МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт Информационных технологий

Кафедра Математического обеспечения и стандартизации информационных технологий

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 1
по дисциплине
«Структуры и алгоритмы обработки данных»

Тема: «Поразрядные операции и их применение»

Выполнил студент группы ИКБО-28-22 Мольников М. А.

Принял преподаватель Филатов А.С.

Лабораторная работа выполнена «__»_____202__ г. *(подпись студента)*

«Зачтено» «__»_____202__ г. *(подпись руководителя)*

Москва 2023

1. Цель работы

Получение навыков применения поразрядных операций в алгоритмах.

2. Постановка задачи

1. Разработать программу, которая продемонстрирует выполнение упражнений варианта. Результаты выполнения упражнения выводить на монитор.

Требования к упражнениям:

1) Определить переменную целого типа, присвоить ей значение, используя константу в шестнадцатеричной системе счисления. Разработать функцию, которое установит заданные в задании биты исходного значения переменной в значение 1, используя соответствующую маску и поразрядную операцию.
2) Разработать функцию, которая обнуляет заданные в задании биты исходного значения целочисленной переменной, введенной пользователем, используя соответствующую маску и поразрядную операцию.
3) Разработать функцию, которая умножает значение целочисленной переменной, введенной пользователем, на множитель, используя соответствующую поразрядную операцию.
4) Разработать функцию, которая делит значение целочисленной переменной, введенной пользователем, на делитель, используя соответствующую поразрядную операцию.
5) Разработать функцию, реализующую задание, в которой используются только поразрядные операции. В выражении используется маска – переменная. Маска инициализируется единицей в младшем разряде (маска 1) или единицей в старшем разряде (маска 2). Изменяемое число и n вводится с клавиатуры.

2. Провести тестирование программы на небольших объемах данных, введенных вручную. Разработанные тесты должны покрывать все случаи входных данных (средний, лучший, худший). Результаты тестирования свести в сводные таблицы.

3. Составить отчет, отобразив в нем описание выполнения всех этапов разработки, тестирования и код всей программы со скриншотами результатов тестирования.

Вариант №15. Условие задания:

Упр. 1	Упр. 2	Упр. 3	Упр. 4	Упр. 5
С 9-ого четыре слева	17-ий, 15-ый, 1-ый	1024	1024	Обнулить n-ый бит, используя маску 1

3. Решение

Поразрядные операции (или битовые операции) представляют собой манипуляции с битами в двоичном представлении чисел. Эти операции часто используются для выполнения сложных вычислений на более низком уровне, что делает их полезными при работе с битовыми данными, оптимизации кода и решении различных задач. В языке программирования C++ поразрядные операции реализованы с помощью побитовых операторов.

Существует несколько основных видов поразрядных операций:

1. Побитовое И (AND): Эта операция выполняет побитовое "И" для двух чисел. Результат будет равен 1 только в том случае, если оба бита равны 1, в противном случае результат будет 0. В c++ используется оператор “&”.
2. Побитовое ИЛИ (OR): Эта операция выполняет побитовое "ИЛИ" для двух чисел. Результат будет равен 1, если хотя бы один из битов равен 1. В c++ используется оператор “|”.
3. Побитовое Исключающее ИЛИ (XOR): Эта операция выполняет побитовое "Исключающее ИЛИ" для двух чисел. Результат будет равен 1, если только один из битов равен 1. В c++ используется оператор “^”.

4. Побитовый сдвиг влево и вправо: Поразрядный сдвиг влево и вправо выполняется с помощью операторов << и >>. Они сдвигают биты числа влево или вправо, добавляя или удаляя нули с соответствующей стороны.
5. Побитовое отрицание (NOT): Эта операция инвертирует все биты числа, превращая 0 в 1 и наоборот. В c++ используется оператор “~”.

Для решения первой задачи нам понадобится разработать функцию, которая установит 4 бита слева от 9-го в значение 1. Для начала нам понадобится создать маску, которую мы будем применять, в двоичном виде она будет выглядеть так: 1111000000000, в шестнадцатеричном: 1E00. Далее, используем оператор “или” для изменения введенного значения, т.к. “или” используется для записи “1” в указанные разряды.

Для решения второй задачи нужно разработать функцию, которая обнуляет 17-й, 15-й и 1-й биты исходного значения. Для этого используем маску 14001_{16}. Далее, используем оператор “И” и отрицание “~” нашей маски, чтобы установить нужным битам значение 0.

Для решение третьей задачи нужно разработать функцию, которая умножает значение целочисленной переменной, введенной пользователем, на 1024, используя операцию сдвига “<<”. Операция сдвига “<<” умножает на 2^n, соответственно, чтобы умножить данное число на 1024, нам нужно применить оператор с числом 10, т.к. $2^{10}=1024$.

