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-13-21	Черномуров С.А.
Руководитель практики	Ассистент	Асадова Ю.С.
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

Для решения задачи используются:

- Оператор цикла со счетчиком for. Используется для циклического создания объектов.
- Оператор цикла с предусловием while. Используется для выполнения операций.
- Объекты стандартных потоков ввода и вывода cin и cout соответственно. Используются для ввода с клавиатуры и вывода на экран.
- Условный оператор if .. else. Используется для выбора операции.
- Функции-операторы "&" и "|". Используются для объектов класса Class.
- **Класс Class:**
    - Свойства/поля:
        - Свойство:
            - Наименование - _byte;
            - Тип - беззнаковый символьный;
            - Модификатор доступа - открытый.
        - Свойство:
            - Наименование - Template;
            - Тип - беззнаковый символьный;
            - Модификатор доступа - открытый.
    - Методы:
        - Метод Class:
            - Функционал - параметризированный конструктор.

Описание алгоритма

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

Конструктор класса: Class

Модификатор доступа: public

Функционал: Параметризированный конструктор

Параметры: Символьные беззнаковые параметры _byte, Template

Алгоритм конструктора представлен в таблице 1.

Таблица 1. Алгоритм конструктора класса Class

№	Предикат	Действия	№ перехода	Комментарий
1		Присвоение свойству _byte текущего объекта значения _byte	2	
2		Присвоение свойству Template текущего объекта значения Template	∅	

Функция: operator&

Функционал: Побитовое "И" значений памяти бита с учетом шаблонов активных битов

Параметры: Ссылка на объект класса Class - e1, ссылка на константный объект класса Class - e2

Возвращаемое значение: Объект класса Class - e1

Алгоритм функции представлен в таблице 2.

Таблица 2. Алгоритм функции operator&

№	Предикат	Действия	№ перехода	Комментарий
1		Объявление символьных беззнаковых переменных byte1,byte2	2	
2		Объявление символьных беззнаковых переменных Template1,Template2	3	
3		Присвоение byte1 значения свойства _byte объекта e1	4	
4		Присвоение byte2 значения свойства _byte объекта e2	5	
5		Присвоение Template1 значения свойства Template объекта e1	6	
6		Присвоение Template2 значения свойства Template объекта e2	7	
7		Объявление с инициализацией символьной беззнаковой переменной temp=Template1&Template2	8	
8		Присвоение свойству _byte объекта e1 значения (temp & (byte1 & byte2)) + ((~(temp)) & byte1)	9	
9		Возврат функцией e1	∅	

Функция: operator|

Функционал: Побитовое "ИЛИ" значений памяти бита с учетом шаблонов активных битов

Параметры: Ссылка на объект класса Class - e1, ссылка на константный объект класса Class - e2

Возвращаемое значение: Объект класса Class - e1

Алгоритм функции представлен в таблице 3.

Таблица 3. Алгоритм функции operator|

№	Предикат	Действия	№ перехода	Комментарий
1		Объявление символьных беззнаковых переменных byte1,byte2	2	
2		Объявление символьных беззнаковых переменных Template1,Template2	3	
3		Присвоение byte1 значения свойства _byte объекта e1	4	
4		Присвоение byte2 значения свойства _byte объекта e2	5	
5		Присвоение Template1 значения свойства Template объекта e1	6	
6		Присвоение Template2 значения свойства Template объекта e2	7	
7		Объявление с инициализацией символьной беззнаковой переменной temp=Template1&Template2	8	
8		Присвоение свойству _byte объекта e1 значения (((temp & byte1) | (temp & byte2)) & temp) + ((~(temp)) & byte1)	9	
9		Возврат функцией e1	∅	

Функция: main

Функционал: Основной алгоритм программы

Параметры: Отсутствуют

Возвращаемое значение: Целочисленный тип данных - код возврата

Алгоритм функции представлен в таблице 4.

