться знания GRASP-паттернов.

2. Требования к системе

2.1. Общие требования

ID	Наименование
R1	После запуска, игра должна осуществляться в автоматическом режиме с участием трех игроков-ботов, в консольном режиме.
R2	Параметры окончания игры должны задаваться при ее инициализации. Условием окончания игры может быть указание количества раздач, размер пули (на усмотрение разработчика).
R3	Все действия в процессе игры должны логироваться. Как минимум – в консоль. В идеале – должно быть дополнительное логирование в отдельный файл.
R4	Основные правила игры должны быть жестко прописаны внутри приложения (старшинство мастей, правила торговли, выкладывание карт в масть при розыгрыше)
R5	Выбор конвенций (Сочи, Ленинград, Ростов) – на усмотрение разработчика. Параметризация конвенции при инициализации игры будет плюсом.
R6	Игрок-бот должен быть инициализирован, как минимум, каким-нибудь именем. Параметризация стратегии игры игрока-бота будет плюсом (например, агрессивная/нормальная/осторожная).
R7	Спроектированные сущности и их взаимосвязи должны быть предоставлены в виде UML-диаграммы классов.
R8	Игра может быть написана на любом языке программирования (реализация на Java будет плюсом).
R9	Исходный код проекта должен быть размещен на GitHub.

2.2. Требования к API

В рамках реализации приложения должны быть реализованы следующие методы, которые должны вызываться сразу после окончания игрового процесса:

ID	Наименование
API1	Метод получения данных определенной раздачи (кому какие карты были розданы, что в прикупе, кто начинает торговлю/чей ход). Номер раздачи должен быть вынесен в параметры метода. Результаты работы метода

	должны быть выданы на экран/записаны в файл.
API2	Метод получения данных о процессе торговли для определенной раздачи (включая данные о полученном прикупе). Номер раздачи должен быть вынесен в параметры метода. Результаты работы метода должны быть выданы на экран/записаны в файл
API3	Метод получения данных о процессе заявки игрока (какую игру заказал для определенной раздачи) и реакции других игроков (вист/паст, игра в открытую/закрытую). Номер раздачи должен быть вынесен в параметры метода. Результаты работы метода должны быть выданы на экран/записаны в файл
API4	Метод получения данных о процессе розыгрыша (последовательность ходов и принадлежность взяток) определенной раздачи. Номер раздачи должен быть вынесен в параметры метода. Результаты работы метода должны быть выданы на экран/записаны в файл
API5	Метод получения данных о результатах розыгрыша определенной раздачи (кто сколько взяток взял, какие цифры записаны в пулю, гору и висты). Номер раздачи должен быть вынесен в параметры метода. Результаты работы метода должны быть выданы на экран/записаны в файл
API6	Метод получения данных о полном процессе розыгрыша определенной раздачи (раздача, торговля, заявка, игра, результаты). Номер раздачи должен быть вынесен в параметры метода. Результаты работы метода должны быть выданы на экран/записаны в файл
API7	Метод получения данных о текущем состоянии пули, горы и вистах игрока после определенной раздачи. Номер раздачи и идентификатор игрока должны быть вынесены в параметры метода. Результаты работы метода должны быть выданы на экран/записаны в файл
API8	Метод получения промежуточного результата игрока после определенной раздачи. Результат работы метода – определенное число, показывающее как успешно играет игрок (определяется исходя из состояния пули, горы и вистов всех игроков, как если бы игра была прекращена после указанной раздачи и надо было бы подсчитывать результаты). Номер раздачи и идентификатор игрока должны быть вынесены в параметры метода. Результаты работы метода должны быть выданы на экран/записаны в файл
API9	Метод получения статистики по игроку после розыгрыша определенной раздачи – сколько и каких игр было сыграно, сколько успешных распасов, мизеров. Номер раздачи и идентификатор игрока должны быть вынесены в параметры метода. Результаты работы метода должны быть выданы на экран/записаны в файл
API10	Метод получения промежуточного результата всех игроков после определенной раздачи. Результат работы метода – определенное число, показывающее, как успешно играют игроки (определяется исходя из состояния пули, горы и вистов всех игроков, как если бы игра была прекращена после указанной раздачи и надо было бы подсчитывать

	результаты). Номер раздачи должны быть вынесены в параметры метода. Результаты работы метода должны быть выданы на экран/записаны в файл

2.3. Рабочие сценарии

Каждый сценарий представляет собой законченный вариант работы программы (main-класс в Java), в котором вызываются и отрабатывают все нужные методы.

ID	Наименование
S1	Проинициализировать и запустить игру на 12 раздач. После окончания игры: - вызвать метод API1 для 2-ой и 5-ой раздачи. - вызвать метод API2 для 4-ой раздачи. - вызвать метод API3 для 3-ей раздачи. - вызвать метод API4 для 7-ей раздачи. - вызвать метод API5 для 6-ой раздачи. - вызвать метод API7 для 8-ой раздачи и 1-го игрока. - вызвать метод API8 для 10-ой раздачи и 2-го игрока. - вызвать метод API9 для 10-ой раздачи и 3-го игрока. - вызвать метод API10 после окончания игры.
S2	Проинициализировать и запустить игру на 12 раздач. После окончания игры: - вызвать метод API6 последовательно для всех раздач.