|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | 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-10-21 | Смольников А.Б. |
| Принял преподаватель | Филатов А.С. |

|  |  |  |
| --- | --- | --- |
| Лабораторная работа выполнена | «\_\_»\_\_\_\_\_\_\_202\_\_ г. | *(подпись студента)* |
|  |  |  |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_202\_\_ г. | *(подпись руководителя)* |

Москва 2022

# **Цель работы**

Получение навыков применения поразрядных операций в алгоритмах.

# **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.
6. 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.
7. Составить отчет, отобразив в нем описание выполнения всех этапов разработки, тестирования и код всей программы со скриншотами результатов тестирования.

Вариант №3. Условие задания:

|  |  |
| --- | --- |
| Упражнение 1 | Все четные |
| Упражнение 2 | 7-ой, 9-ый, 11-ый |
| Упражнение 3 | 16 |
| Упражнение 4 | 16 |
| Упражнение 5 | Обнулить n-ый бит, используя маску 1 |

# **Решение**

В С++ каждая переменная имеет свое битовое представление (набор единиц и нулей). Все операции работают на индивидуальном битовом уровне. Поразрядный оператор применим только к символьным и целочисленным типам данных. Побитовые операции: побитовое ИЛИ («|»), побитовое И («&»), побитовое НЕ («~»), побитовое исключающее ИЛИ («^»), побитовый сдвиг влево («<<»), побитовый сдвиг вправо («>>»). Используя данный набор операций, можно взаимодействовать напрямую с битами переменных.

Функция coutp() принимает на вход целое беззнаковое число и выводит значения его битов на экран с помощью поразрядных операций.

|  |
| --- |
| //Функция вывода на экран с помощью поразрядных операций (упр. 1-5)  void coutp(unsigned int x)  {  int n=sizeof(int)\*8;  unsigned maska=1<<(n-1);  for(int i=0; i<n; i++)  {  cout<<(x&maska ? '1':'0');  x<<=1;  }  } |

Функция set\_even\_1() принимает на вход целое беззнаковое число и устанавливает все четные биты в единицу с помощью поразрядных операций.

|  |
| --- |
| //Функция установки четных битов в единицу с помощью поразрядных операций (упр. 1)  unsigned int set\_even\_1(unsigned int inp){  return inp|0b10101010101010101010101010101010;  } |

Функция set\_11\_9\_7\_to\_1() принимает на вход целое беззнаковое число и устанавливает 11, 9, 7 биты в единицу с помощью поразрядных операций.

|  |
| --- |
| //Функция установки 11, 9, 7 битов в единицу с помощью поразрядных операций (упр. 2)  unsigned int set\_11\_9\_7\_to\_1(unsigned int inp){  return inp|0b101010000000;  } |

Функция multiply\_by\_16() принимает на вход целое беззнаковое число и умножает его на 16 с помощью поразрядного сдвига.

|  |
| --- |
| //Функция умножения на 16 с помощью поразрядных операций (упр. 3)  unsigned int multiply\_by\_16(unsigned int inp){  return inp<<4;  } |

Функция multiply\_by\_16() принимает на вход целое беззнаковое число и умножает его на 16 с помощью поразрядного сдвига.

|  |
| --- |
| //Функция деления на 16 с помощью поразрядных операций (упр. 4)  unsigned int divide\_by\_16(unsigned int inp){  return inp>>4;  } |

Функция zero\_nth\_bit() принимает на вход целое беззнаковое число и устанавливает n-ый бит в 0 с помощью поразрядных операций и маски 1 (единица в младшем разряде).

|  |
| --- |
| //Функция установки n-ого бита в 0 с помощью поразрядных операций и маски 1 (упр. 5)  unsigned int zero\_nth\_bit(unsigned int inp, unsigned int n){  return inp & ~(1<<n);  } |

При запуске программы пользователю предоставляется выбор одного из пяти упражнений при помощи ввода чисел 1-5, любой другой введенный символ спровоцирует завершение программы.