Таблица 4. Алгоритм функции main

№	Предикат	Действия	№ перехода	Комментарий
1		Объявление целочисленной переменной n	2	
2		Считывание с клавиатуры значения переменной n	3	
3		Создание контейнера класса vector с типом данных Class - vec	4	
4		Объявление целочисленной переменной с инициализацией i=0	5	Использование i в качестве счетчика
5	Значение i меньше значения n	Объявление целочисленных переменных a,b	6	
			11	Выход из цикла
6		Считывание с клавиатуры значений переменных a,b в шестнадцатеричном виде	7	
7		Объявление символьных беззнаковых переменных с инициализацией byte=a, temp=b	8	
8		Создание объекта obj класса Class с параметрами byte,temp	9	
9		Вызов метода push_back объекта vec с параметром obj	10	
10		Инкрементирование i	5	
11		Объявление целочисленных	12	

		переменных num1,num2		
12		Объявление символьной беззнаковой переменной oper	13	
13	Значения num1,oper,num2 считаны с клавиатуры		14	
			15	Выход из цикла
14	Значение oper равно '&'	Присвоение vec[num1-1] значения vec[num1-1]&vec[num2-1]	13	
		Присвоение vec[num1-1] значения vec[num1-1]| vec[num2-1]	13	
15	Значение vec[num1-1], приведенного к типу int меньше 16	Вывод на экран "0"	16	
			16	
16		Вывод на экран значения свойства _byte объекта vec[num1-1], приведенного к типу int, в шестнадцатеричном формате в верхнем регистре	∅	

Блок-схема алгоритма

Представим описание алгоритмов в графическом виде на рисунках ниже.

