

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

Моторин Е.В.

Преподаватель

Размочаева Н.В.

Санкт-Петербург

2019

Цель работы:

Изучить умные указатели на примере `shared_ptr` в языке программирования C++.

Задача:

Необходимо реализовать умный указатель разделяемого владения объектом (`shared_ptr`). Поведение реализованных функций должно быть аналогично

функциям `std::shared_ptr` (http://ru.cppreference.com/w/cpp/memory/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.
Поведение реализованных функций должно быть аналогично функциям `std::shared_ptr` (http://ru.cppreference.com/w/cpp/memory/shared_ptr).

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`, который хранит следующие переменные:

- `Type* m_ptr`
- `Type* m_count`

В классе реализованы следующие методы:

- Конструктор класса
- Конструктор копирования
- Оператор копирования
- Оператор сравнения
- Функция для проверки хранения элементов
- Функция `get()`, предоставляющая доступ к хранимым элементам
- Функция `use_count()`, которая возвращает количество объектов `shared_ptr`, ссылающиеся на этот же управляемый объект
- Оператор `*`, который возвращает ссылку на управляемый объект
- Оператор `->`, который возвращает указатель на управляемый объект
- Функция `swap()`, которая обменивает указатели
- Функция `reset()`, которая замещает указатель на другой

2. Написаны тесты для `shared_ptr`

Вывод:

Таким образом, в ходе данной лабораторной работы были изучены умные указатели на примере `shared_ptr`, реализована их работа на языке программирования C++, написаны тесты для практического применения.