**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №4**

**по дисциплине «Объектно-ориентированное программирование»**

Тема: Полиморфизм

|  |  |  |
| --- | --- | --- |
| Студент гр. 8303 |  | Логинов Е.А. |
| Преподаватель |  | Филатов А.Ю. |

Санкт-Петербург

2020

**Цель работы.**

Реализовать набор классов, для ведения логирования действий и состояний программы.

**Ход работы.**

1. Была добавлена система логирования действий пользователя, базы и юнитов. Для этого был создан класс LoggerProxy (файлы LoggerProxy.h/.cpp), вызывающий два метода:

printConsole(log) – вывод логов в консоль

printFile(log) – вывод логов в файл

Данные методы реализованы в классе LogSwitch (файл LogSwitch.h)

Метод printFile() взаимодействует с классом FileRAII(файл FileRAII.h), который работает на основе идиомы RAII. Конструктор FileRAII() принимает на вход имя файла, в который будет производиться запись.

Идиома RAII заключается в том, что получение потока std::ofstream записи в файл мы неразрывно связываем с созданием экземпляра класса FileRAII. При уничтожении объекта этот поток закрывается.

1. Для реализации разных видов логирования используется паттерн «Адаптер». За это отвечает класс Adapter (файл Adapter.h/cpp).

Класс адаптер наследуется от прокси-логгера LoggerProxy (LoggerProxy.h) и содержит множество методов для реализации записи различных форматов. Каждый метод принимает различные параметры для логирования той или иной ситуация, после чего адаптер с помощью соответствующих своих методов формирует строку для вывода в лог и вызывает метод print родительского класса LoggerProxy.

Экземпляр этого класса Adapter в «Фасаде» (файл Facade.h/cpp), после этого указатель на него передается в конструктор класса поле (файл Field.cpp). Класс поля содержит в себе указатель на «Адаптер» в своем конструкторе, который инициализирует его на логировании действий пользователя. Для создания лога поля и других методов через указатель на класс Adapter вызывается соответствующий метод класса Adapter и в этом метод передается информация, необходимая для логирования.

Для логирования состояний в классе Adapter был перегружен оператор вывода в поток. Было реализовано три метода перегрузки, для вывода состояния базы, юнита и поля. Перегруженные методы возвращают поток с информацией юнита, базы или поля. Эти перегрузки используются в методах вывода информации.

1. Переключение между типами логирования (файл, консоль или без логирования) реализовано в классе LoggerProxy (файл LoggerProxy.h).

Класс LoggerProxy наследуется от класса LogSwitch(файл LogSwitch.h). LoggerProxy содержит метод void print(std::string log, bool consoleWrite, bool fileWrite), который при помощи переменных consoleWrite(вывод логов в консоль) и fileWrite(вывод логов в файл), передаются на вход из метода switchLog() класса Adapter).

Пользователь может изменить параметры логирования командой "switchLog", в паттерне "Цепочка обазанностей" класса chainOfResponsibility (находится в файле chainOfResponsibility.h) обработается строка, изменятся переменные адаптера, которые будут преданы в LoggerProxy и будет определен тип записи логов.

**Выводы.**

В ходе выполнения лабораторной работы был реализован набор классов, для ведения логирования действий пользователя и состояний программы.