МИНИСТЕРСТВО НАУКИ И ВЫСШЕГООБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования   
**«Национальный исследовательский   
Нижегородский государственный университет им. Н.И. Лобачевского»**

**(ННГУ)**

**Институт информационных технологий, математики и механики**

**ЛАБОРАТОРНАЯ РАБОТА**

на тему:

**«Битовые поля и множества»**

**Выполнил(а):** студент(ка) группы 3822Б1ФИ2

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Холин К.И

Подпись

**Проверил:** к.т.н, доцент каф. ВВиСП

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Кустикова В.Д./

Подпись

Нижний Новгород  
2023

**Содержание**

[Введение 3](#_Toc147915966)

[1 Постановка задачи 4](#_Toc147915967)

[2 Руководство пользователя 5](#_Toc147915968)

[2.1 Приложение для демонстрации работы битовых полей 5](#_Toc147915969)

[2.2 Приложение для демонстрации работы множеств 5](#_Toc147915970)

[2.3 «Решето Эратосфено» 5](#_Toc147915971)

[3 Руководство программиста 6](#_Toc147915972)

[3.1 Описание алгоритмов 6](#_Toc147915973)

[3.1.1 Битовые поля 6](#_Toc147915974)

[3.1.2 Множества 6](#_Toc147915975)

[3.1.3 «Решето Эратосфена» 6](#_Toc147915976)

[3.2 Описание программной реализации 6](#_Toc147915977)

[3.2.1 Описание класса TBitField 6](#_Toc147915978)

[3.2.2 Описание класса TSet 7](#_Toc147915979)

[Заключение 8](#_Toc147915980)

[Литература 9](#_Toc147915981)

[Приложения 10](#_Toc147915982)

[Приложение А. Реализация класса TBitField 10](#_Toc147915983)

[Приложение Б. Реализация класса TSet 10](#_Toc147915984)

# Введение

В C++ иногда возникают такие ситуации,когда информацию об объекте достаточно хранить в формате состояний(статусов),представляющих из себя 0 и 1. На этом основывается проект Множества,который использует интерфейс битовых полей для реализации работы с теоретико-множественными операциями. Это самый оптимальный вариант,поскольку он даёт нам возможность использовать не всю предоставляемую типом данных память,а только его часть. Обращение к определённому биту позволяет нам узнать его состояние для выполнения конкретной задачи. Например,чтобы проверить элемент на принадлежность множеству в нашем случае. Битовые поля в этом случае играют важную роль.

# Постановка задачи

Цель – реализовать классы: TSet и TBitField

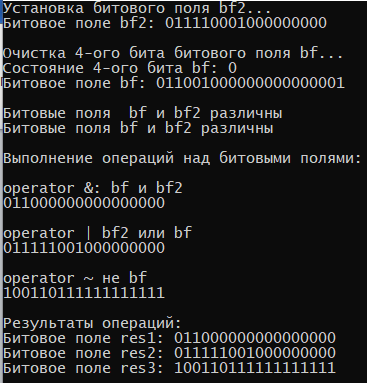
Задачи:

1. Класс для работы с множествами должен поддерживать эффективное хранение данных.
2. Написать следующие операции для работы с битовыми полями: установить бит в 1,установить бит в 0,получить значение бита,сравнить два битовых поля,сложить и инвертировать,вывести битовое поле требуемого формата и ввести битовое поле.
3. Добавить вспомогательные операции получения бита,маски бита,длины битового поля.
4. Написать следующие операции для работы с множествами: вставка элемента,удаление,проверка наличия,сравнение множеств,объединение множеств,пересечение,разность,копирование,вычисление мощности множества,вывод элементов множества требуемого формата и ввод.
5. Добавить вспомогательные операции для получения мощности множества.

# Руководство пользователя

## Приложение для демонстрации работы битовых полей

1. Запустите приложение с названием sample\_TBitField.exe. В результате появится окно, показанное ниже (рис. 1).



1. Основное окно программы
2. На первом шаге создаются 3 битовых поля(рис.2)



Рис.2 Создание битовых полей

1. На следующем шаге выполняется установка битового поля bf и выводится его длина(рис.3)

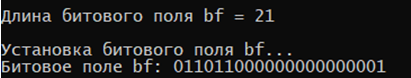


Рис.3 Установка битового поля bf с выводом длины

1. Далее выполняется установка битового поля bf2(рис.4)



Рис.4 Установка битового поля bf2

1. На 5 шаге удаляется бит с номером 4 из битового поля bf(рис.5)



Рис.5 Удаление 4-го бита битового поля bf2

1. В первой строке проверяется операция равенства битовых полей bf и bf2,а во второй- операция неравенства(рис.6)



Рис.6 Сравнение битовых полей

1. На данном этапе выполняются различные операции с битовыми полями(рис.7)

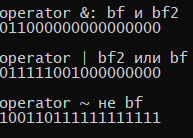


Рис.7 Операции над битовыми полями

1. На завершающем шаге выводятся результаты вычислений(рис.8)



Рис.8 Результаты вычислений

## Приложение для демонстрации работы множеств

1. Запустите приложение с названием sample\_tset.exe. В результате появится окно, показанное ниже (рис. 1).

