ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ НИЖЕГОРОДСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМ. Н.И. ЛОБАЧЕВСКОГО»

Отчет по лабораторной работе №1 по дисциплине «Алгоритмы и структуры данных»

Выполнил:

Студент Ненев А.Е.

Группа 3824Б1ФИ2

Нижний Новгород

2025г

1) Постановка задачи

Реализация структуры данных для хранения множества – TSet, с использованием битовых полей – TBitField и проверка их корректной работы с помощью тестов Google Test.

2) Описание реализации программы

2.1) Класс битовая строка – TBitField

Класс предназначен для работы с битовыми строками – структурой, которая хранит последовательность бит 0 и 1, с возможностью проводить над ней различные операции.

Поля класса:

int BitLen – длина битового поля

TELEM* pMem – память битового поля

int MemLen количество элементов в pMem

Методы класса:

1. **Конструктор TBitField::TBitField(int len)**

Параметры: len — желаемая длина битового поля в битах (> 0).

Описание функционала: Инициализирует поле длины len со всеми битами, равными 0. Вычисляет MemLen = (BitLen + 31) / 32, выделяет динамическую память под массив pMem и обнуляет его.

Сложность: O(len), так как цикл инициализации проходит по MemLen словам (O(len/32) = O(len)).

2. **Конструктор копирования TBitField::TBitField(const TBitField& bf)**

Параметры: bf — копируемый объект.

Описание функционала: Копирует поля BitLen, MemLen и содержимое массива pMem из bf.

Сложность: O(len), где len = bf.BitLen, из-за копирования массива в цикле.

3. **Деструктор TBitField::~TBitField()**

Описание функционала: Освобождает память, выделенную под pMem, и устанавливает указатель в nullptr.

4. **Метод int TBitField::GetMemIndex(const int n) const**

Параметры: n — номер бита.

Возвращаемые значения: Индекс слова в pMem, содержащего бит n (n / (sizeof(TELEM) * 8)).

Описание функционала: Вычисляет позицию слова для доступа к биту n.

5. **Метод TELEM TBitField::GetMemMask(const int n) const**

Параметры: n — номер бита.

Возвращаемые значения: Маска (TELEM) для бита n в его слове (1 << (n % (sizeof(TELEM) * 8))).

Описание функционала: Формирует маску для побитовых операций с битом n.

6. **Метод int TBitField::GetLength(void) const**

Возвращаемые значения: Длина битового поля (BitLen).

Описание функционала: Возвращает текущую длину поля.

7. **Метод void TBitField::SetBit(const int n)**

Параметры: n — номер бита для установки.

Описание функционала: Устанавливает бит n в 1 путем побитового OR слова с маской.

8. **Метод void TBitField::ClrBit(const int n)**

Параметры: n — номер бита для очистки.

Описание функционала: Устанавливает бит n в 0 путем побитового AND слова с инвертированной маской.

9. **Метод int TBitField::GetBit(const int n) const**

Параметры: n — номер бита.

Возвращаемые значения: 1, если бит установлен, иначе 0.

Описание функционала: Проверяет состояние бита n путем AND с маской.

10. **Оператор int TBitField::operator==(const TBitField& bf) const**

Параметры: bf — объект для сравнения.

Возвращаемые значения: 1, если поля равны (длина и все биты), иначе 0.

Описание функционала: Сравнивает длины, затем сравнивает слова pMem.

Сложность: O(MemLen) = O(len/32) = O(len), цикл по словам.

11.Оператор int TBitField::operator!=(const TBitField& bf) const

Параметры: bf — объект для сравнения.

Возвращаемые значения: 1, если поля не равны, иначе 0.

Описание функционала: Передает проверку на operator==, возвращает противоположный результат

Сложность: O(len).

12.Оператор TBitField& TBitField::operator=(const TBitField& bf)

Параметры: bf — присваиваемый объект.

Возвращаемые значения: Ссылка на *this.

Описание функционала: Если объект не тот же, освобождает старую память, копирует размеры и данные из bf.

Сложность: O(len), копирование массива.

13.Оператор TBitField TBitField::operator|(const TBitField& other)

Параметры: other — второй операнд.

Возвращаемые значения: Новый объект с результатом побитового OR (длина — максимум из двух).

Описание функционала: Создает поле максимальной длины; для каждого бита устанавливает 1, если хотя бы в одном операнде 1 (использует GetBit).

Сложность: O(maxLen), где maxLen = max(BitLen, other.BitLen); цикл по битам, каждый GetBit — O(1).

14.Оператор TBitField TBitField::operator&(const TBitField& other)

Параметры: other — второй операнд.

Возвращаемые значения: Новый объект с результатом побитового AND.

Описание функционала: Аналогично |, но устанавливает 1 только если оба бита 1.

