МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

« МИРЭА Российский технологический университет»

РТУ МИРЭА

Институт Информационных технологий

Кафедра Вычислительной техники

УЧЕБНОЕ ЗАДАНИЕ

по дисциплине

« Объектно-ориентированное программирование»

Наименование задачи:

« Задача 9_1_2 »

С тудент группы	ИКБО-27-21	Шевелёв И.А.
Руководитель практики	Ассистент	Морозов В.А.
Работа представлена	«___»_____ 2022 г.	_____
		(подпись студента)
Оценка		_____
		(подпись руководителя)

Москва 2022

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Постановка задачи

Перегрузка побитовых логических операции

Задан элемент, состоящий из ячейки памяти данных **объемом один байт** и шаблона активных битов (размер также равен 1 байту). Между данными из ячеек памяти двух элементов можно выполнить побитовые логические операции умножения и сложения. От каждого элемента в операциях участвуют только те биты данных, которые соответствуют шаблону активных битов элемента.

Работа с элементами выполняется следующим образом. Первоначально создаём элементы, определяем для них содержимое ячейки памяти и значение шаблона в шестнадцатеричной системе счисления. Далее описываем логические выражения, включающие эти элементы.

Написать программу, которая моделирует работу с элементами.

В основной программе реализовать алгоритм:

1. Ввод количества элементов n.
2. В цикле для каждого элемента вводится исходное значение ячейки памяти и значение шаблона активных битов. Далее создается объект, в конструктор которого передаются значения памяти и шаблона. Каждому объекту присваивается свой номер от 1 до n.
3. В цикле, последовательно и построчно, вводится «номер первого объекта»«символ логической операции & или |»«номер второго объекта»
4. После каждого нового ввода логического выражения выполняется логическая операция, результат записывается в ячейку памяти первого элемента (объекта).
5. Цикл завершается в тот момент, когда на ввод больше нет данных.
6. Выводится результат последней операции в шестнадцатеричном формате.

Количество элементов больше или равно 2.
Использовать перегрузку логических побитовых операций, реализовав в составе описания класса.

Пояснения.
Значения в пояснении заданы в шестнадцатеричной системе счисления.
Значение логической единицы (1) в шаблоне задаёт активный бит значения из ячейки памяти. Если значение шаблона равно 15, то активными будут считаться 4-й, 2-й и 0-й биты значения из ячейки памяти.
В логической операции между двумя элементами участвуют только те активные биты ячеек памяти, позиции которых совпадают у обоих элементов (находятся на пересечении). Например, если значение шаблона одного элемента равно 0F, а другого 0C, то в логической операции участвуют только 3-й и 2-й биты обоих значений. Соответственно, при записи результата в первый элемент изменениям подвергаются только те биты, которые участвовали в операции.

Первый элемент e1: значение памяти 8F, значение шаблона 0F.
Второй элемент e2: значение памяти 02, значение шаблона 01.
Операция e1 & e2. Значение первого элемента равно 8E,

Первый элемент e1: значение памяти 8F, значение шаблона 0F.
Второй элемент e2: значение памяти 02, значение шаблона F0.
Операция e1 & e2. Значение первого элемента равно 8F,

Описание входных данных

Первая строка содержит значение количества элементов n: «Натуральное значение»

Далее n строк содержат «Шестнадцатеричное значение»«Шестнадцатеричное значение»

Начиная с n + 2 строки: «Натуральное значение»«Знак операции»«Натуральное значение»

Описание выходных данных

«Шестнадцатеричное значение»

Метод решения

Основная программа:

Целочисленный тип данных

Символьный тип данных

Объекты ввода/вывода потока данных (cin/cout библиотеки <iostream>)

Цикл while, for

Условный оператор if

Вектор библиотеки <vector>

Класс Hex:

Модификатор доступа public, private

Символьный тип данных

Конструктор - для возможности объявления объекта

Параметризированный конструктор - для передачи значений в свойства объекта

Побитовые операторы &, |

Методы getId, operator &=, operator |= - перегрузка побитовых операций &=, |=, и метод перевода в целочисленный тип данных

Номер класса	Класс	Модификатор доступа	Описание	Номер	Коментарий
1	Hex	public	Перегрузка побитовых операций		

Описание алгоритма

Согласно этапам разработки, после определения необходимого инструментария в разделе «Метод», составляются подробные описания алгоритмов для методов классов и функций.