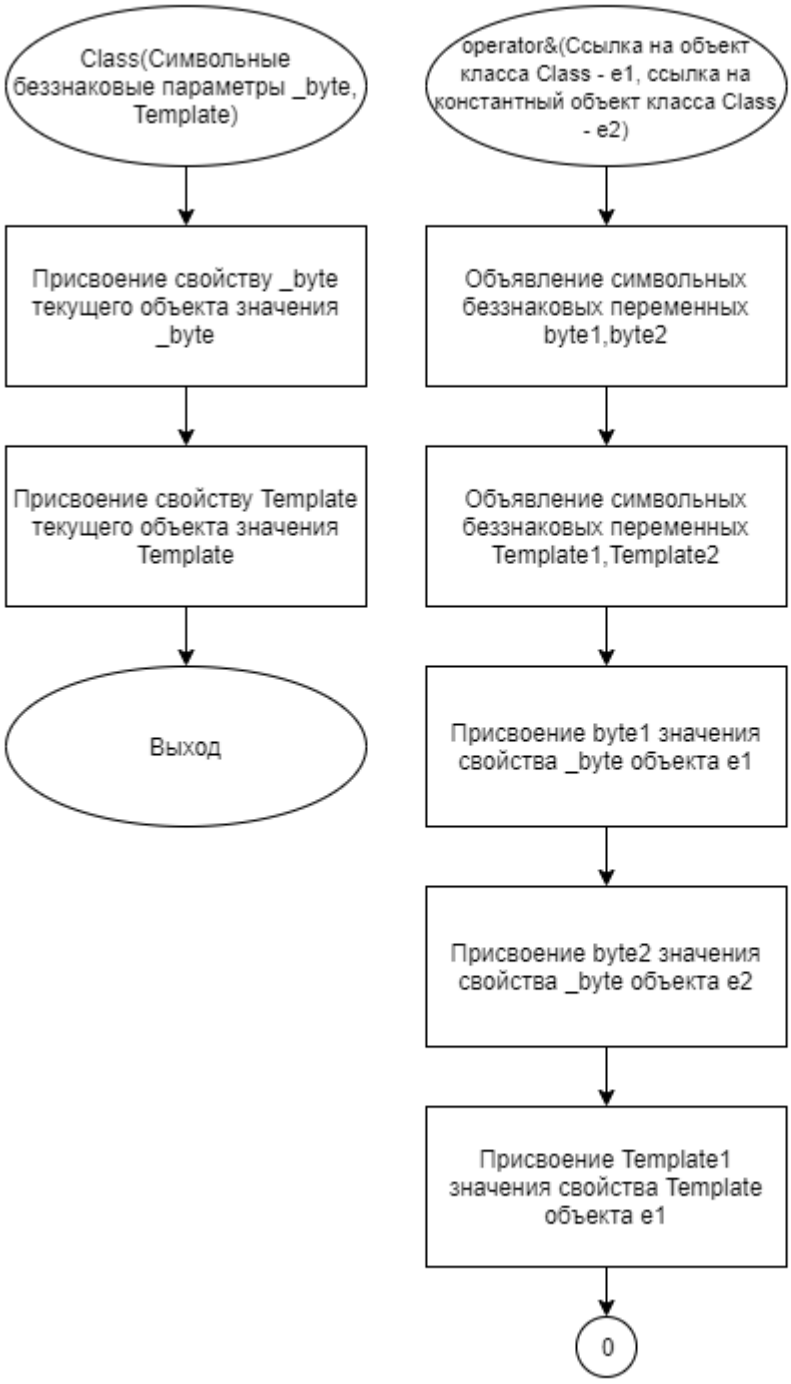

Рис. 1. Блок-схема алгоритма.