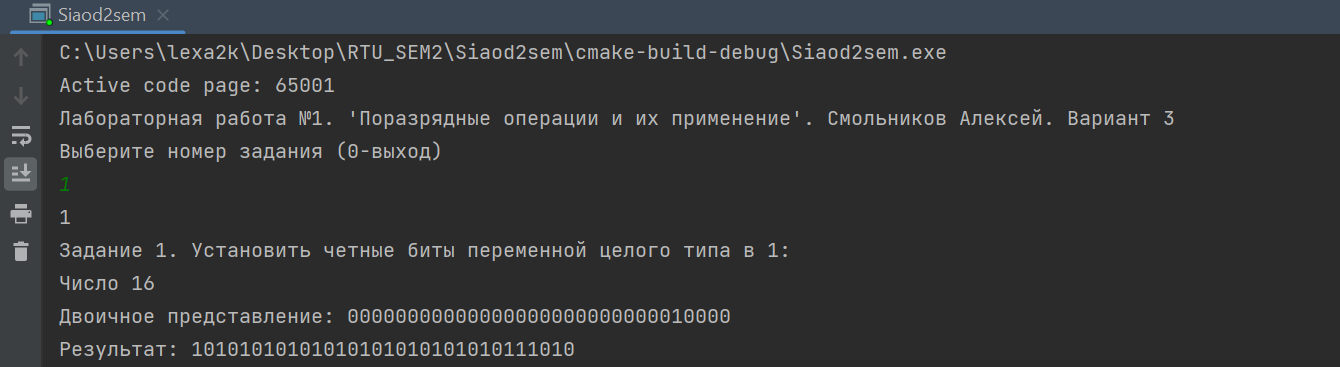


Рисунок 1. Интерфейс программы

# **Тестирование**

В силу простоты вычислений совместим тестовый прогон с рабочим, проверив каждое задание по одному разу.

Все функции получают на вход целое беззнаковое число, это сделано специально для того, чтобы не возникало изменение знака числа, например, при умножении на 16 (сдвига на 4 разряда влево). Проведем тестирование каждого элемента программы на двух пограничных наборах и одном произвольном:

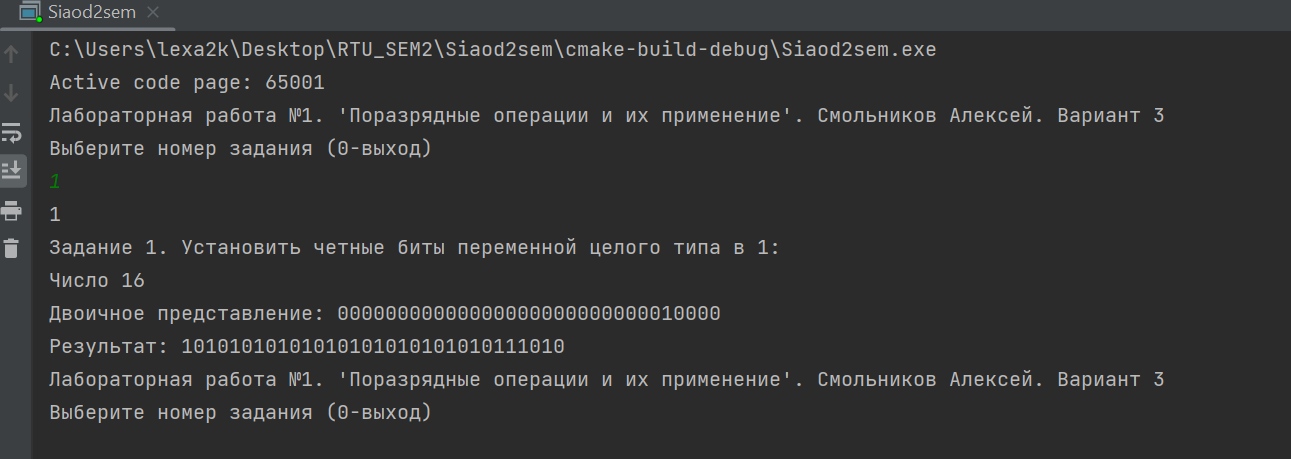


Рисунок 2. Тестирование задания 1