Класс объекта: Hex

Модификатор доступа: public

Метод: operator &=

Функционал: Выполнение побитовой операции &

Параметры: Hex& other

Возвращаемое значение: Hex&, *this

Алгоритм метода представлен в таблице 2.

Таблица 2. Алгоритм метода operator &= класса Hex

№	Предикат	Действия	№ перехода	Комментарий
1		Инцилизация символьного типа данных templ1 с значением base, templ2 с значение other.base	2	
2		Инцилизация символьного типа данных tmp с значение templ & other.templ	3	
3		Присвоение templ1 значение templ1 & tmp	4	
4		Присвоение templ2 значение templ2 & tmp	5	
5		Присвоение templ2 значение	6	

		templ2 & templ1		
6		Присвоение base значение base & ~tmp	7	
7		Присвоение base значение base | templ2	8	
8		Возрат *this	∅	

Класс объекта: Hex

Модификатор доступа: public

Метод: operator |=

Функционал: Выполнение побитовой операции |

Параметры: Hex& other

Возвращаемое значение: Hex&, *this

Алгоритм метода представлен в таблице 3.

Таблица 3. Алгоритм метода operator |= класса Hex

№	Предикат	Действия	№ перехода	Комментарий
1		Присвоение свойству base значение base | (other.base & (templ & other.templ))	2	
2		Возрат *this	∅	

Класс объекта: Hex

Модификатор доступа: public

Метод: getId

Функционал: Возрат значение в типе int

Параметры: нет

Возвращаемое значение: int, Целое число

Алгоритм метода представлен в таблице 4.

Таблица 4. Алгоритм метода getId класса Hex

№	Предикат	Действия	№ перехода	Комментарий
1		return static_cast<int>(base)	∅	

Функция: main

Функционал: Главная функция программы

Параметры: нет

Возвращаемое значение: int, Код возрата

Алгоритм функции представлен в таблице 5.

Таблица 5. Алгоритм функции main

№	Предикат	Действия	№ перехода	Комментарий
1		Объявление целочисленных переменных n, num1, num2, num_obj1, num_obj2	2	
2		Объявление переменной symbol символьного типа данных	3	
3		Ввод значения n	4	
4		Вызов метода unsetf(ios::dec) объекта cin	5	
5		Вызов метода setf(ios::hex) объекта cin	6	
6		Объявление объекта mas класса vector	7	
7	i < n	Инкремент i	8	

			10	
8		Ввод значений num1, num2	9	
9		Присвоение элементу массива mas[i] объект Hex(num1, num2)	7	
10	cin.get() != EOF		11	
			13	
11		Ввод значений num_obj1, symbol, num_obj2	12	
12	symbol == '&'	Присвоение элементу массива mas[num_obj1-1] значение &= mas[num_obj2-1]	10	
	symbol == '|'	Присвоение элементу массива mas[num_obj1-1] значение |= mas[num_obj2-1]	10	
13		Вызов метода unsetf(ios::dec) объекта cin	14	
14		Вызов метода setf(ios::hex | ios::uppercase) объекта cin	15	
15		Вызов метода fill('0') объекта cin	16	
16		Вызов метода width('2') объекта cin	17	
17		Вывод значение метода getId объекта элемента массива mas[num_obj1-1]	∅	

Блок-схема алгоритма

Представим описание алгоритмов в графическом виде на рисунках ниже.

