**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ   
«МОСКОВСКИЙ ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ  
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)»**

Кафедра информатики и вычислительной математики

Лабораторная работа

На тему: Исследование ёмкости std::vector

Выполнил:

Студент Б03-831а группы

Физтех-школы аэрокосмических технологий

Попов Н. В.

Преподаватель:

Макаров И. С.

Москва 2020

Оглавление

[Введение 3](#_Toc33935707)

[Базовые понятия об аллокации вектора 4](#_Toc33935708)

[Поведение вектора при больших значениях capacity 5](#_Toc33935709)

[Вывод 10](#_Toc33935710)

# Введение

Целью данной работы является исследование поведения контейнера std::vector при изменении его размера, в частности, при применении функции resize() c параметрами, по порядку величины близким к max\_size().

Технические характеристики оборудования:

* Процессор Intel Core i7-7700HQ (тактовая частота 2.8 ГГЦ)
* 12 ГБ ОЗУ
* операционная система MS Windows 10 x64
* среда разработки MS VS 2019 Community

Ссылка на соответствующий cpp-файл:

<https://github.com/0rthodox/HomeworkCPP/blob/master/Sem3/task1.h>

# Базовые понятия об аллокации std::vector

В начале были проведены исследования на выяснение механизма выделения памяти при малых величинах ёмкости. Обнаружено, что:

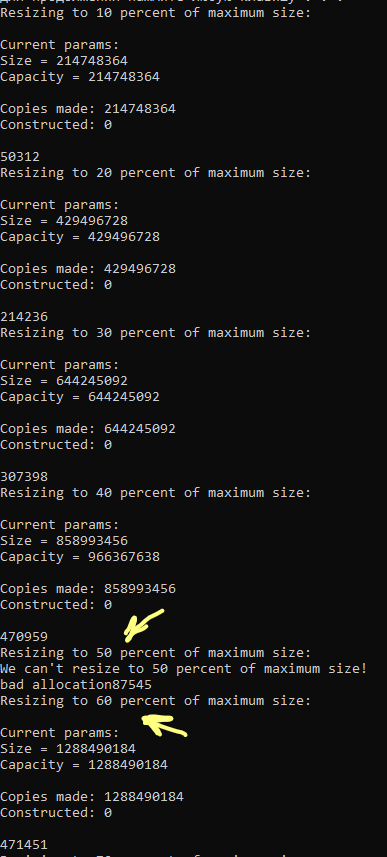
* По умолчанию создаётся std::vector нулевого размера и нулевой ёмкости.
* При увеличении размера вектора до величины, большей, чем текущая ёмкость, ёмкость вектора увеличивается как минимум в 1.5 раза (с округлением вниз), но если необходимо увеличить ёмкость до величины, превышающей это значение, она станет в точности равной этой величине.

В приведённом файле присутствуют примеры с пояснениями, иллюстрирующие и/или дополняющие утверждения выше.

# Поведение std::vector при больших значениях capacity

В исследовании было использовано значение

.

С помощью методов assign() и resize() был исследован механизм выделения памяти при значениях, равных десяткам процентов от данного числа, шаблоном выступал объект Logger, имеющий два статических поля.

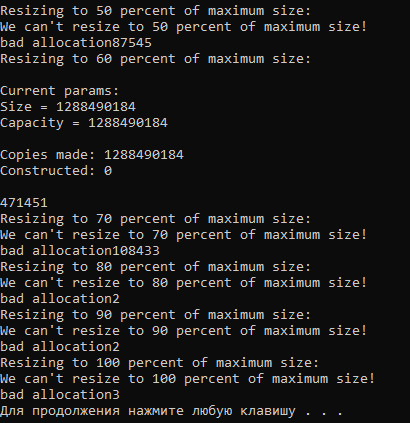
Были получены графики зависимости времени реаллоцирования от требуемой ёмкости при последовательном увеличении и уменьшении размера вектора.

Выяснено, что можно увеличить ёмкость std::vector до 60-70% от max\_size(). При увеличении ёмкости замечен парадокс: не удалось увеличить размер до 50%, но удалось до 60%. Это объясняется параллельным использованием компьютера в тот момент и ещё раз подтверждает тот факт, что память под нужды std::vector выделяется из кучи.

Замечено, что при недостатке памяти выбрасывается exception с

Рис.1. Увеличение (I)

.

Полученные результаты подтверждают, что при реаллокации происходит копирование, а при уменьшении размера не происходит реаллокации.

Заметим, что освобождение памяти в данном случае занимает примерно на порядок меньше времени, чем копирование объектов.



Рис.2. Увеличение (II)