Сложность: O(maxLen), цикл по битам.

15.Оператор TBitField TBitField::operator~(void)

Возвращаемые значения: Новый объект с инвертированными битами (в пределах BitLen).

Описание функционала: Инвертирует все слова побитовым ~, затем маскирует лишние биты в последнем (частично заполненном) слове.

Сложность: O(MemLen) = O(len/32), цикл по словам.

16. **Оператор ввода std::istream& operator>>(std::istream& is, TBitField& bf)**

Параметры: is — поток ввода; bf — поле для заполнения.

Возвращаемые значения: Ссылка на is.

Описание функционала: Читает строку из is; длина должна равняться bf.BitLen, содержать только '0'/'1'. Устанавливает биты: строка интерпретируется как биты от старшего (индекс 0 в строке — MSB, соответствует биту BitLen-1) к младшему.

Сложность: O(BitLen), цикл по битам с вызовами SetBit/ClrBit.

17. **Оператор вывода ostream& operator<<(ostream& ostr, const TBitField& bf)**

Параметры: ostr — поток вывода; bf — поле.

Возвращаемые значения: Ссылка на ostr.

Описание функционала: Выводит биты от старшего (BitLen-1) к младшему (0), т.е. MSB первым.

Сложность: O(BitLen), цикл с вызовами GetBit.

2.2) Класс TSet

Класс TSet реализует множество, используя характеристический вектор, представленный в виде класса TBitField. Каждый элемент соответствует одному биту в битовой строке. 0 – элемент отсутствует в множестве, 1 – элемент присутствует в множестве.

Поля класса:

Int MaxPower – максимальная мощность множества – размер универса

TBitField BitField – характеристический вектор в виде битовой строки

Методы класса:

1. **Конструктор TSet::TSet(int mp)**

Параметры: mp — максимальная мощность множества (> 0, проверка делегируется в TBitField).

Описание функционала: Инициализирует пустое множество с универсумом 0..mp-1, создавая TBitField(mp) со всеми битами 0.

Сложность: O(mp), из-за инициализации TBitField.

2. **Конструктор копирования TSet::TSet(const TSet& s)**

Параметры: s — копируемый объект.

Описание функционала: Копирует MaxPower и содержимое BitField из s.

Сложность: O(mp), где mp = s.MaxPower, из-за копирования TBitField.

3. **Конструктор TSet::TSet(const TBitField& bf)**

Параметры: bf — битовое поле для инициализации.

Описание функционала: Создает множество на основе bf: MaxPower = bf.GetLength(), BitField копирует bf.

Сложность: O(len), где len = bf.GetLength(), из-за копирования TBitField.

4. **Оператор преобразования TSet::operator TBitField()**

Возвращаемые значения: Копия BitField.

Описание функционала: Позволяет использовать множество как битовое поле.

Сложность: O(mp), копирование TBitField.

5. **Метод int TSet::GetMaxPower(void) const**

Возвращаемые значения: Значение MaxPower.

Описание функционала: Возвращает размер универсума.

6. **Метод void TSet::InsElem(const int Elem)**

Параметры: Elem — элемент для добавления ($0 \leq Elem < MaxPower$).

Описание функционала: Добавляет Elem в множество, устанавливая соответствующий бит в 1.

7. **Метод void TSet::DelElem(const int Elem)**

Параметры: Elem — элемент для удаления.

Описание функционала: Удаляет Elem из множества, устанавливая бит в 0.

8. **Метод int TSet::IsMember(const int Elem) const**

Параметры: Elem — проверяемый элемент.

Возвращаемые значения: 1, если Elem в множестве, иначе 0.

Описание функционала: Проверяет наличие Elem по состоянию бита.

9. Оператор int TSet::operator==(const TSet& s) const

Параметры: s — множество для сравнения.

Возвращаемые значения: 1, если множества равны (по битам BitField), иначе 0.

Описание функционала: Делегирует сравнение в TBitField::operator==.

Сложность: O(mp), как у TBitField::==.

10. Оператор int TSet::operator!=(const TSet& s) const

Параметры: s — множество для сравнения.

Возвращаемые значения: 1, если множества не равны, иначе 0.

Описание функционала: Инвертирует результат ==.

Сложность: O(mp).

11.Оператор TSet& TSet::operator=(const TSet& s)

Параметры: s — присваиваемое множество.

Возвращаемые значения: Ссылка на *this.

Описание функционала: Если объект не тот же, копирует MaxPower и BitField из s.

Сложность: O(mp), копирование TBitField.

12.Оператор TSet TSet::operator+(const int Elem)

Параметры: Elem — добавляемый элемент.

Возвращаемые значения: Новое множество с добавленным Elem.