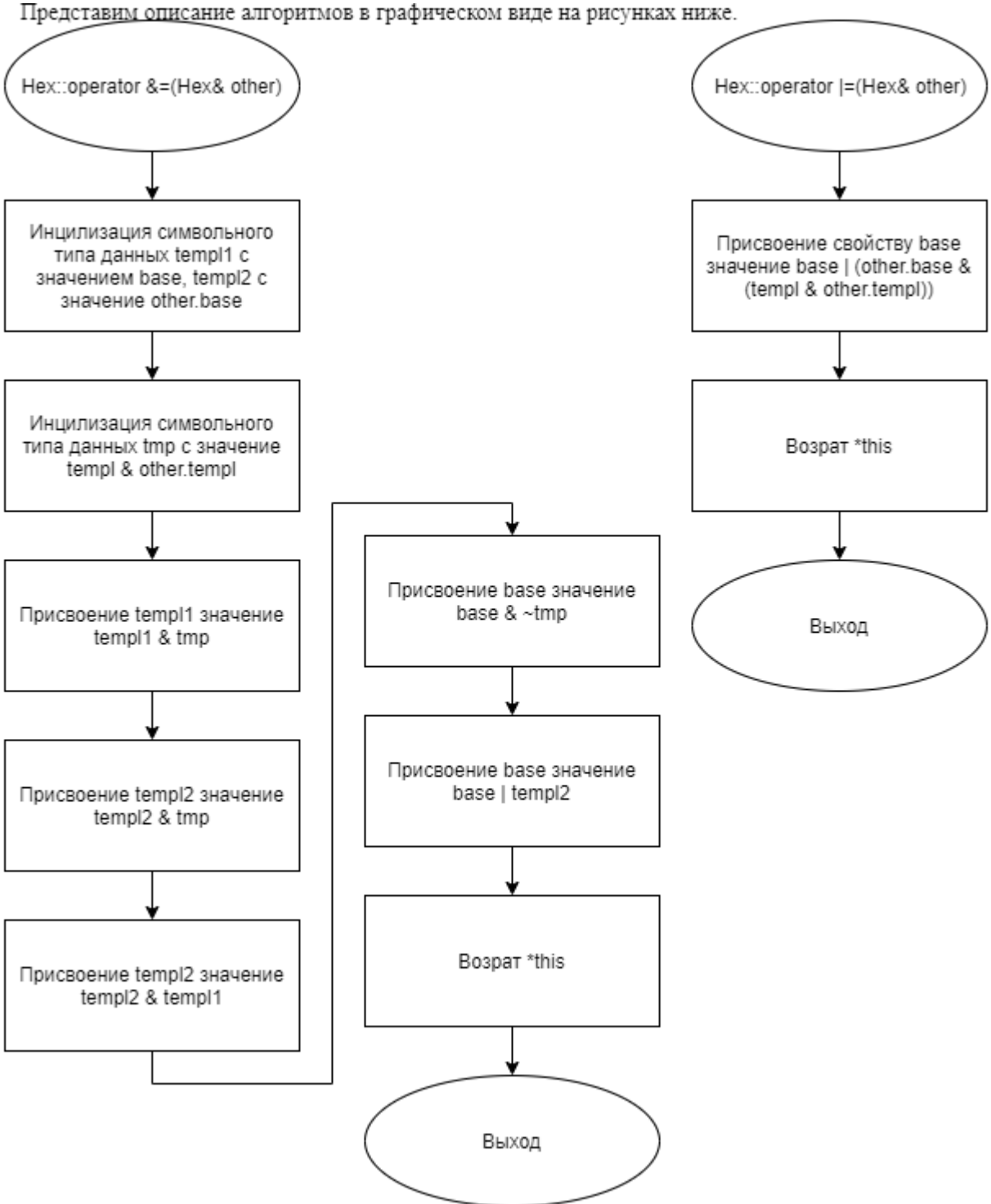

Рис. 1. Блок-схема алгоритма.

