Здесь будет титульник, листай ниже

СОДЕРЖАНИЕ

1 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,

1.1 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»

1.2 Описание выходных данных

«Шестнадцатеричное значение»

2 МЕТОД РЕШЕНИЯ

Для решения задачи используется:

- объект класса logic предназначен для ;
- функция main для главная функция программы;
- библиотека iostream;
- библиотека string;
- библиотека vector;
- библиотека iomanip;
- объект стандартного потока ввода cin с клавиатуры;
- объект стандартного потока вывода cout на экран;
- цикл со счётчиком for;
- цикл с предусловием while.

Класс logic:

- свойства/поля:
    - поле значение памяти:
        - наименование — data;
        - тип — unsigned char;
        - модификатор доступа — public;
    - поле значение шаблона:
        - наименование — mask;
        - тип — unsigned char;
        - модификатор доступа — public;
- функционал:
    - метод logic — параметриированный конструктор класса logic;
    - метод operator& — метод перегрузки унарного оператора &;
    - метод operator| — метод перегрузки унарного оператора |.

3 ОПИСАНИЕ АЛГОРИТМОВ

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

3.1 Алгоритм конструктора класса logic

Функционал: параметриированный конструктор класса logic.

Параметры: нет.

Алгоритм конструктора представлен в таблице 1.

Таблица 1 – Алгоритм конструктора класса logic

№	Предикат	Действия	№ перехода
1		Присвоение полю data переданного значения data_other	2
2		Присвоение полю mask переданного значения mask_other	∅

3.2 Алгоритм метода operator& класса logic

Функционал: метод перегрузки унарного оператора &.

Параметры: нет.

Возвращаемое значение: Объект класса logic.

Алгоритм метода представлен в таблице 2.

Таблица 2 – Алгоритм метода operator& класса logic

№	Предикат	Действия	№ перехода
1		Инициализация беззнаковой символьной переменной c_mask значением (mask & other.mask)	2
2		Инициализация беззнаковой символьной переменной result значением (data & other.mask)	3

№	Предикат	Действия	№ перехода
3		Возврат объекта класса logic при помощи вызова параметризированного конструктора класса logic с передаваемыми параметрами ((data & ~c_mask) | result) и mask	∅

3.3 Алгоритм метода operator| класса logic

Функционал: метод перегрузки унарного оператора |.

Параметры: нет.

Возвращаемое значение: Объект класса logic.

Алгоритм метода представлен в таблице 3.

Таблица 3 – Алгоритм метода operator| класса logic

№	Предикат	Действия	№ перехода
1		Инициализация беззнаковой символьной переменной c_mask значением (mask & other.mask)	2
2		Инициализация беззнаковой символьной переменной result значением (data | other.mask) & c_mask	3
3		Возврат объекта класса logic при помощи вызова параметризированного конструктора класса logic с передаваемыми параметрами ((data & ~c_mask) | result) и mask	∅

3.4 Алгоритм функции main

Функционал: главная функция программы.

Параметры: нет.

Возвращаемое значение: целое, индикация корректности работы программы.

Алгоритм функции представлен в таблице 4.

Таблица 4 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		Объявление целочисленной переменной n	2
2		Ввод значения переменной n	3
3		Объявление целочисленных беззнаковых переменных data и mask	4
4		Объявление вектора elements	5
5		Инициализация целочисленной переменной i значением 0	6
6	Значение переменной i меньше n	Ввод значений переменных data и mask	7
	Значение переменной i не меньше n		9
7		Добавление в вектор elements объекта класса logic при помощи вызова параметризированного конструктора с передаваемыми параметрами data и mask	8
8		Увеличение значения параметра i на 1	9
9		Объявление целочисленных переменных a и b	10
10		Объявление символьной переменной operation	11
11		Объявление указателя ans на класс logic	12
12	Есть a, operation, b	Ввод значений переменных a, operation, b	13
	Не есть a, operation, b		∅
13		Уменьшение значения переменных a и b на 1	14
14	Значение переменной operation равняется "|"?	Результат операции | между элементами вектора elements с индексами a и b	15

№	Предикат	Действия	№ перехода
	Значение переменной operation равняется "&"?	Результат операции & между элементами вектора elements с индексами a и b	15
15		Присвоение указателю ans адреса элемента a вектора elements	16
16		Вывод: <<свойство data объекта,адрес которого хранит указатель ans>>	∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-2.