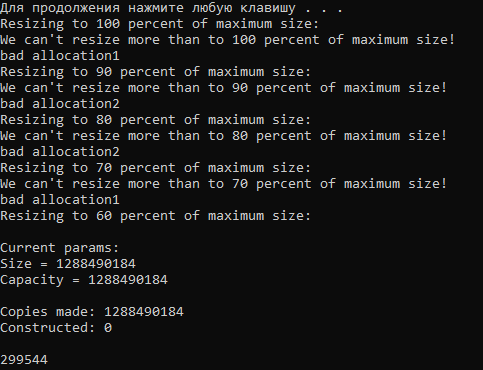
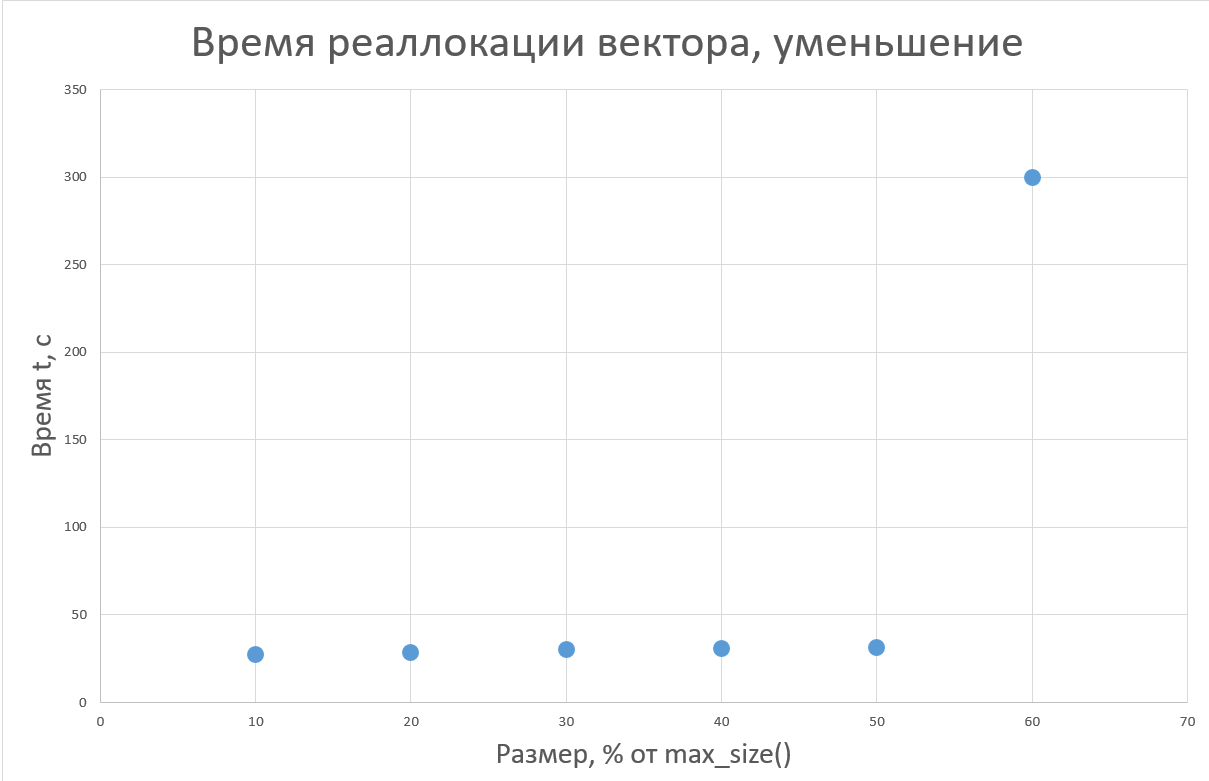


Рис.3. Уменьшение (I)



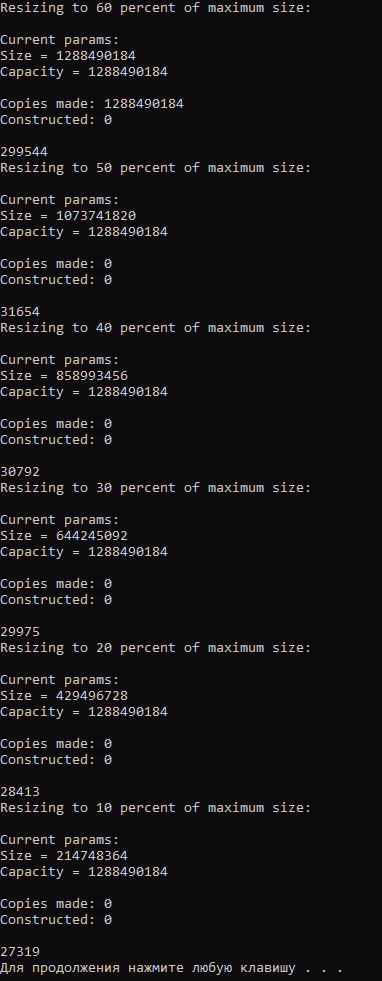


Рис.4. Уменьшение (II)

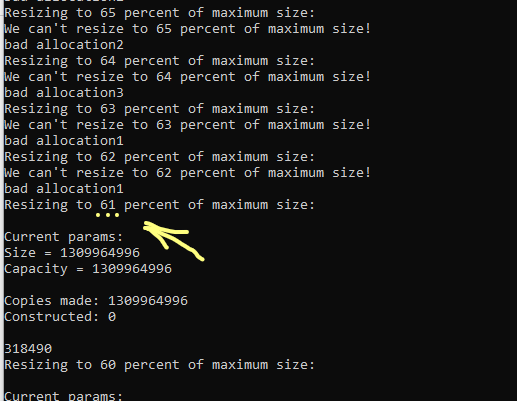
Далее было уточ-нено максималь-ное достижимое значение ёмкости, равное 61% от max\_size().

Рис.4. Уменьшение (II)

# Вывод

В данной работе был исследован механизм выделения памяти в std::vector, определены характерные отношения времён аллокации/деаллокации, получено максимальное достижимое значение ёмкости для данного оборудования при имеющихся задачах, обнаружено и объяснено парадоксальное явление, возникшее при реаллокации памяти.