

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

Отчёт по лабораторной работе №1
дисциплины «Алгоритмы и структуры данных»

Выполнил:

Студент группы 3824Б1ФИ2

Старостин Д.Д.

1. 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.

2. Описание программной реализации

2.1.1. Описание класса TBitField

Класс TBitField реализует структуру данных битовое поле. Содержит поля:

- BitLen — максимальное количество бит в поле;
- pMem — память для представления битового поля;
- MemLen — количество элементов pMem для представления битового поля.

Методы класса обеспечивают доступ к отдельным битам, выполнение битовых операций над полем, а также ввод и вывод данных.

2.1.2. Описание методов класса и функций

1) TBitField(int len) — конструктор с параметром len.

- Параметры и возвращаемые значения: int len — задаёт значение поля BitLen.
- Функционал: присваивает полю BitLen значение len. Рассчитывает значение для пол MemLen. Выделяет динамическую память под массив из MemLen элементов типа TELEM и присваивает полю pMem указатель на начало этого массива. Присваивает всем элементам массива pMem значение 0.
- Сложность: линейная $O(\text{MemLen})$, так как для обнуления массива pMem используется цикл от 0 до MemLen $O(\text{MemLen})$.

2) TBitField(const TBitField &bf) — конструктор копирования.

- Параметры и возвращаемые значения: const TBitField &bf — константная ссылка на копируемый объект.
- Функционал: создаёт копию объекта bf.
- Сложность: линейная $O(\text{bf.MemLen})$, так как выполняется копирование элементов bf.pMem в цикле с bf.MemLen итерациями.

3) ~TBitField() – деструктор.

- Параметры и возвращаемые значения: -
- Функционал: освобождает память поля pMem.

4) int GetMemIndex(const int n) const

- Параметры и возвращаемые значения: const int n – номер бита. Возвращает индекс (int) бита n в pMem.
- Функционал: вычисляет индекс в pMem для бита n.

5) TELEM GetMemMask (const int n) const

- Параметры и возвращаемые значения: const int n – номер бита. Возвращает маску (TELEM) для бита n.
- Функционал: Вычисляет и возвращает маску для бита n.

6) int GetLength(void) const

- Параметры и возвращаемые значения: возвращает длину битового поля (int)
- Функционал: возвращает длину битового поля

7) void SetBit(const int n)

- Параметры и возвращаемые значения: const int n – номер бита.
- Функционал: устанавливает значение 1 для бита n.

8) void ClrBit(const int n)

- Параметры и возвращаемые значения: const int n – номер бита.
- Функционал: устанавливает значение 0 для бита n.

9) int GetBit(const int n) const;

- Параметры и возвращаемые значения: const int n – номер бита. Возвращает значение бита n (int) в pMem.
- Функционал: возвращает значение бита n в pMem.

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

- Параметры и возвращаемые значения: `const TBitField &bf` – объект, с которым происходит сравнение. Возвращает результат сравнение объектов `this` и `bf`.
- Функционал: сравнивает `this` и `bf` и возвращает 0, если они не равны, или 1 иначе.
- Сложность: В среднем и худшем случае $O(\text{MemLen})$, так как в цикле от 0 до `MemLen` на каждой итерации происходит сравнение `i` с `MemLen`, инкрементация `i`, доступ к `i` элементу `pMem` и `bf.pMem`, операция XOR; итого как минимум 5 операций на каждой итерации. В лучшем случае $O(1)$, если у битовых полей разные размеры или они отличаются в первых нескольких элементах `pMem`.

11) `int operator!=(const TBitField &bf) const`

- Параметры и возвращаемые значения: `const TBitField &bf` – объект, с которым происходит сравнение. Возвращает результат сравнение объектов `this` и `bf`.
- Функционал: сравнивает `this` и `bf` и возвращает 1, если они не равны, или 0 иначе.
- Сложность: аналогична сложности аналогична сложности `int operator==(const TBitField &bf) const`.

12) `TBitField& operator=(const TBitField &bf)`

- Параметры и возвращаемые значения: `const TBitField &bf` – ссылка на объект-источник данных. Возвращает ссылку на `this` после присваивания.
- Функционал: обеспечивает копирование данных из объекта `bf` в текущий объект.
- Сложность: $O(\text{bf.MemLen})$, так как происходит поэлементное копирование элементов `bf` в `this` в цикле.

13) `TBitField operator|(const TBitField &bf)`

- Параметры и возвращаемые значения: `const TBitField &bf` – ссылка на второй операнд. Возвращает копию объекта-результата выполнения операции.
- Функционал: реализует операцию побитового ИЛИ между битовыми полями.

