

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

Отчёт №1

по учебной дисциплине

«Алгоритмы и структуры данных»

Студента Д.Д. Ципина

группы 3824Б1ФИ2

Нижний Новгород – 2025 г.

Постановка задачи

Цель данной работы — разработка структуры данных для хранения множеств (TSet) с использованием битовых полей (TBitField), а также освоение фреймворка для разработки автоматических тестов Google Test.

Описание класса TBitField

Класс TBitField предназначен для хранения и обработки битовой информации фиксированной длины. Он реализует поле булевых значений, позволяя выполнять операции установки, сброса и проверки отдельных битов, а также логические операции над битовыми полями.

Описание ключевых методов TBitField

TBitField(int len)

- **Параметры:** len — количество битов.
- **Функционал:** выделяет память под битовое поле длиной len, инициализирует все биты нулями.
- **Сложность:** $O(n)$, где n — количество элементов массива pMem.

void SetBit(int n)

- **Параметры:** n — индекс бита.
- **Функционал:** устанавливает бит с номером n в значение 1.

void ClrBit(int n)

- **Функционал:** сбрасывает бит n в значение 0.

int GetBit(int n) const

- **Функционал:** возвращает значение бита n (0 или 1).

TBitField operator~() const

- **Функционал:** возвращает новое битовое поле, в котором каждый бит инвертирован.
- **Сложность:** $O(n)$

TBitField operator|(const TBitField& bf) const

- **Функционал:** возвращает результат побитового «ИЛИ» двух битовых полей.
- **Сложность:** $O(n)$

TBitField operator&(const TBitField& bf) const

- **Функционал:** возвращает результат побитового «И» двух битовых полей.
- **Сложность:** $O(n)$

bool operator==(const TBitField& bf) const

- **Функционал:** сравнивает два битовых поля на равенство.
- **Сложность:** $O(n)$

friend std::istream& operator>>(std::istream& istr, TBitField& bf)

- **Функционал:** считывает строку из символов '0' и '1', устанавливает соответствующие биты.
- **Сложность:** $O(n)$

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

- **Функционал:** выводит битовое поле в виде строки из '0' и '1'.
- **Сложность:** $O(n)$

Описание класса TSet

Класс TSet реализует множество целых чисел от 0 до `MaxPower - 1`, на основе класса TBitField. Это позволяет выполнять операции над множествами, такие как

включение и исключение элементов, проверку принадлежности, объединение, пересечение и дополнение.

Описание ключевых методов TSet

TSet(int mp)

- **Параметры:** mp — максимальная мощность множества.
- **Функционал:** создаёт множество, способное хранить элементы от 0 до mp - 1.
- **Сложность:** $O(n)$, где n — количество битов в BitField.

void InsElem(int Elem)

- **Параметры:** Elem — индекс элемента.
- **Функционал:** включает элемент Elem в множество, устанавливая соответствующий бит.

void DelElem(int Elem)

- **Функционал:** исключает элемент Elem из множества, сбрасывая соответствующий бит.

bool IsMember(int Elem) const

- **Функционал:** возвращает true, если элемент Elem принадлежит множеству.

TSet operator+(const TSet& s) const

- **Функционал:** возвращает объединение двух множеств.
- **Сложность:** $O(n)$

TSet operator*(const TSet& s) const

- **Функционал:** возвращает пересечение двух множеств.
- **Сложность:** $O(n)$

TSet operator~() const

- **Функционал:** возвращает дополнение множества относительно универсального множества мощности MaxPower.
- **Сложность:** O(n)

operator TBitField()

- **Функционал:** преобразует множество в соответствующее битовое поле.

friend std::istream& operator>>(std::istream& istr, TSet& s)

- **Функционал:** считывает строку из '0' и '1', устанавливает соответствующие элементы множества.
- **Сложность:** O(n)

friend std::ostream& operator<<(std::ostream& ostr, const TSet& s)

- **Функционал:** выводит множество в виде строки '0' и '1', отражающее включённые элементы.
- **Сложность:** O(n)

Краткие комментарии к тестам

Для проверки корректности работы классов TBitField и TSet были разработаны тесты с использованием фреймворка **Google Test**. Тесты охватывают базовые операции, граничные случаи и поведение при ошибках.

Тесты класса TBitField

- **ConstructAndLength** – Проверяет корректность создания битового поля с заданной длиной (10) и выброс исключения при попытке создать поле с отрицательной длиной (-1).
- **SetClrGetBit** – Последовательно проверяет установку (SetBit), сброс (ClrBit) и чтение (GetBit) бита с индексом 2. Также проверяется выброс

исключений при выходе за границы: установка бита с индексом 5 (равным длине) и чтение бита с отрицательным индексом.

- **LogicOperators** – Проверяет работу логических операций:
 - `|` (объединение): устанавливаются биты в разных позициях, результат должен содержать оба.
 - `&` (пересечение): результат не должен содержать ни один бит, если они не совпадают.
 - `~` (инверсия): проверяется, что инвертированный бит 1 становится 0, а остальные — 1.

Тесты класса TSet

- **InsDelIsMemTest** – Проверяет базовые операции над множеством:
 - `IsMember` — начальная проверка отсутствия элемента.
 - `InsElem` — включение элемента 3.
 - `DelElem` — исключение элемента 3. Все действия сопровождаются проверкой принадлежности.
- **UniIntersOps** – Проверяет работу операций над множествами:
 - `+` (объединение): элементы 1 и 2 из разных множеств должны присутствовать в объединении.
 - `*` (пересечение): множества не пересекаются, результат не должен содержать ни один элемент.
- **Complement** – Проверяет операцию дополнения множества:
 - Исходное множество содержит элемент 0.
 - После инверсии (`~`) элемент 0 должен отсутствовать, а элементы 1 и 2 — присутствовать.