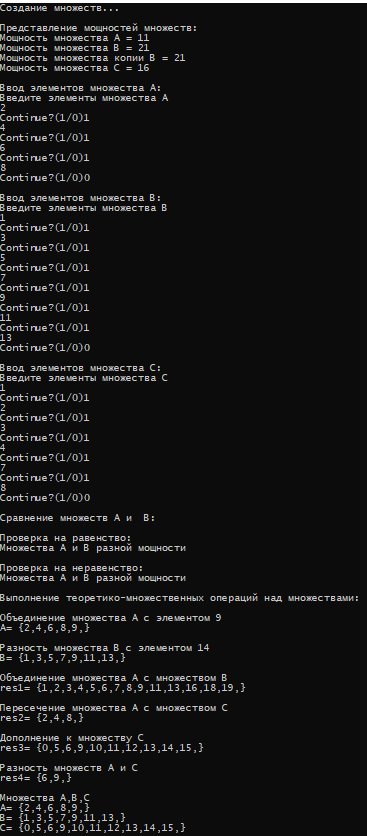


Рис.1 Окно основной программы

1. По началу множества пустые,так как в них не содержится элементов. Далее представлены мощности множеств(рис.2)



Рис.2 Мощности множеств A,B,C

1. На третьем этапе представлен процесс заполнения множеств элементами, введёными с клавиатуры(рис.3)

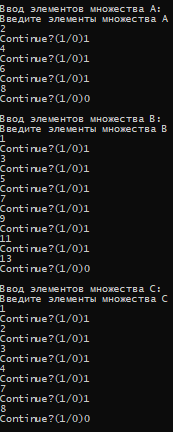


Рис.3 Заполнение множеств A,B,C

1. В этом случае сравниваются два множества на равенство и неравенство(рис.4)

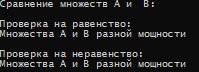


Рис.4 Сравнение множеств A и B

1. На рис.5 приведены основные операции с множествами(рис.5)

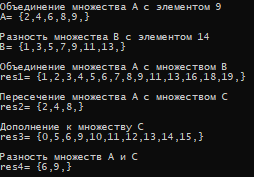


Рис.5 Основные операции с множествами A,B,C

1. В завершение были выведены множества A,B,C, которые принимали участие в программе(рис.6)



Рис.6 Вывод множеств A,B,C

## «Решето Эратосфена»

1. Откройте приложение sample\_primenumbers.exe.В результате появится окно ниже(рис.1)

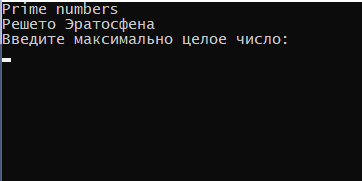
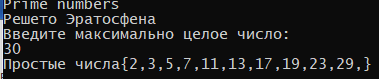


Рис.1 Окно основной программы

1. Вам будет необходимо ввести число ,до которого будут выведены все простые числа на экран. Для примера введём число 30 и посмотрим на результат(рис.2)



1. Рис.2 Все простые числа от 2 до 30

# Руководство программиста

## Описание алгоритмов

### Битовые поля

Битовые поля представляют из себя характеристические массивы,где индексы каждого элемента – это элементы множества. Каждое битовое поле задаётся длиной(унивёрс битов),количеством единиц памяти(кол-во характеристических массивов) и памятью для их хранения. Элемент битового поля может находиться в двух состояниях: 1 и 0. 1- элемент содержится в множестве,а 0 – элемент не содержится в множестве. Данный алгоритм позволяет реализовать интерфейс для работы с множествами.

### Множества

Множества по идее наследуются от класса TBitField. Множество – это класс TSet,реализованный на основе класса TBitField. Работа TSet заключается в том,что он использует класс TBitField как инструмент для создания множеств и осуществления теоретико-множественных операций. Максимальная мощность множества – это и есть длина битового поля. Таким образом, главная роль отводится классу TBitField,который и отвечает за техническую часть работы множеств.

### «Решето Эратосфена»

Решето Эратосфена – это алгоритм, позволяющий найти все простые числа до заданного числа n. Суть этого алгоритма заключается в следующем:

1. Выписать подряд все числа от 2 до n
2. Пусть у нас есть переменная p=2 –первое простое число
3. Зачёркиваем все числа,кратные 2p,3p,4p…
4. Находим первое простое число в списке,большее p. Присваиваем его p
5. Повторяем шаги 3 и 4.