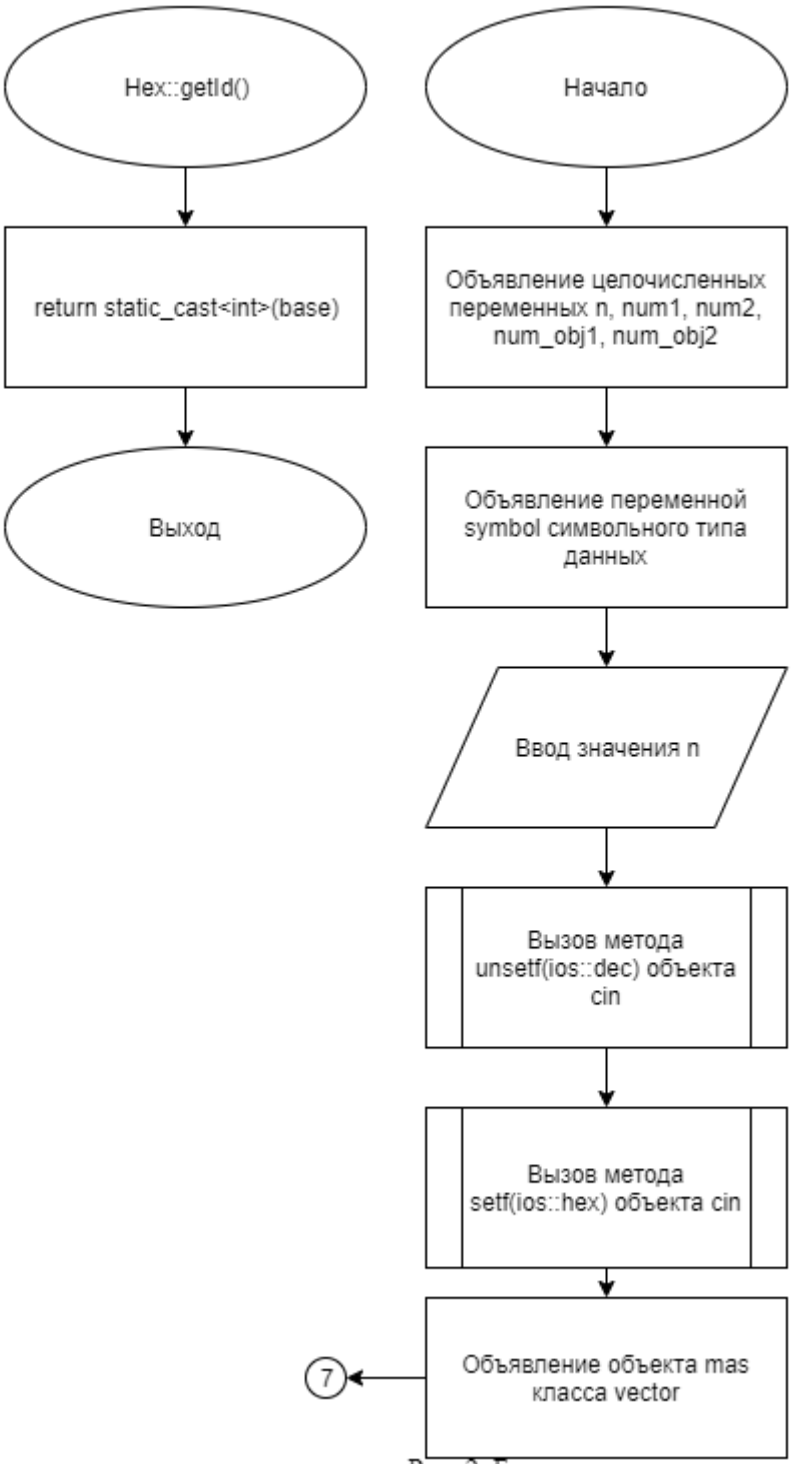

Рис. 2. Блок-схема алгоритма.