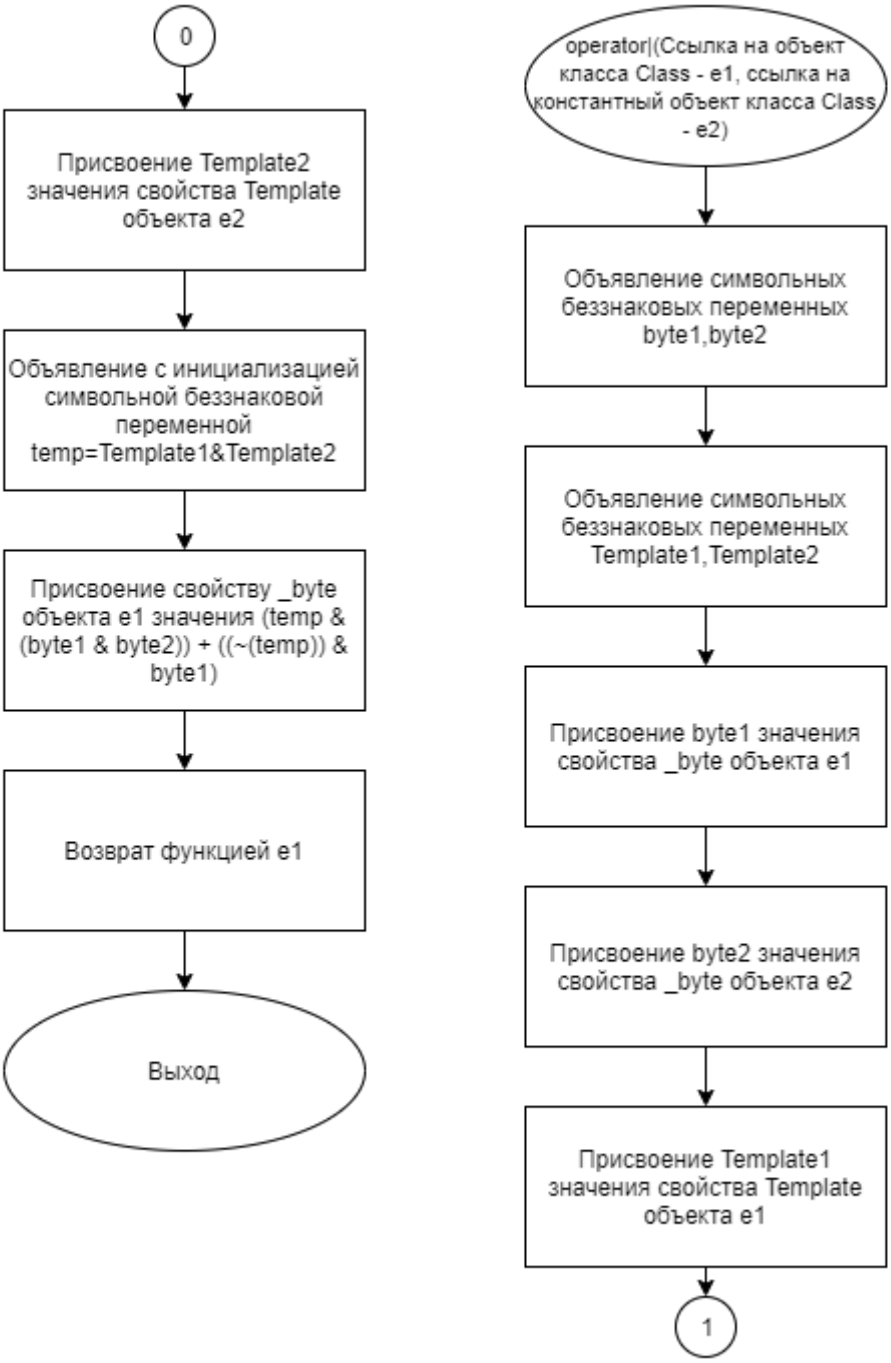

Рис. 2. Блок-схема алгоритма.

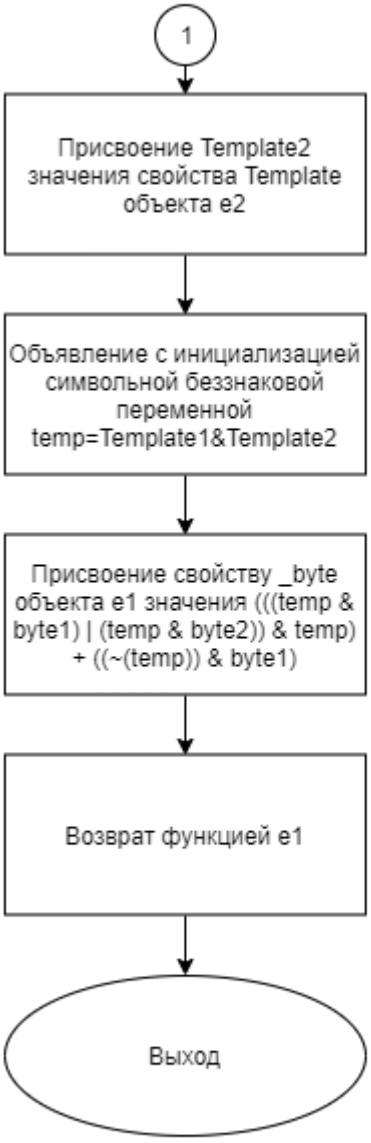

Рис. 3. Блок-схема алгоритма.