В четвертом задании нужно разработать функцию, которая делит значение целочисленной переменной, введенной пользователем, на 1024, используя операцию сдвига “>>”. Делаем всё также, как в третьем задании, только меняем “<<” на “>>”.

В пятом задании требуется разработать функцию, обнуляющую младший бит, используя маску 0x1 (Единица в младшем разряде). Для этого используем такой же метод, как и во втором задании, а именно инверсию маски и применение оператора “И”.

Так же, разработаем меню для удобного тестирования работы программы. Пользователь выбирает номер задания, вводит свои данные и получает результат.

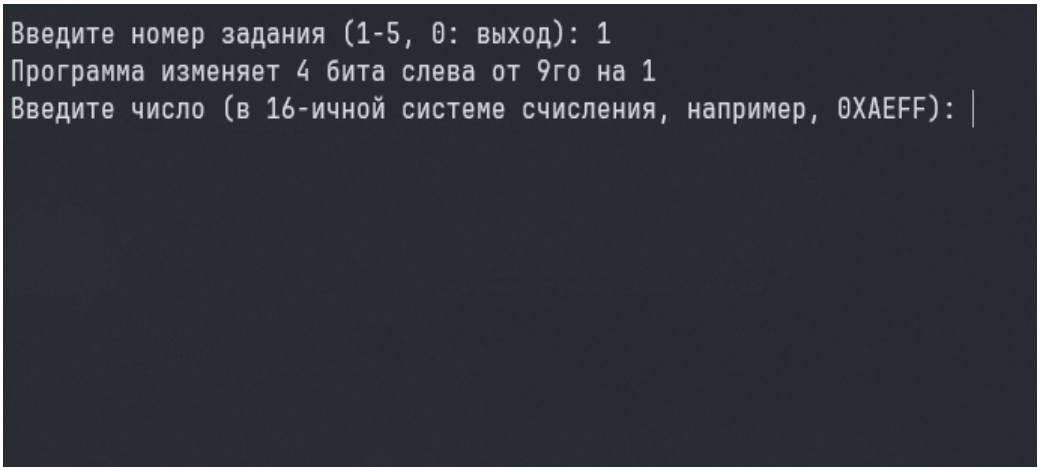

Рисунок 1. Меню программы.

4. Тестирование

Протестируем работу программы. На картинках 2-6 мы увидим выбранное задание, его описание, ввод пользователя в шестнадцатеричной и двоичной системах, а также результат работы программы в шестнадцатеричной и двоичной системах.

Первое задание: вводим число 0, 4 бита слева от 9-го изменились на единицы, всё верно.

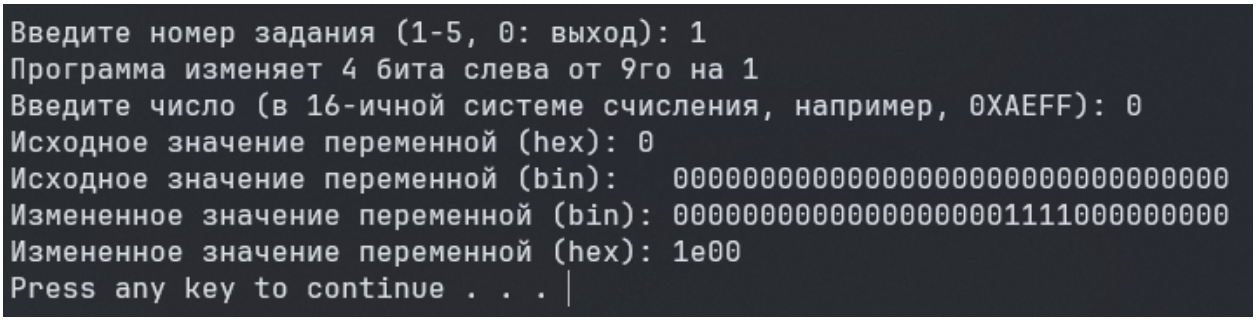

Рисунок 2. Тестирование программы

Второе задание: вводим число $FFFFF_{16}$, 1-й, 15-й и 17-й биты изменились на нули, всё верно.