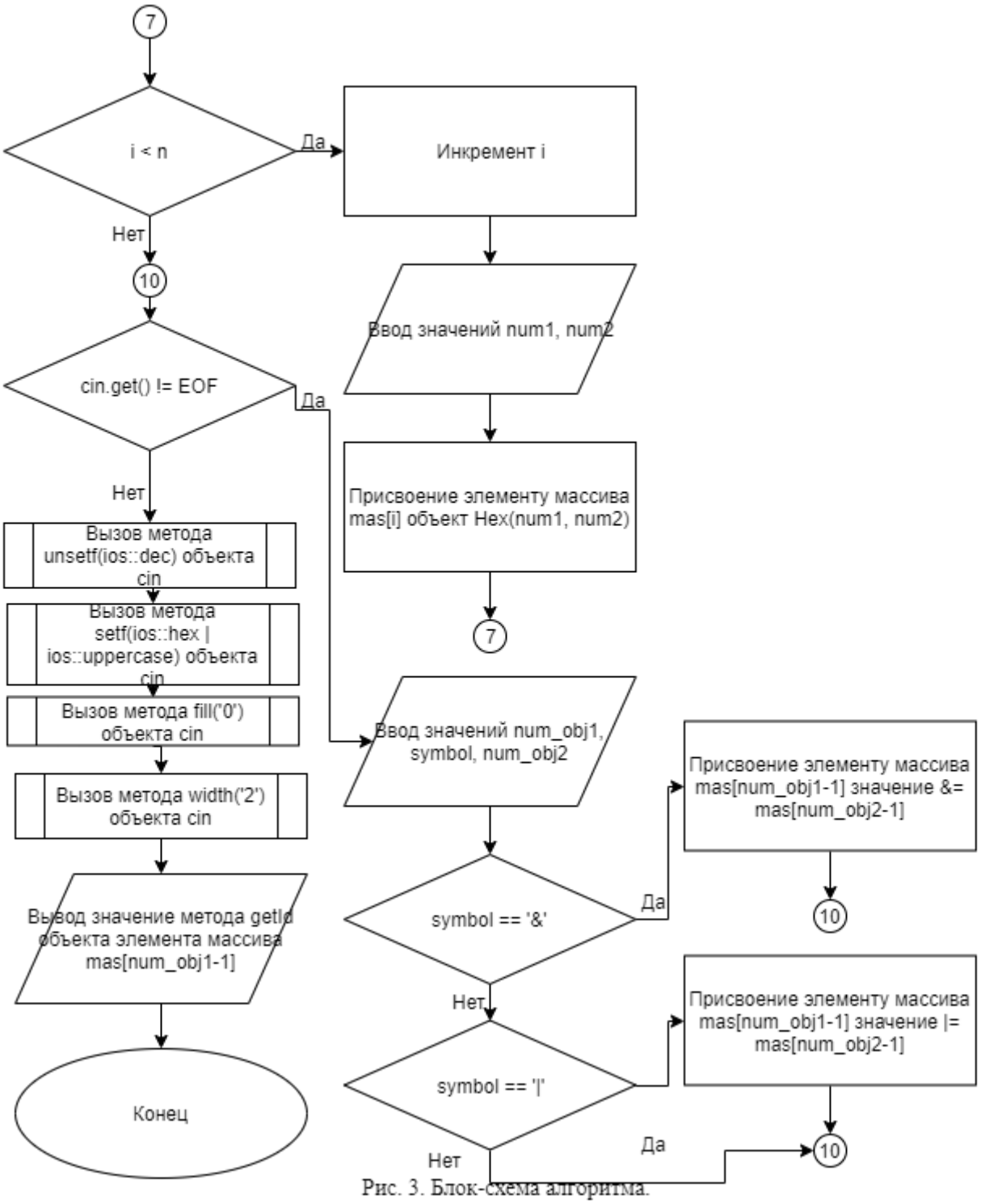

Рис. 3. Блок-схема алгоритма.

