МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №3

по дисциплине «ООП»

Тема: Связывание классов

Студент гр. 3385 Першин А.А.

Преподаватель Жангиров Т.Р.

Санкт-Петербург

Цель работы

Связать созданные классы в игровой процесс.

Задание

Создать класс игры, который реализует следующий игровой цикл:

- Начало игры
- Раунд, в котором чередуются ходы пользователя и компьютерного врага. В свой ход пользователь может применить способность и выполняет атаку. Компьютерный враг только наносит атаку.
- В случае проигрыша пользователь начинает новую игру
- В случае победы в раунде, начинается следующий раунд, причем состояние поля и способностей пользователя переносятся.

Класс игры должен содержать методы управления игрой, начало новой игры, выполнить ход, и т.д., чтобы в следующей лаб. работе можно было выполнять управление исходя из ввода игрока.

Реализовать класс состояния игры, и переопределить операторы ввода и вывода в поток для состояния игры. Реализовать сохранение и загрузку игры. Сохраняться и загружаться можно в любой момент, когда у пользователя приоритет в игре. Должна быть возможность загружать сохранение после перезапуска всей программы.

Примечание:

Класс игры может знать о игровых сущностях, но не наоборот

Игровые сущности не должны сами порождать объекты состояния

Для управления самой игрой можно использовать обертки над командами

При работе с файлом используйте идиому RAII.

Выполнение работы

1. Класс GameState

Назначение: Класс, отвечающий за хранение всех важных для игры объектов, которые могут быть использованы для вывода состояния в файл сохранения. Содержит переопределённые операторы ввода и вывода в поток.

Связи: Создает менеджер ввода для способностей, поля и менеджеры кораблей игрока и компьютера.

Поля и методы:

public:

Field& getPlayerField() const; - геттеры

Field& getBotField() const;

ShipManager& getPlayerShipManager() const;

ShipManager& getBotShipManager() const;

SpellManager& getSpellManager() const;

void shipPlacementPhase(); - Фазы игры

void spellCastPhase();

void shootCellPhase();

void botTurnPhase();

void savePhase() const;

void loadPhase();

void printFields(); - вывод полей.

v

b

bool win();

privated

Input* input;

int fieldWidth;

int fieldHeight;

Field* playerField;

Field* botField;

проверки на победу / поражение игрока

t

e

ShipManager* playerShipManager;

ShipManager* botShipManager;

SpellManager* spellManager;

std::ostream& operator<<(std::ostream& os, const GameState& gameState);

std::istream& operator>>(std::istream& is, GameState& gameState);

Класс Game

Назначение: класс, управляющий игрой и изменяющий состояния в соответствии с фазами.

Связи: хранит указатель на объект класса gameState.

Поля и методы:

public:

void start();

private

GameState* gameState

Выводы

Реализовано связывание классов, образующее непрерывный процесс игры по заданию.