федеральное государственное бюджетное образовательное учреждение высшего образования
«Вологодский государственный университет»

Институт математики, естественных и компьютернных наук

Кафедра автоматики и вычислительной техники

Отчет по лабораторной работе №4

Дисциплина: «Программирование»

09.03.04	**43.10**		**01**	**2025**
Код направления подготовки/ специальности	Код выпускающей кафедры	Регистрационный номер по журналу	Код формы обучения	Год

Руководитель	доц., Кочкин Дмитрий Валерьевич
Выполнил студент	Тестов Максим Олегович
Группа, курс	4Б09 РПС-21
Дата сдачи	
Дата защиты	
Оценка по защите	

Вологда 2025

1. Битовый массив

Создайте класс BitArray, реализующий структуру данных "битовый массив".

```
#include <iostream>
#include "BitArray.h"
using namespace std;

class BitArray
{
public:
    BitArray(int n);
    BitArray(const BitArray &b);
    ~BitArray();

    int size() const;
    int operator [] (const unsigned int i) const;
    void setbit(int i, int v);
    BitArray& operator = (const BitArray &b);
    BitArray operator &(const BitArray &b) const;
    BitArray operator | (const BitArray &b) const;
    BitArray operator ~ () const;
    friend ostream& operator << (ostream &os, const BitArray &b);
    bool operator == (const BitArray &b) const;
    bool operator != (const BitArray &b) const;

private:
    int* bit_arr;
    int size_arr;
};

BitArray::BitArray(int n): size_arr(n > 0 ? n : 0){
    bit_arr = new int[size_arr]();
}

BitArray::BitArray(const BitArray &b) : size_arr(b.size_arr){
    bit_arr = new int[size_arr];
    for (int i = 0; i < size_arr; ++i)
        bit_arr[i] = b.bit_arr[i];
}

BitArray::~BitArray(){
    delete[] bit_arr;
}
```

```
int BitArray::size() const{
    return size_arr;
}

int BitArray::operator[](const unsigned int i) const {
    if (i >= static_cast<unsigned int>(size_arr)) {
        return 0;
    }
    return bit_arr[i];
}

void BitArray::setbit(int i, int v){
    if (i >= 0 && i < size_arr && (v == 0 || v == 1)) {
        bit_arr[i] = v;
    }
}

BitArray& BitArray::operator = (const BitArray &b){
    if(this == &b) return *this;

    delete [] bit_arr;
    size_arr = b.size_arr;
    bit_arr = new int[size_arr];

    for(int i = 0; i < size_arr; i++)
        bit_arr[i] = b.bit_arr[i];

    return *this;
}

BitArray BitArray::operator&(const BitArray &b) const {
    int max_size = (size_arr > b.size_arr) ? size_arr : b.size_arr;
    BitArray result(max_size);

    for(int i = 0; i < max_size; i++) {
        int bit1 = (i < size_arr) ? bit_arr[i] : 0;
        int bit2 = (i < b.size_arr) ? b.bit_arr[i] : 0;
        result.bit_arr[i] = bit1 & bit2;
    }

    return result;
}
```

```
BitArray BitArray::operator|(const BitArray &b) const {
    int max_size = (size_arr > b.size_arr) ? size_arr : b.size_arr;
    BitArray result(max_size);

    for(int i = 0; i < max_size; i++) {
        int bit1 = (i < size_arr) ? bit_arr[i] : 0;
        int bit2 = (i < b.size_arr) ? b.bit_arr[i] : 0;
        result.bit_arr[i] = bit1 | bit2;
    }

    return result;
}

BitArray BitArray::operator ~ () const{
    BitArray result(size_arr);
    for (int i = 0; i < size_arr; i++) {
        result.bit_arr[i] = (bit_arr[i] == 1) ? 0 : 1;
    }
    return result;
}

ostream& operator << (ostream &os, const BitArray &b){
    for(int i = 0; i < b.size(); i++)
        os << b.bit_arr[i];
    return os;
}

bool BitArray::operator == (const BitArray &b) const{
    if(size_arr != b.size_arr) return false;

    for(int i = 0; i < size_arr; i++) {
        if(bit_arr[i] != b.bit_arr[i])
            return false;
    }
    return true;
}

bool BitArray::operator != (const BitArray &b) const{
    return !(*this == b);
}
// #include "bit-array-test.h"
```

2. Битовый массив-2

Дополнительное задание для тех, кому интересен C++.
Условие задачи такое же, как и в задаче "Битовый массив только вместо функции setbit используется операция индексирования слева от знака присваивания.

Пример исходных данных

```
BitArray a(10);
a[2] = 1;
cout << a << endl;
```

Пример результата

```
0010000000
```

Решение:

```
int& BitArray::operator[](const unsigned int i) {
    static int fake = 0;
    if (i >= static_cast<unsigned int>(size_arr)) {
        return fake;
    }
    return bit_arr[i];
}
```