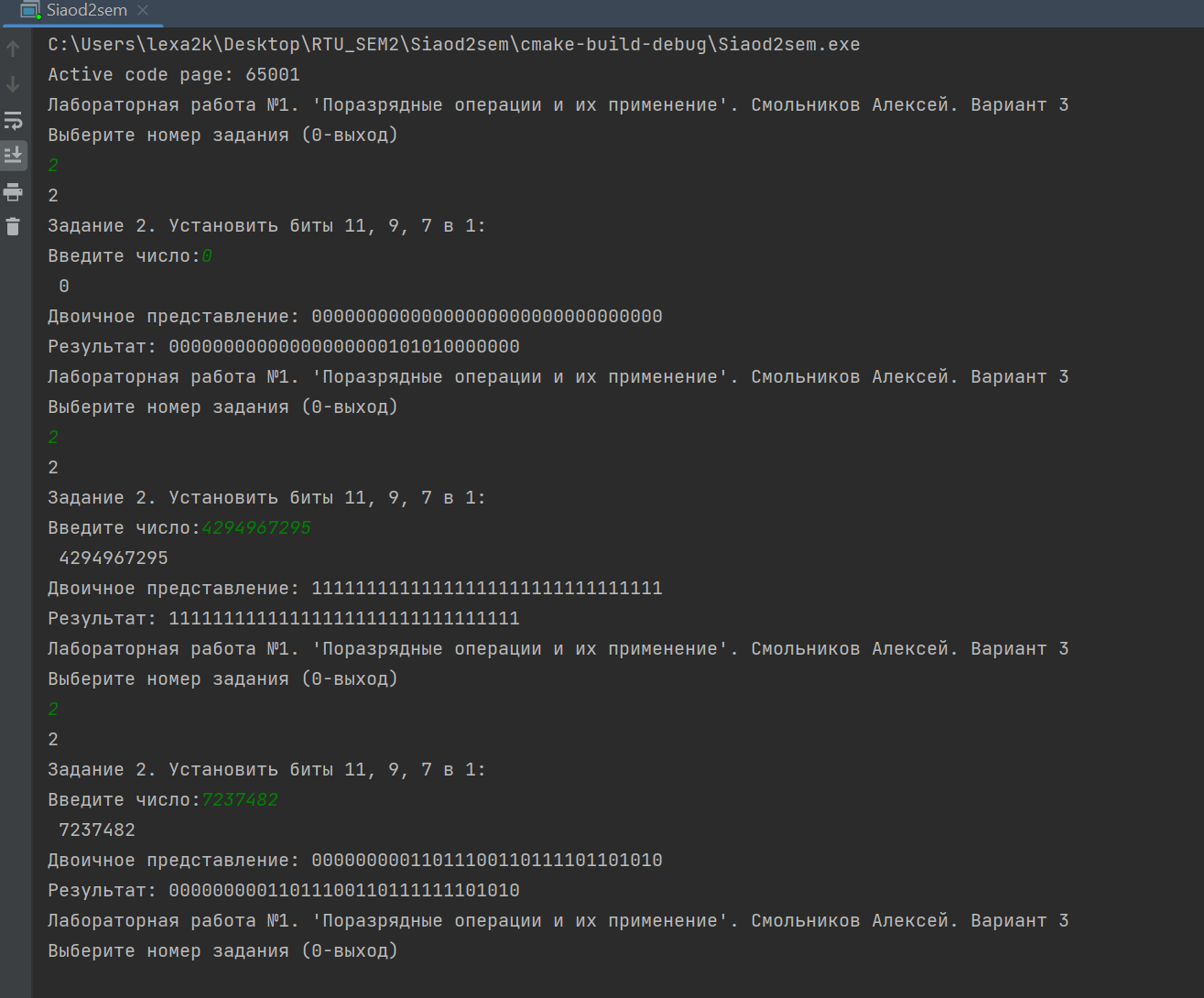


Рисунок 3. Тестирование задания 2

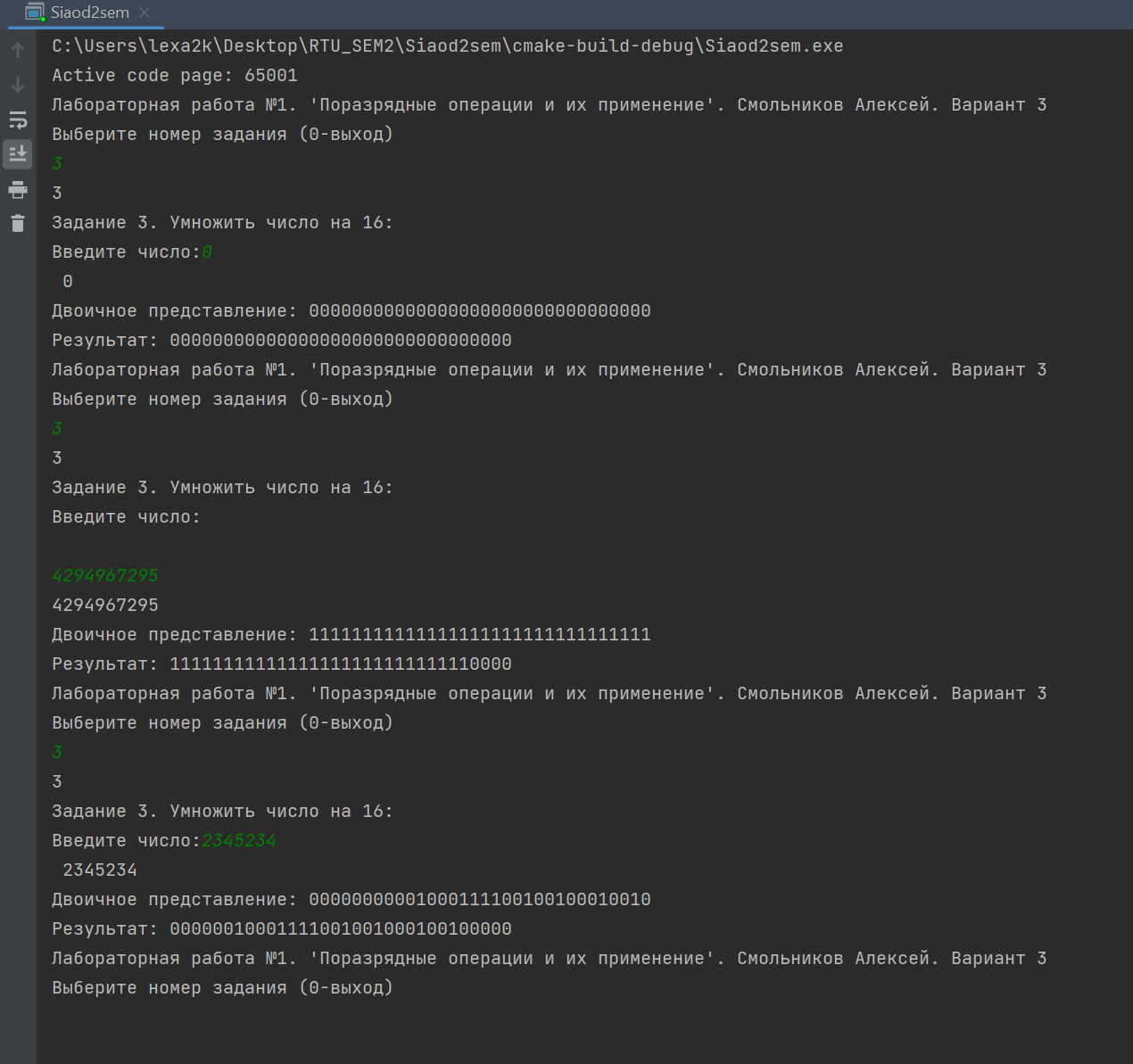