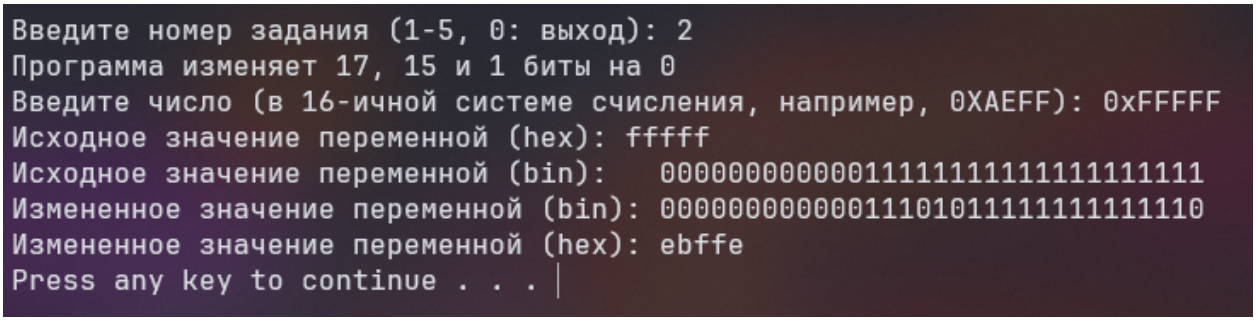

Рисунок 3. Тестирование программы

Третье задание: вводим число 5, должно получиться 5*1024=5120, всё верно.

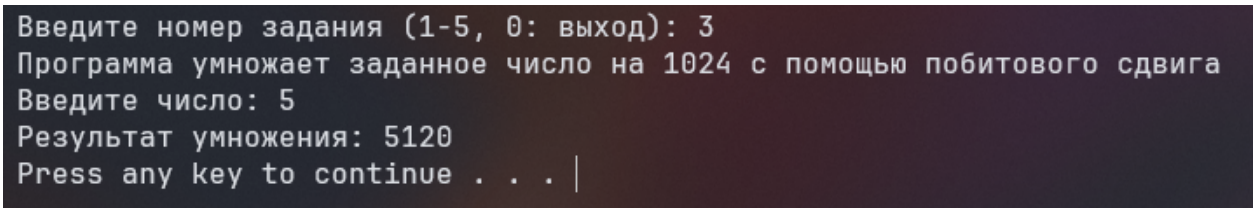

Рисунок 4. Тестирование программы

Четвертое задание: вводим число 2048, должно получиться 2048/1024=2, всё верно.

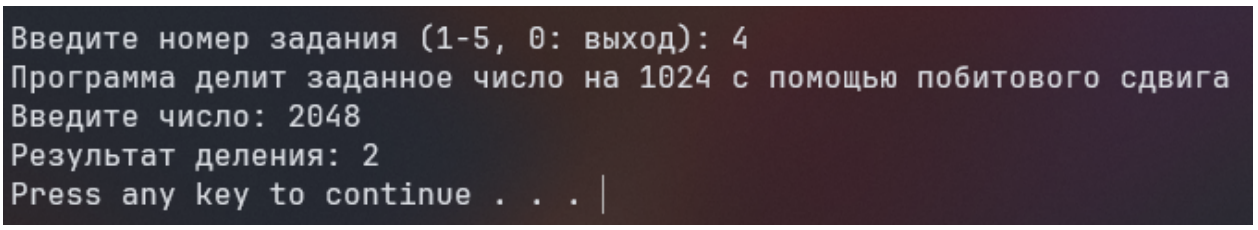

Рисунок 5. Тестирование программы

Пятое задание: вводим число $FFFF_{16}$, младший бит должен обнулиться, всё верно.

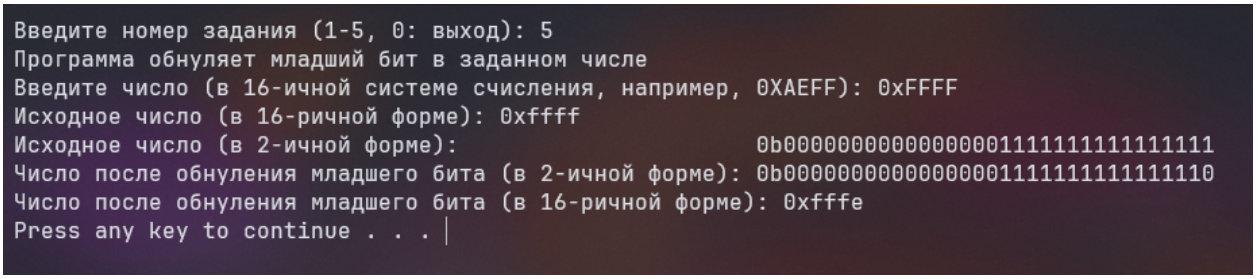

Рисунок 6. Тестирование программы

Из результатов выполнения программы видно:

1. Все задания выполнены верно.

5. Вывод

В результате выполнения работы я получил навыки по применению поразрядных операций в алгоритмах.