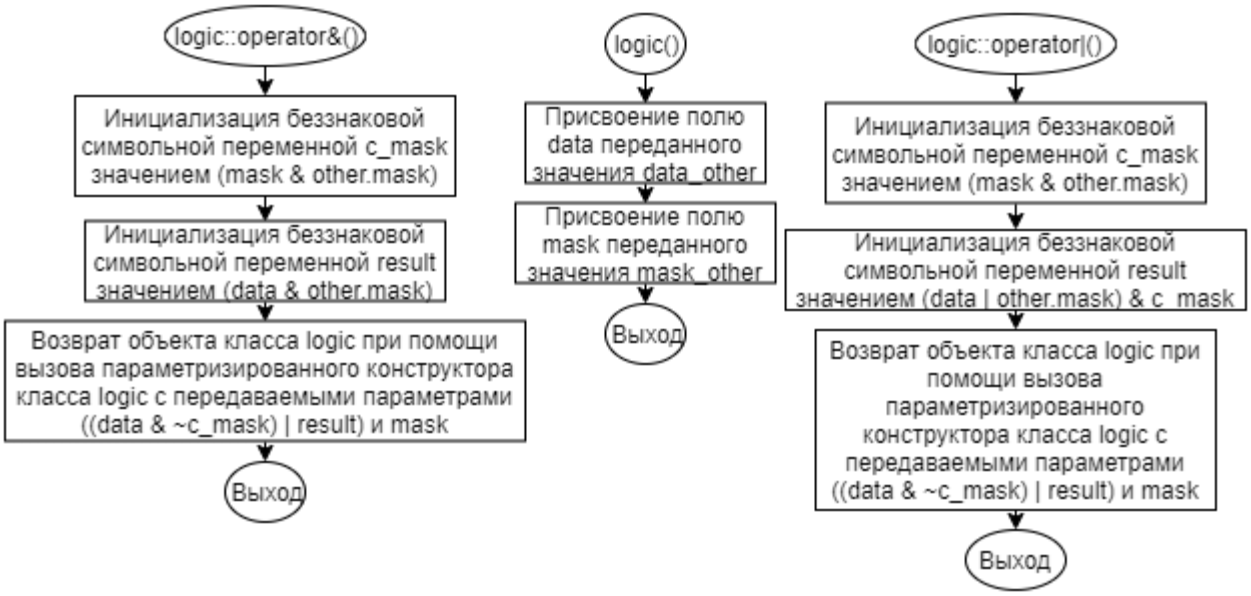

Рисунок 1 – Блок-схема алгоритма

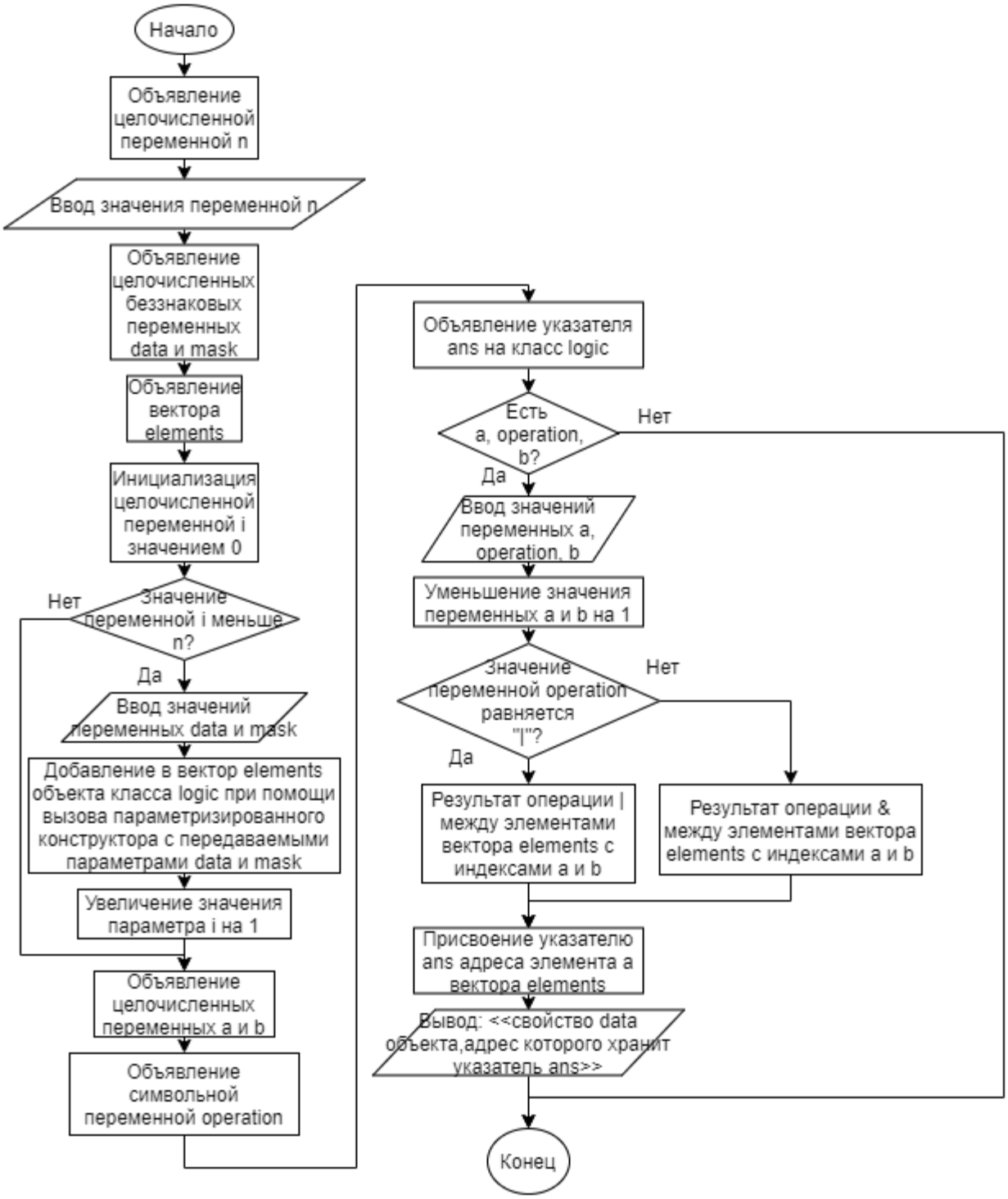

Рисунок 2 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл logic.cpp

Листинг 1 – logic.cpp

```
#include "logic.h"

logic :: logic (unsigned char data_other, unsigned char mask_other)
{
   data = data_other;
   mask = mask_other;
}

logic logic :: operator & (logic& other)
{
   unsigned char c_mask = mask & other.mask;
   unsigned char result = (data & other.data) & c_mask;
   return logic((data & ~c_mask) | result, mask);
}

logic logic :: operator | (logic& other)
{
   unsigned char c_mask = mask & other.mask;
   unsigned char result = (data | other.data) & c_mask;
   return logic((data & ~c_mask) | result, mask);
}
```

5.2 Файл logic.h

Листинг 2 – logic.h

```
#ifndef __LOGIC__H
#define __LOGIC__H

class logic
{
   public:
      unsigned char data;
      unsigned char mask;
```

```
        logic(unsigned char data_other, unsigned char mask_other);
        logic operator & (logic& other);
        logic operator | (logic& other);
};


#endif
```

5.3 Файл main.cpp

Листинг 3 – main.cpp

```
#include "logic.h"

#include <string>
#include <vector>
#include <iostream>
#include <iomanip>

using namespace std;

int main()
{
   int n;
   cin >> n;
   unsigned int data, mask;
   vector <logic> elements;
   for (int i = 0; i < n; ++i)
   {
      cin >> hex >> data >> mask;
      elements.push_back(logic(data, mask));
   }
   logic* ans;
   int a, b;
   char operation;
   while (cin.peek() != EOF)
   {
      cin >> a >> operation >> b;
      --a;
      --b;
      if (operation == '|'){elements[a] = elements[a] | elements[b];}
      else if (operation == '&'){elements[a] = elements[a] & elements[b];}
      ans = &elements[a];
   }
   cout   <<   hex   <<   setw(2)   <<   setfill('0')   <<   uppercase   <<
static_cast<int>(ans -> data);
   return 0;
}
```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 5.

Таблица 5 – Результат тестирования программы

Входные данные	**Ожидаемые выходные данные**	**Фактические выходные данные**
2 1C 5A EA 3D 1 | 2	1C	1C

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ГОСТ 19 Единая система программной документации.

2. Методическое пособие студента для выполнения 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).

3. 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).

4. Шилдт Г. C++: базовый курс. 3-е изд. Пер. с англ.. — М.: Вильямс, 2019. — 624 с.

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).