Федеральное агентство по образованию Российской Федерации

Государственное образовательное учреждение

высшего профессионального образования

Нижегородский государственный университет им. Н.И. Лобачевского

Институт информационных технологий, математики и механики

Отчёт по лабораторной работе

Множества на основе битовых полей

Выполнил:

студент института ИТММ гр. 381908-4

Нарлыев П.

Проверил:

ассистент каф. МОСТ, ИИТММ

Лебедев И.Г.

Содержание

[Введение 3](#_Toc59364219)

[Постановка задачи 4](#_Toc59364220)

[Руководство пользователя 5](#_Toc59364221)

[Руководство программиста 6](#_Toc59364222)

[Описание структуры программы 6](#_Toc59364223)

[Описание структур данных 6](#_Toc59364224)

[Описание алгоритмов 9](#_Toc59364225)

[Заключение 12](#_Toc59364226)

[Литература 13](#_Toc59364227)

# Введение

Множество — это структура данных, эквивалентная множествам в математике.

Множество может состоять из различных элементов, порядок элементов в множестве не определен. В множество можно добавлять и удалять элементы, можно перебирать элементы множества, можно выполнять операции над множествами (объединение, пересечение, разность). Можно проверять принадлежность элементу множества.

В программирование множества помогают решать задачи различного характера. Например, есть множество всех работников предприятия и множество работников, прошедших ежемесячную проверку. Тогда с легкостью возможно найти тех, кто из работников не прошел проверку.

# Постановка задачи

Задача конкретной работы — разработка структуры данных для внедрения битовых полей с использованием множеств, кроме того, нужно было освоить инструменты разработки программного обеспечения, такие как Git и Google Test.

Используя шаблон, содержащий интерфейсы классов битового поля и множества (h-файлы), готовый набор тестов для каждого из указанных классов, пример использования класса битового поля и множества для решения задачи поиска простых чисел с помощью алгоритма "Решето Эратосфена", нужно было написать программу, решающую следующие задачи:

* Реализация класса битового поля TBitField согласно заданному интерфейсу,
* Реализация класса множества TSet согласно заданному интерфейсу,
* Обеспечение работоспособности тестов и примера использования,
* Реализация нескольких простых тестов на базе Google Test,
* Публикация исходных кодов в личном репозитории на GitHub.

# Руководство пользователя

Пользователю нужно запустить файл sample\_prime\_numbers.exe.

Откроется консольное приложение для тестирования программ поддержки битового поля на основе решета Эратосфена.

Программа запросит пользователя ввести верхнюю границу целых значений.

После ввода числа, программа выполнит действия и выведет результат в консоль.

Для повторного выполнения потребуется перезапустить программу.

# Руководство программиста

## **Описание структуры программы**

#include "tbitfield.h" – подключение файла TBitField.h, описывающего работу с битовыми полями.

#include "tset.h" – подключение файла TSet.h, описывающего работу с множествами, на основе битовых полей.

#include <iomanip> - файл описывает манипуляторы при работе с потоковыми операциями.

Код классов в приложении.

## 

## **Описание структур данных**

Программа состоит из нескольких основных классов – tbitfield, tset.

**tbitfield.cpp, заголовок tbitfield.h.**

Реализованы конструктор с параметром, конструктор копирования, деструктор.

Содержит поля:

* BitLen для хранения длины битового поля,
* TELEM \*pMem для предоставления памяти битовому полю,
* MemLen для хранения количества элементов битового поля.

Реализованы методы:

* GetMemIndex – получение индекса,
* GetMemMask – маска для бита,
* GetLength – получение длины,
* SetBit – установка бита,
* ClrBit – очистка бита,
* GetBit – получение бита.

Реализованы операции:

* operator== - оператор сравнения,
* operator!= - оператор сравнения,
* operator= - оператор присваивания,
* operator| - операция «ИЛИ»,
* operator& - оператор «И»,
* operator~ - оператор отрицания,
* &operator>> -оператор ввода,
* &operator<< - оператор вывода.

**tset.cpp, заголовок tset.h**

Реализованы конструктор с параметром, конструктор копирования, конструктор преобразования типа.

Содержит поля:

* MaxPower для хранения максимальной мощности множества,
* TBitField BitField – битовое поле для хранения характеристического вектора.

Реализует методы:

* GetMaxPower – получение максимальной мощности множества,
* InsElem – включение элемента в множество,
* DelElem – удаление элемента из множества,

Реализует операции:

* operator== - оператор сравнения,
* operator!= - оператор сравнения,
* operator= - оператор присваивания,
* operator+ - операция объединение с элементом,
* operator- - операция разность с элементом,
* operator+ - операция объединение множеств,
* operator\* - операция пересечение множеств,
* operator~ - операция дополнения,
* &operator>> -оператор ввода,
* &operator<< - оператор вывода.

# Описание алгоритмов

Создание множества:

* Инициализируем битовое поле размером, равным мощности множества,
* Выделяем память,
* Заполняем элементы нулями.

Добавление элемента в множетсво:

* Инициализируем битовое поле,
* Передаем элемент в класс битового поля,
* На основе элемента получаем индекс и маску,
* Используя побитовое «ИЛИ», присваиваем по полученному индексу, полученную маску.

Удаление элемента из множества:

* Передаем элемент в класс битового поля,
* На основе элемента получаем индекс и маску,
* Используя побитовое «И», присваиваем по полученному индексу, полученную маску, предварительно применив к маске побитовую инверсию.

# Заключение

При выполнении работы были получены навыки работы с Google Tests и CMake и реализации множества на основе битовых полей.

# Литература

* Столлингс, В. Структурная организация и архитектура компьютерных систем, 5-е изд.: Пер. с англ. — М.: Издательский дом «Вильямс», 2002. — 896 с.: ил. — Парал. тит. англ.
* Johnson M. Superscalar Microprocessor Design. — Englewood Cliff, New Jersey: Prentice Hall, 1991.
* Касперски К. Техника оптимизации программ. Эффективное использование памяти. — СПб.: БХВ-Петербург, 2003. — 464 с.: ил.
* Stone H. High performance Computer Architecture. — Reading, MA: Addison-Wesley, 1993.
* Tullsen D.M., Eggers S.J. Effective Cache Prefetching on a Bus-Based Multiprocessor. — ACM Transactions on Computer Systems, pp. 57-88, Feb 1995.
* Chandra D., Guo F., Kim S., Solihin Y. Predicting inter-thread cache contention on a chip multi-processor architecture. — Proceedings of the 11th International Symposium on High Performance Computer Architecture (HPCA), pp. 340–351, Feb 2005.
* Press W., Teukolsky S., Vetterling W., Flannery B. Numerical Recipes in C. The Art of Scientific Computing. Second Edition. — Cambridge University Press, 1992.
* Камаев А.М., Сиднев А.А., Сысоев А.В. Об одном подходе к анализу эффективности приложений // Труды 50-й научной конференции МФТИ «Современные проблемы фундаментальных и прикладных наук»: Часть I. Радиотехника и кибернетика. - М.: МФТИ, 2007.
* Debugging and performance monitoring. Intel® 64 and IA-32 Architectures Software Developer’s Manual. Volume 3B: System Programming Guide, Part 2. May 2007. — [http://www.intel.com/products/processor/manuals/]
* Юнаковский А.Д. Начала вычислительных методов для физиков. – Н. Новгород: ИПФ РАН, 2007.