**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

отчет

**по лабораторной работе №4**

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

Тема: Умные указатели

|  |  |  |
| --- | --- | --- |
| Студент гр. 7303 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Ермолаев Д.В. |
| Преподаватель | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Размочаева Н.В. |

Санкт-Петербург

2019

**Цель работы.**

Реализовать умный указатель разделяемого владения объектом. Поведение реализованных функций должно быть аналогично функциям std::shared\_ptr

**Задание.**

Реализовать базовые методы умного указателя разделяемого владения объектом, после чего модифицировать созданный shared\_ptr так, чтобы он был пригоден для полиморфного использования. Должны быть обеспечены следующие возможности:

* копирование указателей на полиморфные объекты
* сравнение shared\_ptr как указателей на хранимые объекты.

**Ход работы.**

Реализация shared\_ptr.

В качестве полей были созданы: указатель на объект типа T, и счетчик для подсчета количества умных указателей, которые ссылаются на один объект.

Также была написана функция count\_dec(), осуществляющая уменьшение счетчика, и при необходимости удаляющая поля указателя.

Был реализованы: конструктор, принимающий указатель, и деструктор. В конструкторе поле, отвечающее за указатель, задается переданным указателем, а счетчик инициализируется значением 1. Деструктор вызывает функцию count\_dec(), которая удалит данные при разрушении умного указателя только в том случае, если данный умный указатель – единственный.

Реализованные операторы: оператор присваивания, оператор bool(), который проверяет, указывает ли указатель на объект, методы get(), use\_count(), операторы \* и ->, swap() и reset() (для замены объекта, которым владеет умный указатель).

Для полиморфного использования внутри класса был объявлен friend- класс shared\_ptr другого типа, связанного с типом реализуемого указателя посредством наследования. Были переписаны конструктор копирования и оператор присваивания с использованием template. Также были добавлены операторы != и ==, чтобы умные указатели можно было сравнивать как указатели на хранимые объекты.

**Вывод.**

При выполнении лабораторной работы были реализованы основные функции для умного указателя разделяемого владения объектом, также класс был изменён для полиморфного использования.