**СУ “Св. Климент Охридски”,**

**ФМИ – Софтуерно инженерство**

**Курсов проект по Обектно-ориентирано програмиране**

**Vector**

Антон Александров Петков, Факултетен № 61793

Съдържание

[1. Въведение 2](#_Toc409876355)

[2. Описание на приложените алгоритми 2](#_Toc409876356)

[3. Описание на програмния код 2](#_Toc409876357)

[4. Използвани технологии 2](#_Toc409876358)

# Въведение

Имплементация на структурата от данни Vector, използвайки C++11, RAII, copy-and-swap idiom. Управлението на паметта е делегирано на помощен вектор, който използва стандартния std::allocator. Целта ми беше да пиша качествен код, да спазя добър обектно-ориентиран дизайн и да се възползвам от инструментите на C++11.

Проектът е качен в github на адрес <https://github.com/tonynho/Vector>.

# Описание на приложените алгоритми

Множителят на растеж на вектора е 1.5.

void std::uninitialized\_fill(ForwardIt first, ForwardIt last, const T& value) – конструира всеки елемент в интервала [first, last) чрез копиращия конструктор на T и стойност value. В случай, че при извикването на конструктора се хвърли изключение, функцията го обработва и унищожава всички обекти, които е създала до момента, за да остави състоянието, което е заварила първоначално.

ForwardIt std::uninitialized\_copy(ForwardIt first, ForwardIt last, ForwardIt dest) – Копира поелементно всички обекти от интервала [first, last) в интервала започващ с dest включително, използвайки конструктора за копиране. В случай, че при извикването на конструктора се хвърли изключение, функцията го обработва и унищожава всички обекти, които е създала до момента.

void uninitialized\_move(T\* begin, T\* end, T\* dest) - Помощна функция за преместващо копиране на интервал [begin, end) в интервал, започващ с dest включително. След изпълнението на функцията, елементите от интервала [begin, end) ще са унищожени.

void swap\_range\_backwards(T\* begin, T\* end) - Помощна функция, приемаща интервала [begin, end), която премества end наляво до позиция begin чрез последователни разменяния на директните съседни елементи. Функцията се използва за реализиране на insert метода.

# Описание на програмния код

VectorBase<T, A> забранява копиращ конструктор и оператор за копиращо присвояване, защото не са нужни. Нужни са само преместващ конструктор и оператор за преместващо присвояване, защото след копиране на VectorBase аргумента го изтриваме, т.е. директно преместваме съдържанието и не заделяме излишни обекти.

Vector<T, A> е композиран (has-a relationship) от VectorBase<T, A> (RAII). Така спестяваме писането на излишен код и разделяме отговорностите на класовете.

Операторът за копиращо присвояване и shrink\_to\_fit са имплементирани чрез copy-and-swap идиома.

std::swap гарантира, че не хвърля изключния, а функциите std::uninitialized\_fill и std::uninitialized\_copy обработват евентуални изключения хвърлени от конструкторите и връщат началното състояние на обектите, по които работят.

Vector<T, A> минимизира изискванията към шаблонния тип Т, като не изисква той да е default constructable.

Кодът гарантира минимум basic exception safety. Докато например Vector<T, A>::operator=(const Vector& other) гарантира strong exception safety, защото ако нещо се провали, няма странични ефекти.

За по-подробно обяснение на имплементацията, разгледайте коментарите във програмния код.

# Използвани технологии

Code::Blocks IDE, GNU GCC Compiler, C++11 standard [-std=c++11]