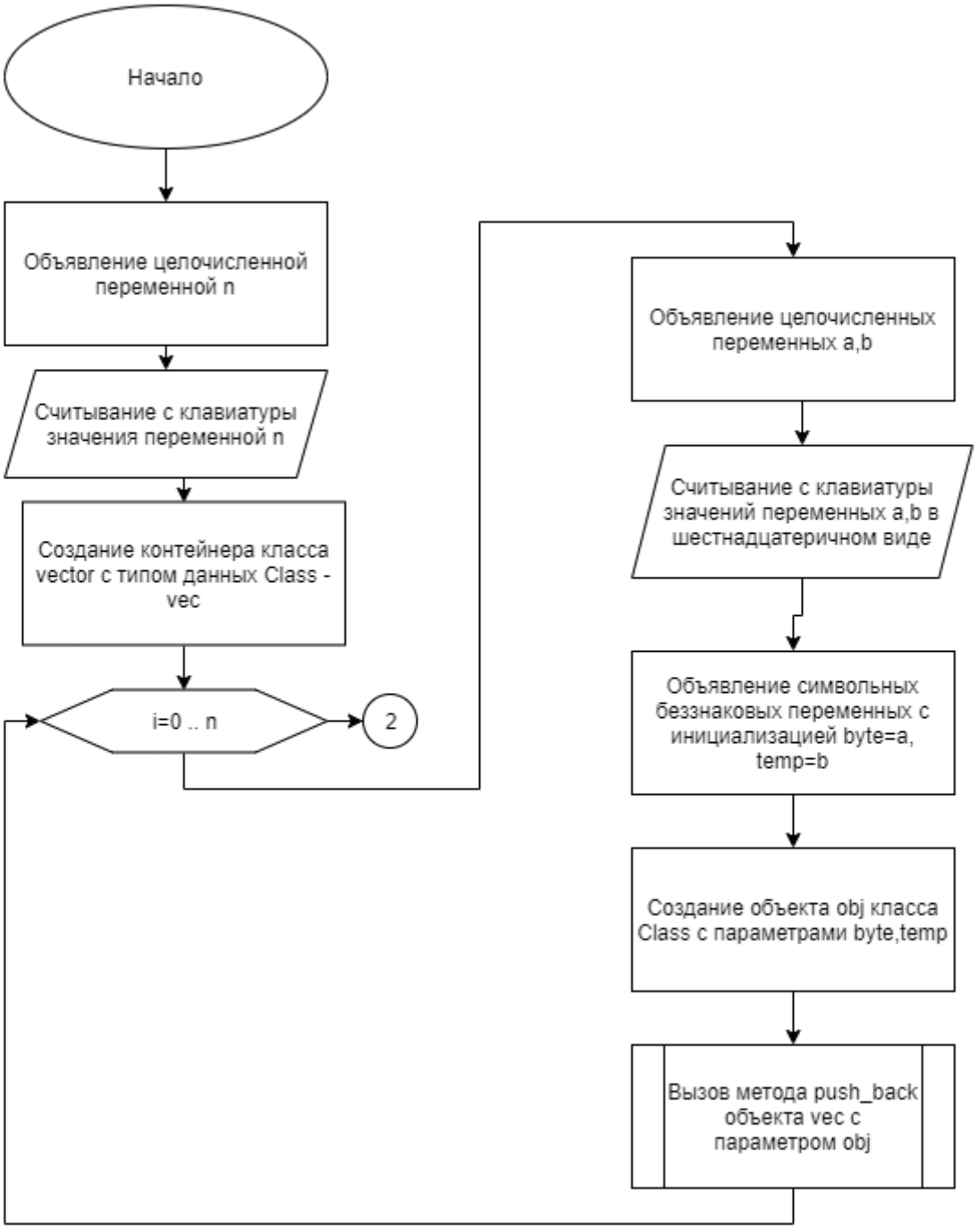

Рис. 4. Блок-схема алгоритма.