6. Исходный код программы

```
#include <iostream>
#include <bitset>
#include <iomanip>
#include <string>

namespace first {
  // Функция для установки битов в 1 с использованием маски
  void setBits(int& number, int mask) {
    number |= mask;
  }
}

namespace second {
  // Функция для установки битов в 0 с использованием маски
  void setBits(int& number, int mask) {
    number &= ~mask;
  }
}

namespace third {
  // Функция для умножения числа на 2 в степени multiplier
  int multiplyByShift(int num, int multiplier) {
```

```
    long int result = num << multiplier;
    return result;
  }
}

namespace fourth {
  // Функция для деления числа на 2 в степени divider
  int divideByShift(int num, int divider) {
    long int result = num >> divider;
    return result;
  }
}

namespace fifth {
  // Функция для установки битов в 0 с использованием маски
  void setBits(int& number, int mask) {
    number &= ~mask;
  }
}

int main() {
  setlocale(LC_ALL, "Russian");

  int choice;

  while (true) {
    system("cls");

    std::cout << "Введите номер задания (1-5, 0: выход): ";
    std::cin >> choice;

    switch (choice) {
      case 1: {
        std::string sValue;

        std::cout << "Программа изменяет 4 бита слева от 9го на 1" <<
std::endl;
        std::cout << "Введите число (в 16-ичной системе счисления,
например, 0XAEFF): ";
        std::cin >> sValue;

        int value = std::stoi(sValue, nullptr, 16);
        int mask = 0x1E00; // девятый и 4 слева биты

        std::cout << "Исходное значение переменной (hex): " << std::hex <<
value << std::endl;
        std::cout << "Исходное значение переменной (bin): " <<
std::bitset<sizeof(int) * 8>(value) << std::endl;
```

```
        first::setBits(value, mask);

        std::cout << "Измененное значение переменной (hex): " << std::hex
<< value << std::endl;
        std::cout << "Измененное значение переменной (bin): " <<
std::bitset<sizeof(int) * 8>(value) << std::endl;

        system("pause");
        break;
      }
      case 2: {
        std::string sValue;

        std::cout << "Программа изменяет 17, 15 и 1 биты на 0" <<
std::endl;
        std::cout << "Введите число (в 16-ичной системе счисления,
например, 0XAEFF): ";
        std::cin >> sValue;

        int value = std::stoi(sValue, nullptr, 16);

        std::cout << "Исходное значение переменной (hex): " << std::hex <<
value << std::endl;
        std::cout << "Исходное значение переменной (bin): " <<
std::bitset<sizeof(int) * 8>(value) << std::endl;

        // Маска для изменения 17го, 15го и 1го битов в шестнадцатеричной
системе счисления
        int mask = 0x14001;

        second::setBits(value, mask);

        std::cout << "Измененное значение переменной (hex): " << std::hex
<< value << std::endl;
        std::cout << "Измененное значение переменной (bin): " <<
std::bitset<sizeof(int) * 8>(value) << std::endl;

        system("pause");
        break;
      }
      case 3: {
        long int inputNum;
        long int multiplier;

        std::cout << "Программа умножает заданное число на 1024 с помощью
побитового сдвига" << std::endl;
        std::cout << "Введите число: ";
```

```
        std::cin >> inputNum;

        multiplier = 10; // 2**10

        int product = third::multiplyByShift(inputNum, multiplier);

        std::cout << "Результат умножения: " << std::dec << product <<
std::endl;

        system("pause");
        break;
      }
      case 4: {
        long int inputNum;
        int divider;

        std::cout << "Программа делит заданное число на 1024 с помощью
побитового сдвига" << std::endl;
        std::cout << "Введите число: ";
        std::cin >> inputNum;

        divider = 10; // 2**10

        int fraction = fourth::divideByShift(inputNum, divider);

        std::cout << "Результат деления: " << fraction << std::endl;

        system("pause");
        break;
      }
      case 5: {
        std::string sNum;
        int num;
        int mask;

        std::cout << "Программа обнуляет младший бит в заданном числе" <<
std::endl;
        std::cout << "Введите число (в 16-ичной системе счисления,
например, 0XAEFF): ";
        std::cin >> sNum;

        num = std::stoi(sNum, nullptr, 16);

        // Задаем маску n, равную единице в младшем бите
        mask = 0x1;

        std::cout << "Исходное число (в 16-ричной форме): 0x" << std::hex
<< num << std::endl;
```

```
        std::cout << "Исходное число (в 2-ичной форме): 0b" <<
std::bitset<sizeof(int) * 8>(num) << std::endl;

        fifth::setBits(num, mask);

        std::cout << "Число после обнуления младшего бита (в 16-ричной
форме): 0x" << std::hex << num << std::endl;
        std::cout << "Число после обнуления младшего бита (в 2-ичной
форме): 0b" << std::bitset<sizeof(int) * 8>(num) << std::endl;

        system("pause");
        break;
      }
      case 0: {
        return 0;
      }
    }
  }
  return 0;
}
```