- Сложность: $O(\text{MemLen})$, так как при исполнении вызывается конструктор с параметром `MemLen` и выполняется цикл с `MemLen` итерациями.

14) `TBitField operator&(const TBitField &bf)`

- Параметры и возвращаемые значения: `const TBitField &bf` – ссылка на второй операнд. Возвращает копию объекта-результата выполнения операции.
- Функционал: реализует операцию побитового И между битовыми полями.
- Сложность: $O(\text{MemLen})$, так как при исполнении вызывается конструктор с параметром `MemLen` и выполняется цикл с `MemLen` итерациями.

15) `TBitField operator~(void)`

- Параметры и возвращаемые значения: возвращает копию объекта-результата выполнения операции.
- Функционал: инвертирует все биты поля.
- Сложность: $O(\text{MemLen})$, так как при исполнении вызывается конструктор копирования от объекта с размером `MemLen` у поля `pMem`, а также выполняется цикл с `MemLen` итерациями.

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

- Параметры и возвращаемые значения: `std::istream &istr` – ссылка на поток ввода. `TBitField &bf` – ссылка на объект, в который происходит ввод. Возвращает ссылку на объект `istr`.
- Функционал: реализует ввод объекта класса `TBitField`.
- Сложность: $O(\text{BitLen})$, так как выполняется цикл с `BitLen` итерациями.

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

- Параметры и возвращаемые значения: `std::ostream &ostr` – ссылка на поток вывода. `const TBitField &bf` – ссылка на выводимый объект. Возвращает ссылку на объект `ostr`.
- Функционал: осуществляет вывод битовой строки.
- Сложность: $O(\text{BitLen})$, так как выполняется цикл с `BitLen` итерациями.

2.2.1. Описание класса TSet

Класс TSet реализует структуру данных множество. Содержит поля:

- MaxPower – максимальная мощность множества;
- BitField – битовое поле для хранения характеристического вектора.

Методы класса обеспечивают доступ к отдельным элементам, выполнение теоретико-множественных операций, а также ввод и вывод данных.

2.2.2. Описание методов класса и функций

1) TSet(int mp) – конструктор с параметром mp.

- Параметры и возвращаемые значения: int mp – задаёт максимальную мощность множества.
- Функционал: инициализирует поле MaxPower значением mp. Инициализирует поле BitField вызывая для него конструктор от mp.
- Сложность: $O(mp)$, так как вызывается конструктор с параметром класса TBitField, имеющий такую сложность.

2) TSet(const TSet &s) – конструктор копирования.

- Параметры и возвращаемые значения: const TSet &bf – константная ссылка на копируемый объект.
- Функционал: создаёт копию объекта s
- Сложность: $O(bf.MemLen)$, так как вызывает конструктор копирования класса TBitField, имеющий такую сложность.

3) TSet(const TBitField &bf) – конструктор преобразования типа.

- Параметры и возвращаемые значения: const TBitField &bf константная ссылка на объект, который преобразуется.
- Функционал: Преобразует объект класса TBitField в объект класса TSet.
- Сложность: $O(bf.MemLen)$, так как вызывает конструктор копирования класса TBitField, имеющий такую сложность.

4) operator TBitField()

- Параметры и возвращаемые значения: возвращает копию поля TBitField BitField.

- Функционал: обеспечивает неявное преобразование множества к битовому полю.

5) int GetMaxPower() const

- Параметры и возвращаемые значения: возвращает копию поля int MaxPower.
- Функционал: возвращает максимальную мощность множества.

6) void InsElem(const int Elem)

- Параметры и возвращаемые значения: const int Elem – элемент, который нужно включить в множество.
- Функционал: включает элемент в множество.

7) void DelElem(const int Elem)

- Параметры и возвращаемые значения: const int Elem – элемент, который нужно удалить из множества.
- Функционал: удаляет элемент из множества.

8) int IsMember(const int Elem) const

- Параметры и возвращаемые значения: const int Elem – проверяемый элемент. Возвращает 0, если Elem не принадлежит множеству и 1 иначе.
- Функционал: проверяет принадлежность элемента множеству.

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

- Параметры и возвращаемые значения: const TSet &s – ссылка на объект, с которым производится сравнение. Возвращает 1, если объекты равны и 0 иначе.
- Функционал: сравнение двух множеств.
- Сложность: $O(\text{BitField.MemLen})$ так как сравнивает два объекта класса TBitField.

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

- Параметры и возвращаемые значения: `const TSet &s` – ссылка на объект, с которым производится сравнение. Возвращает 1, если объекты не равны и 0 иначе.
- Функционал: сравнение двух множеств.
- Сложность: $O(\text{BitField.MemLen})$ так как сравнивает два объекта класса `TBitField`.