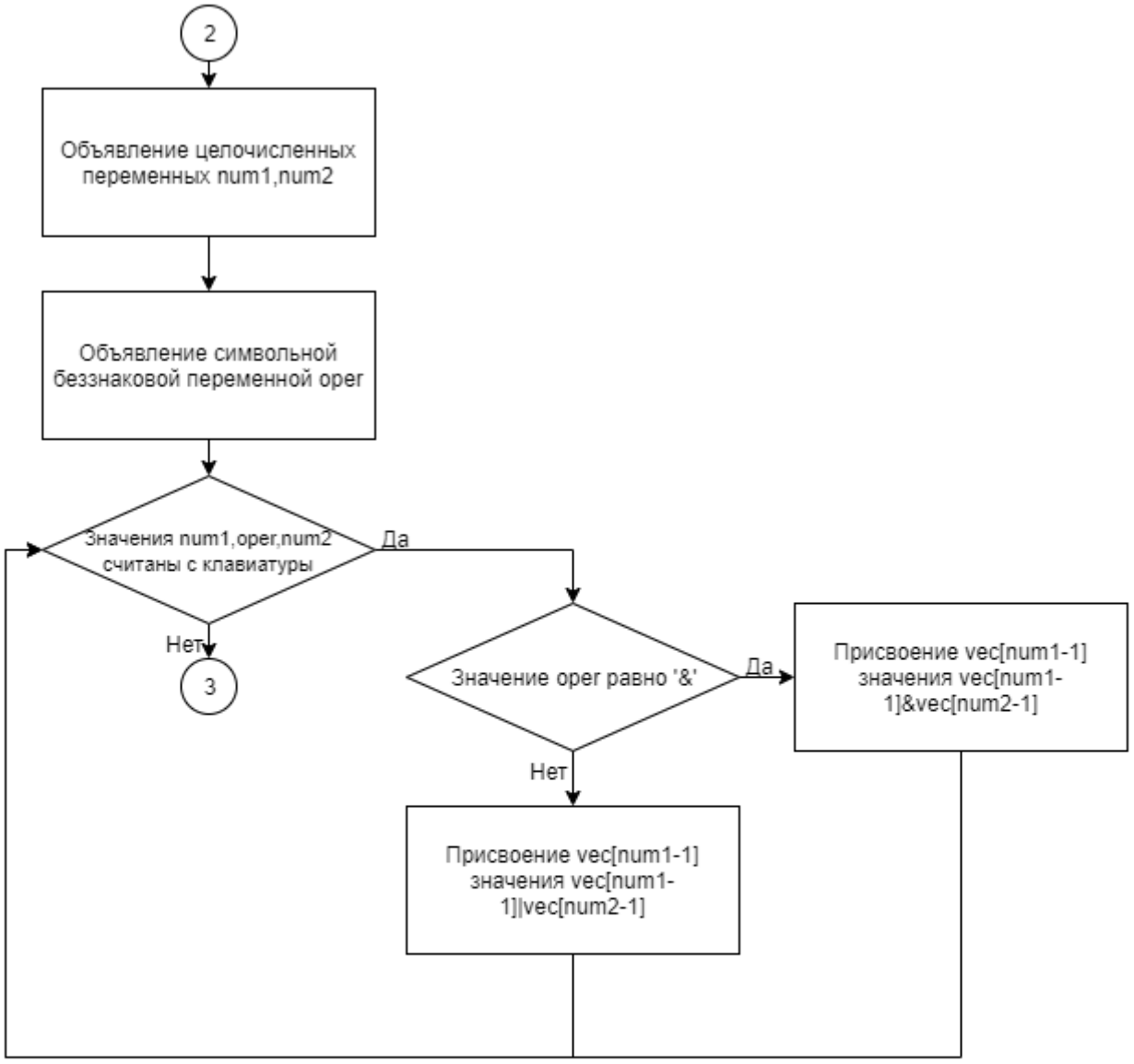

Рис. 5. Блок-схема алгоритма.