Код программы

Программная реализация алгоритмов для решения задачи представлена ниже.

Файл hex.cpp

```
#include "hex.h"

int Hex::getId(){
        return static_cast<int>(base);
}

Hex& Hex::operator &=(Hex& other){
        unsigned char templ1 = base, templ2 = other.base;
        unsigned char tmp = templ & other.templ;
        templ1 &= tmp;
        templ2 &= tmp;
        templ2 &= templ1;

        base &= ~tmp;
        base |= templ2;

        return *this;
}
Hex& Hex::operator |=(Hex& other){
        base = base | (other.base & (templ & other.templ));
        return *this;
}
```

Файл hex.h

```
#ifndef HEX_H
#define HEX_H
#include <iostream>
#include <vector>
#include <stdlib.h>
#include <stdio.h>
using namespace std;

class Hex{
private:
        unsigned char templ, base;
public:
        int getId();
        Hex() {};
        Hex(int base, int templ) : base(static_cast<unsigned char>(base)),
templ(static_cast<unsigned char>(templ)){};
        Hex& operator &=(Hex& other);
        Hex& operator |=(Hex& other);
```

```
};
#endif
```

Файл main.cpp

```
#include "hex.h"

int main()
{
        // program here
        int n, num1, num2, num_obj1, num_obj2;
        char symbol;
        cin >> n;
        cin.unsetf(ios::dec);
        cin.setf(ios::hex);
        vector<Hex> mas(n);
        for(int i = 0; i < n; ++i){
                cin >> num1 >> num2;
                mas[i] = Hex(num1, num2);
        }
        while(cin.get() != EOF){
                cin >> num_obj1 >> symbol >> num_obj2;
                if(symbol == '&'){
                        mas[num_obj1-1] &= mas[num_obj2-1];
                }else if(symbol == '|'){
                        mas[num_obj1-1] |= mas[num_obj2-1];
                }
        }
        cout.unsetf(ios::dec);
        cout.setf(ios::hex | ios::uppercase);
        cout.fill('0');
        cout.width(2);
        cout << mas[num_obj1-1].getId();
        return(0);
}
```

Тестирование

Результат тестирования программы представлен в следующей таблице.

Входные данные	Ожидаемые выходные данные	Фактические выходные данные
4 0F 14 AFD 8F 12 4D AF DB0 1 & 2 2 | 3 4 & 2	AF	AF
3 0F 14 AFD 8F 12 4D 1 & 2 2 | 3 3 & 1	12	12
2 0F 14 AFD 8F 1 & 2 2 & 1 1 | 2	0F	0F

ЗАКЛЮЧЕНИЕ

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ (ИСТОЧНИКОВ)

1. Васильев А.Н. Объектно-ориентированное программирование на C++. Издательство: Наука и Техника. Санкт-Петербург, 2016г. 543 стр.

2. Шилдт Г. C++: базовый курс. 3-е изд. Пер. с англ.. — М.: Вильямс, 2017. — 624 с.

3. Методическое пособие для проведения практических заданий, контрольных и курсовых работ по дисциплине «Объектно-ориентированное программирование» [Электронный ресурс] – URL: https://mirea.aco-avrora.ru/student/files/methodichescoe_posobie_dlya_laboratornyh_rabot_3.pdf (дата обращения 05.05.2021).

4. Приложение к методическому пособию студента по выполнению заданий в рамках курса «Объектно-ориентированное программирование» [Электронный ресурс]. URL: https://mirea.aco-avrora.ru/student/files/Prilozheniye_k_methodichke.pdf (дата обращения 05.05.2021).

5. Видео лекции по курсу «Объектно-ориентированное программирование» [Электронный ресурс]. АСО «Аврора».

6. Антик М.И. Дискретная математика [Электронный ресурс]: Учебное пособие /Антик М.И., Казанцева Л.В. — М.: МИРЭА — Российский технологический университет, 2018 — 1 электрон. опт. диск (CD-ROM).