11) `TSet& operator=(const TSet &s)`

- Параметры и возвращаемые значения: `const TSet &s` – ссылка на объект-источник данных. Возвращает ссылку на `this` после присваивания.
- Функционал: обеспечивает копирование данных из объекта `s` в текущий объект.
- Сложность: $O(\text{BitField.MemLen})$ так как сравнивает два объекта класса `TBitField`.

12) `TSet operator+ (const int Elem)`

- Параметры и возвращаемые значения: `const int Elem` – элемент, который нужно включить в множество. Возвращает объект класса `TSet`, который является результатом операции.
- Функционал: создает новое множество, содержащее все элементы исходного множества и элемент `Elem`.
- Сложность: $O(\text{MaxPower})$, так как вызывает конструктор копирования класса `TSet`.

13) `TSet operator- (const int Elem)`

- Параметры и возвращаемые значения: `const int Elem` – элемент, который нужно удалить из множества. Возвращает объект класса `TSet`, который является результатом операции.
- Функционал: создает новое множество, содержащее все элементы исходного множества кроме элемента `Elem`.
- Сложность: $O(\text{MaxPower})$, так как вызывает конструктор копирования класса `TSet`.

14) `TSet operator+ (const TSet &s)`

- Параметры и возвращаемые значения: `const TSet &s` – множество, с которым происходит объединение. Возвращает объект класса `TSet`, который является результатом операции.
- Функционал: создает новое множество, содержащее все элементы исходного множества и элементы множества `s`.
- Сложность: $O(\text{MaxPower})$, так как вызывает конструктор преобразования класса `TSet`.

15) `TSet operator* (const TSet &s)`

- Параметры и возвращаемые значения: `const TSet &s` – множество, с которым происходит пересечение. Возвращает объект класса `TSet`, который является результатом операции.
- Функционал: создает новое множество, содержащее только те элементы, которые содержатся в исходном множестве и в множестве `s` одновременно.
- Сложность: $O(\text{MaxPower})$, так как вызывает конструктор преобразования класса `TSet`.

16) `TSet operator~ ()`

- Параметры и возвращаемые значения: Возвращает объект класса `TSet`, который является результатом операции.
- Функционал: создает новое множество, содержащее только те элементы, которые не содержатся в исходном множестве.
- Сложность: $O(\text{MaxPower})$, так как вызывает конструктор преобразования класса `TSet`.

17) `friend std::istream &operator>>(std::istream &istr, TSet &bf)`

- Параметры и возвращаемые значения: `std::istream &istr` – ссылка на поток ввода. `TSet &bf` – ссылка на объект, в который происходит ввод. Возвращает ссылку на объект `istr`.
- Функционал: реализует ввод объекта класса `TSet`.
- Сложность: $O(n + m)$, где n – длина введенной строки, m – длина битового поля. Вызывает конструктор с параметром m и выполняет цикл с n итерациями.

18) `friend std::ostream &operator<<(std::ostream &ostr, const TSet &bf)`

- Параметры и возвращаемые значения: `std::ostream &ostr` – ссылка на поток вывода. `const TSet &bf` – ссылка на выводимый объект. Возвращает ссылку на объект `ostr`.
- Функционал: выводит множество в поток вывода в формате { *, *, ..., * }.
- Сложность: $O(bf.MaxPower)$, так как выполняется цикл с `bf.MaxPower` итерациями.

3. Тесты

3.1. Тесты TBitField

1) Блок TBitfield_Constructor:

Проверка конструкторов класса TBitField с использованием случайных аргументов.

2) Блок Bits_Access:

Проверка операций, осуществляющих доступ к битам и работу с ними.

Тест `SetBit_and_GetBit` одновременно проверяет корректность работы методов `SetBit` и `GetBit`.

3) Блок Bitwise_operations:

Проверка битовых операций с битовыми полями.

4) Блок TBitfield_Input_Output:

Проверка работы ввода и вывода

5) Блок TBitfield_Exceptions:

Тестирование обработки исключений.

3.2. Тесты TSet

1) Блок TSet_Constructor:

Проверка конструкторов класса TSet с использованием случайных аргументов.

2) Тест Type_Conversion_Operator, Operator:

Проверка оператора преобразования.

3) Блок Elements_Access:

Проверка операций, осуществляющих доступ к элементам и работу с ними.

4) Блок SetTheoretic_Operations:

Проверка теоретико-множественных операций.

5) Блок TSet_Input_Output:

Проверка работы ввода и вывода.

6) Тест TSet_Exceptions_Test, SetTheoretic_Operations_invalid_argument:

Тестирование обработки исключений в теоретико-множественных операциях при передаче неправильного аргумента.