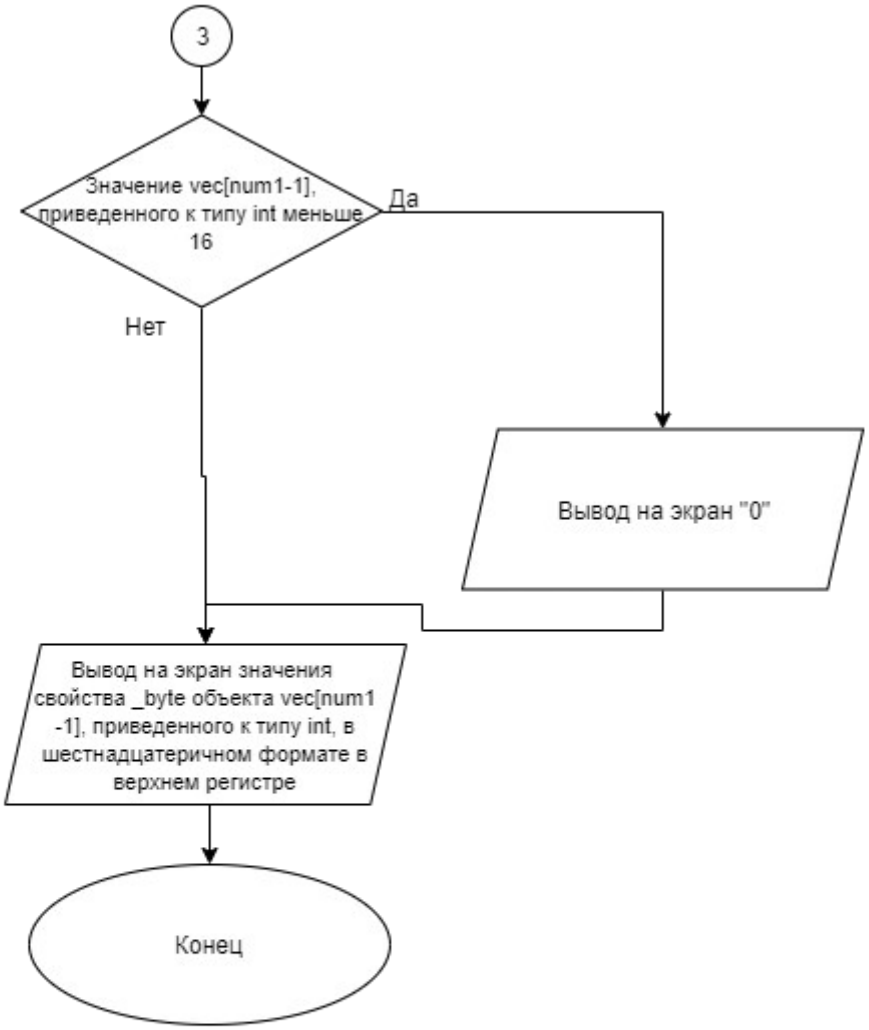

Рис. 6. Блок-схема алгоритма.

Код программы

Программная реализация алгоритмов для решения задачи представлена ниже.

Файл Class.cpp

```
#include "Class.h"

#include <iostream>
using namespace std;

Class::Class(byte _byte, byte Template){
        this->_byte = _byte;
        this->Template = Template;
}

Class operator& (Class &e1, const Class &e2){
        byte byte1, byte2;
        byte Template1, Template2;

        byte1 = e1._byte;
        byte2 = e2._byte;

        Template1 = e1.Template;
        Template2 = e2.Template;

        byte temp = Template1 & Template2;

        e1._byte=(temp & (byte1 & byte2)) + ((~(temp)) & byte1);
        return e1;
}

Class operator| (Class &e1, const Class &e2){
        byte byte1, byte2;
        byte Template1, Template2;

        byte1 = e1._byte;
        byte2 = e2._byte;

        Template1 = e1.Template;
        Template2 = e2.Template;

        byte temp = Template1 & Template2;

        e1._byte=(((temp & byte1) | (temp & byte2)) & temp) + ((~(temp)) &
byte1);
        return e1;
}
```

Файл Class.h

```
#ifndef CL_H
#define CL_H
typedef unsigned char byte;
#include <string>

class Class{
        public:
                byte _byte;
                byte Template;
                Class (unsigned char byte, unsigned char Template);
};

Class operator& (Class &e1, const Class &e2);
Class operator| (Class &e1, const Class &e2);

#endif
```

Файл main.cpp

```
#include "Class.h"
#include <iostream>
#include <vector>
typedef unsigned char byte;
using namespace std;
int main()
{
        int n;
        cin>>n;
        vector <Class> vec;
        for (int i=0;i<n;i++){
                int a,b;
                cin>>hex>>a>>b;

                byte byte=a,temp=b;

                Class obj(byte,temp);
                vec.push_back(obj);
        }
        int num1,num2;
        byte oper;
        while (cin>>num1>>oper>>num2){
                if (oper=='&') vec[num1-1]=vec[num1-1]&vec[num2-1];
                else vec[num1-1]=vec[num1-1]|vec[num2-1];
        }
        if ((int)vec[num1-1]._byte<16) cout<<"0";
        cout<<hex<<uppercase<<(int)vec[num1-1]._byte;
        return 0;
}
```

Тестирование

Результат тестирования программы представлен в следующей таблице.

Входные данные	Ожидаемые выходные данные	Фактические выходные данные
3 8F 0F 66 77 35 4A 1 | 3 2 | 3 3 & 2	35	35
2 8F 0F 02 01 1 & 2	8E	8E
2 8F 0F 02 F0 1 & 2	8F	8F
3 8F 0F 66 77 00 00 1 | 3 2 | 3 3 & 2	00	00

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).