Описание функционала: Создает копию текущего множества и вызывает InsElem(Elem).

Сложность: O(mp) для копии + O(1) для вставки = O(mp).

13.Оператор TSet TSet::operator-(const int Elem)

Параметры: Elem — удаляемый элемент.

Возвращаемые значения: Новое множество без Elem.

Описание функционала: Создает копию и вызывает DelElem(Elem).

Сложность: O(mp).

14.Оператор TSet TSet::operator+(const TSet& s)

Параметры: s — второе множество.

Возвращаемые значения: Новое множество — объединение (элементы из хотя бы одного).

Описание функционала: Проверяет равенство MaxPower; создает пустое множество и присваивает BitField | s.BitField.

Сложность: O(mp), создание результата + сложность TBitField::| (O(mp)).

15. Оператор TSet TSet::operator*(const TSet& s)

Параметры: s — второе множество.

Возвращаемые значения: Новое множество — пересечение (элементы в обоих).

Описание функционала: Аналогично +, но использует BitField & s.BitField.

Сложность: O(mp).

16. Оператор TSet TSet::operator~(void)

Возвращаемые значения: Новое множество — дополнение (элементы универсума, не в текущем).

Описание функционала: Создает пустое множество и присваивает ~BitField (инверсия в пределах MaxPower).

Сложность: O(mp/32), как у TBitField::~ (цикл по словам).

17. Оператор ввода std::istream& operator>>(std::istream& istr, TSet& s)

Параметры: istr — поток ввода; s — множество для заполнения.

Возвращаемые значения: Ссылку на istr.

Описание функционала: Читает битовую строку и заполняет s.BitField (MSB соответствует элементу MaxPower-1).

Сложность: O(mp), как у TBitField::>>.

18. Оператор вывода std::ostream& operator<<(std::ostream& ostr, const TSet& s)

Параметры: ostr — поток вывода; s — множество.

Возвращаемые значения: Ссылку на ostr.

Описание функционала: Выводит битовую строку s.BitField (MSB первым).

Сложность: O(mp), как у TBitField::<<.

3) Краткие комментарии к тестам

Тесты класса TBitField

- Тесты на исключения в TBitField (TestExcept1-4)

Проверяют выброс исключений (invalid_argument или out_of_range):
Инициализация с отрицательной/нулевой длиной (TBitField(-1)).
Операции ClrBit, SetBit, GetBit с индексом за пределами поля (например, 2*n для поля длины n=10).

- Тесты на отсутствие исключений и корректную работу в TBitField (TestNoExcept1-5)

Проверяют нормальную инициализацию (TBitField(10)) и операции в пределах границ, а также конструктор копирования (TBitField B(T) после ввода строки).

- Тесты ввода в TBitField (TestInput1-3)

Проверяют оператор ввода на корректность.

Корректный ввод строки длиной n=8 ("11110010") через.

Исключения: неверные символы ("11110021") и неправильная длина ("111100111001").

- Тесты побитового AND (&) в TBitField (Test_And1-8)

Проверяют оператор & для полей равной/разной длины (до 10 битов):
Корректные случаи: равные длины ("11110011" & "11101010" = "11100010"); разные длины (результат max длины, младшие биты дополняются 0).
Некорректные результаты: EXPECT_FALSE для заведомо неверных ожиданий.

- Тесты побитового OR (|) в TBitField (Test_Or1-7)

Аналогично AND: равные/разные длины ("11110000" | "11001100" = "11111100"); неверные ожидания; вариации длин (5-10 битов).

- Тесты инверсии (~) в TBitField (Test_Add1-2)

Описание: Проверяют ~B на полях длины 8: корректное дополнение до 1 ("11110000" == ~"00001111"); неверный результат.

- Тесты присваивания и сравнений в TBitField (Test_Eq, Test_Eq2, Test_Are_Equal1-2, Test_Are_UnEqual)

Присваивание (B = T) и самоприсваивание (T = T).
Равенство пустых/заполненных полей одинаковой длины (8-10 битов).
Неравенство ("11110000" != "11110011").

Тесты класса TSet

- Тесты инициализации и исключений в TSet (Test_Set1-5)

Нормальная инициализация (TSet(10)); исключение для отрицательной (TSet(-1)).
Исключения в InsElem, DelElem, IsMember с индексом за пределами (2*n).

- Тесты копирования и сравнений в TSet (Test_Set6, Test_Cmp1-2)

Проверяет корректность работы операторов сравнения и копирования.

- Тесты операций над элементами и множествами в TSet (Test_Oper1-5)

Проверяет корректность выполнения операций над множествами

Добавление/удаление элемента (S + 2, S - 2).

Объединение (S + S1 = все биты 1).

Пересечение (S * S1 = пустое).

Дополнение (~S).