

**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: Умные указатели

Студент гр. 7304

Ажель И.В.

Преподаватель

Размочаева. Н.В.

Санкт-Петербург

2019

Цель работы.

Изучить умные указатели на примере *shared_ptr*, изучить их реализацию и работу на языке программирования C++, написать тесты для практического применения.

Основные теоретические положения.

Класс *shared_ptr* описывает объект, который использует подсчет ссылок для управления ресурсами. Объект *shared_ptr* фактически содержит указатель на ресурс, которым он владеет, или содержит пустой указатель (NULL). Обладать ресурсом могут несколько объектов *shared_ptr*; при удалении последнего объекта *shared_ptr*, обладающего тем или иным ресурсом, данный ресурс освобождается.

shared_ptr прекращает владеть ресурсом при переназначении или сбросе.

Аргумент шаблона *T* может быть неполным типом, за исключением случаев, особо отмеченных для определенных функций-членов.

Постановка задачи.

Необходимо реализовать умный указатель разделяемого владения объектом (*shared_ptr*).

Для того, чтобы *shared_ptr* можно было использовать везде, где раньше использовались обычные указатели, он должен полностью поддерживать их семантику. Модифицируйте созданный на предыдущем шаге *shared_ptr*, чтобы он был пригоден для полиморфного использования. Должны быть обеспечены следующие возможности:

- копирование указателей на полиморфные объекты

```
stepik::shared_ptr<Derived> derivedPtr(new Derived);  
stepik::shared_ptr<Base> basePtr = derivedPtr;
```

- сравнение `shared_ptr` как указателей на хранимые объекты.

Требования к реализации: при выполнении этого задания вы можете определять любые вспомогательные функции. Вводить или выводить что-либо **не нужно**. Реализовывать функцию `main` не нужно. Не используйте функции из `cstdlib` (`malloc`, `calloc`, `realloc` и `free`).

Ход работы.

1. Написан класс *shared_ptr*. Класс хранит следующие переменные:

- `Type* m_ptr;`
- `Long* m_count;`

В классе реализованы следующие методы:

- Конструктор класса
 - Конструктор копирования
 - Оператор копирования
 - Оператор сравнения
 - Функция `bool()` для проверки хранения элементов
 - Функция `get()`, предоставляющая доступ к хранимым элементам
 - Функция `use_count()`, которая возвращает количество объектов *shared_ptr*, ссылающиеся на тот же управляемый объект
 - Оператор `*`, который возвращает ссылку на управляемый объект
 - Оператор `->`, который возвращает указатель на управляемый объект
 - Функция `swap(shared_ptr&)`, которая обменивает указатели
 - Функция `reset()`, которая замещает указатель на другой
2. Код представлен в приложении А.
3. Написаны тесты для *shared_ptr*

Выводы.

В ходе выполнения данной лабораторной работы были изучены умные указатели на примере *shared_ptr*, реализована их работа на языке программирования C++, написаны тесты для практического применения.