Данный алгоритм позволяет легко и быстро найти все простые числа.

## Описание программной реализации

### Описание класса TBitField

class TBitField

{

private:

int BitLen;

TELEM \*pMem;

int MemLen;

// методы реализации

int GetMemIndex(const int n) const;

TELEM GetMemMask (const int n) const;

int BitsInMem = 16;

int shiftsize = 4;

public:

TBitField(int len);

TBitField(const TBitField &bf);

~TBitField();

// доступ к битам

int GetLength(void) const;

void SetBit(const int n);

void ClrBit(const int n);

int GetBit(const int n) const;

// битовые операции

int operator==(const TBitField &bf) const;

int operator!=(const TBitField &bf) const;

const TBitField& operator=(const TBitField &bf);

TBitField operator|(const TBitField &bf);

TBitField operator&(const TBitField &bf);

TBitField operator~(void);

friend ostream &operator<<(ostream &ostr, const TBitField &bf);

};

Назначение: представление битового поля.

Поля:

BitLen – длина битового поля – максимальное количество битов.

pMem – память для представления битового поля.

MemLen – количество элементов для представления битового поля.

|  |  |  |  |
| --- | --- | --- | --- |
| Методы | Назначение | Входные  параметры | Выходные параметры |
| GetMemIndex | Получение индекса элемента памяти | n- Номер бита | Номер элемента памяти |
| GetMemMask | Получение битовой маски по номеру бита | n- Номер бита | Битовая маска |
| GetLength | Получение длины битового поля | отсутствуют | Длина битового поля |
| SetBit | Установить бит в единицу | n- Номер бита | отсутствуют |
| ClrBit | Установить бит в ноль | n- Номер бита | отсутствуют |
| GetBit | Получение значения бита | n- Номер бита | Значение бита  (0 или 1) |

|  |  |  |  |
| --- | --- | --- | --- |
| Операции | Назначение | Входные  параметры | Выходные параметры |
| Равенство(==)  Operator== | Проверка на равенство двух битовых полей | bf - Битовое поле | Целое число  (0 или 1) |
| Неравенство(!=)  Operator!= | Проверка на неравенство двух битовых полей | bf - Битовое поле | Целое число  (0 или 1) |
| Присваивание(=)  Operator!= | Присвоение значений полей одного битового поля другому | bf - Битовое поле | Константная ссылка на объект своего класса TBitField |
| Побитовое ИЛИ(|)  Operator| | Сложение двух битовых полей | bf - Битовое поле | Результирующее битовое поле |
| Побитовое И(&)  Operator& | Умножение двух битовых полей | bf - Битовое поле | Результирующее битовое поле |
| Побитовое отрицание(~)  Operator~ | Инвертирование значений битов битового поля | bf - Битовое поле | Результирующее битовое поле |
| Вывод(<<)  Operator<< | Вывод битового поля в формате(1010101001 и т.п) | Lnk ostream,const lnk TBitField | Константная ссылка на поток |

|  |  |  |  |
| --- | --- | --- | --- |
| Конструкторы | Назначение | Входные  параметры | Выходные  параметры |
| Инициализатор | Создание битового поля | Len- Длина битового поля | отсутствуют |
| Копирование | Копирование битовых полей | Const TBitField& bf -Константная ссылка на битовое поле | отсутствуют |
| Деструктор | Освобождение памяти | отсутствуют | отсутствуют |

### Описание класса TSet

class TSet

{

private:

int MaxPower;

TBitField BitField;

public:

TSet(int mp);

TSet(const TSet &s);

TSet(const TBitField &bf);

operator TBitField();

// доступ к битам

int GetMaxPower(void) const;

void InsElem(const int Elem);

void DelElem(const int Elem);

int IsMember(const int Elem) const;

// теоретико-множественные операции

int operator== (const TSet &s) const;

int operator!= (const TSet &s) const;

const TSet& operator=(const TSet &s);

TSet operator+ (const int Elem);

TSet operator- (const int Elem);

TSet operator+ (const TSet &s);

TSet operator\* (const TSet &s);

TSet operator~ (void);

TSet operator-(const TSet& obj);

friend istream &operator>>(istream &istr, TSet &bf);

friend ostream &operator<<(ostream &ostr, const TSet &bf);

};

Битовые поля:

MaxPower – максимальная мощность множества

TBitField – битовое поле

|  |  |  |  |
| --- | --- | --- | --- |
| Методы/  Операции | Назначение | Входные  параметры | Выходные  параметры |
| GetMaxPower | Получение мощности множества | Отсутствуют | Мощность множества |
| InsElem | Добавление элемента в множество | Elem- добавляемый элемент | отсутствуют |
| DelElem | Исключение элемента из множества | Elem – удаляемый элемент | отсутствуют |
| IsMember | Проверка  на принадлежность множеству | Elem – элемент | Значение бита  (0 или 1) |
| Равенство(==)  Operator== | Проверка на равенство двух множеств | s - множество | Целое число  (0 или 1) |
| Неравенство(!=)  Operator!= | Проверка на неравенство двух множеств | s - множество | Целое число  (0 или 1) |
| Присваивание(=)  Operator= | Присвоение значений полей одного объекта класса другому | s - множество | Ссылка на объект своего класса TSet |
| Объединение  с элементом  Operator+ | Побитовое сложение соответствующего элемента множества с элементом | Elem- добавляемый элемент | Результирующее множество |
| Пересечение  с элементом  operator& | Побитовое умножение соответствующего элемента множества с элементом | Elem– добавляемый элемент | Результирующее множество |
| Пересечение множеств  Operator& | Побитовое умножение элементов двух множеств | s - множество | Результирующее множество |
| Разность  с элементом - | Исключение соответствующего элемента множества | Elem – вычитаемый элемент | Результирующее множество |

|  |  |  |  |
| --- | --- | --- | --- |
| Разность множеств  Operator- | Исключение элементов одного множества элементами другого множества | s - множество | Результирующее множество |
| Дополнение  к множеству  operator~ | Инвертировать значения битового поля. Это и будет дополнение к множеству. | отсутствуют | Результирующее множество |
| Вывод  Operator<< | Вывод элементов множества в формате({e1,e2,…,en}) | Lnk ostream,const lnk TSet | Ссылка на поток  вывода |
| Ввод  Operator>> | Заполнение множества | Lnk ostream, lnk TSet | Ссылка на поток ввода |

# 

|  |  |  |  |
| --- | --- | --- | --- |
| Конструкты/  операторы | Предназначение | Входные  параметры | Выходные  параметры |
| Инициализатор | Создание множеств | Mp – мощность множества | Отсутствуют |
| Копирование | Копирование множеств | s - множество | Отсутствуют |
| Преобразование  типа | Преобразование из TBitField в TSet | Bf – Битовое поле | Отсутствуют |
| Оператор преобразования в тип TBitField | Преобразование из TSet в TBitField | отсутствуют | Объект класса TBitField |

# Заключение

По результатам лабораторной работы были реализованы классы TSet и TBitField,а также написаны приложения и тесты для проверки работоспособности реализации. К лабораторной работе был составлен полный отчёт по теме со всеми подробными описаниями.

# Литература

1. Битовые поля и операции над ними <https://rep.bntu.by/bitstream/handle/data/32621/Bitovye_polya_i_operacii_nad_nimi.pdf?sequence=1&isAllowed=y>
2. Битовые поля. Урок 32

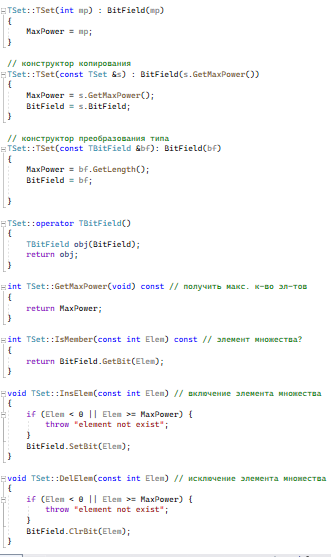
<https://narodstream.ru/c-urok-32-bitovye-polya/>

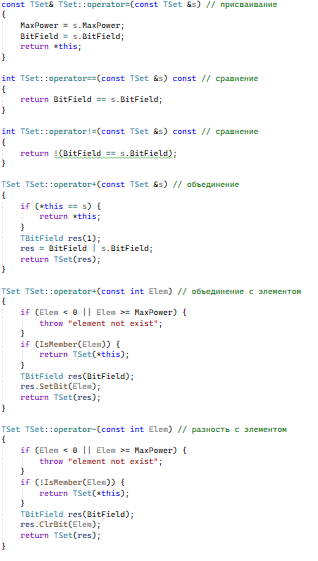
1. Битовые поля

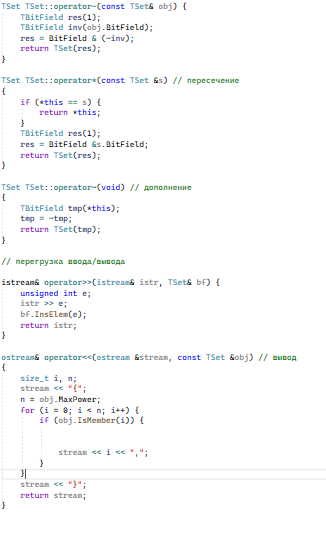
<https://www.c-cpp.ru/books/bitovye-polya>

# Приложения

## Приложение А. Реализация класса TSet







## Приложение Б. Реализация класса TBitField