Рисунок 4. Тестирование задания 3

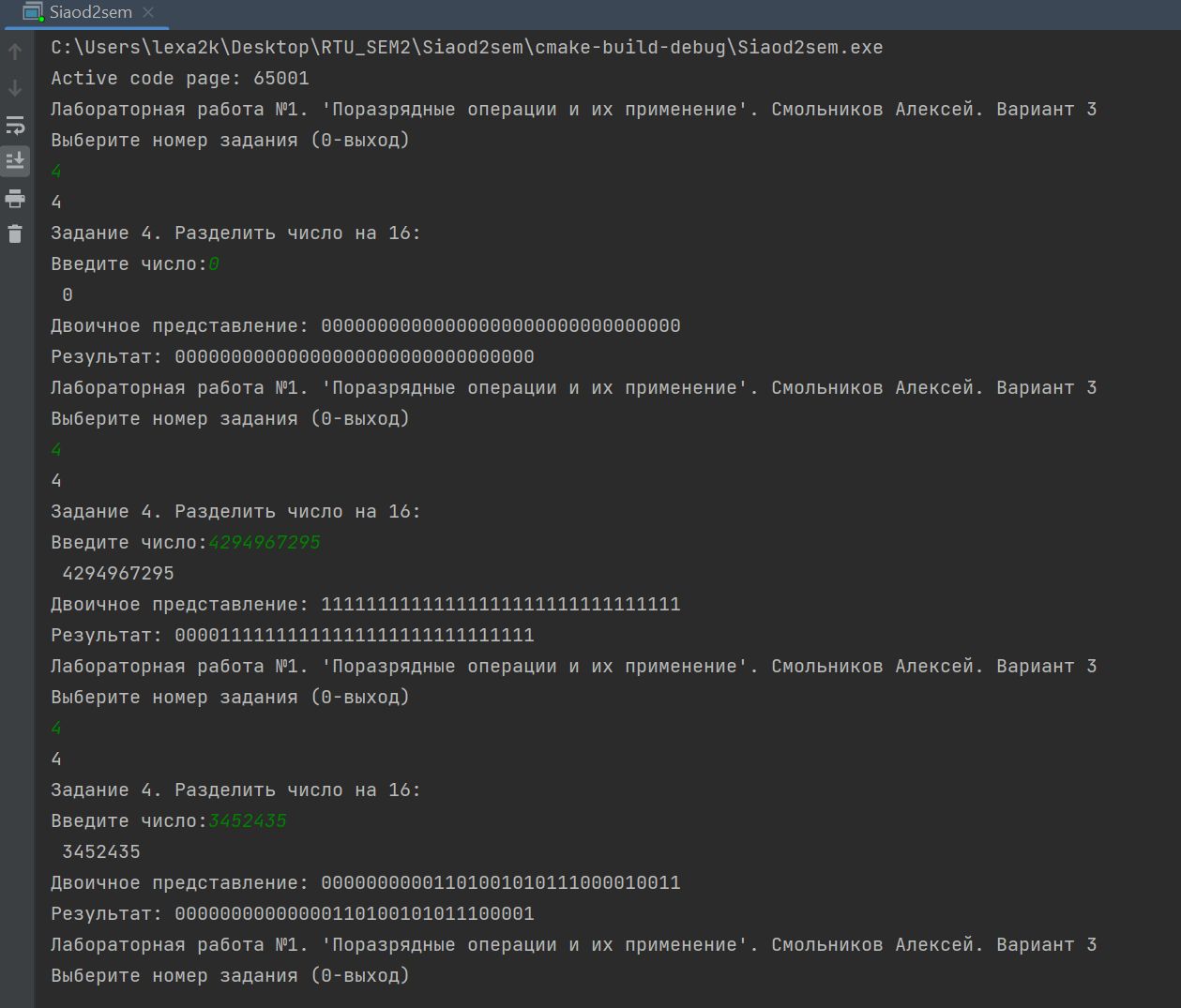
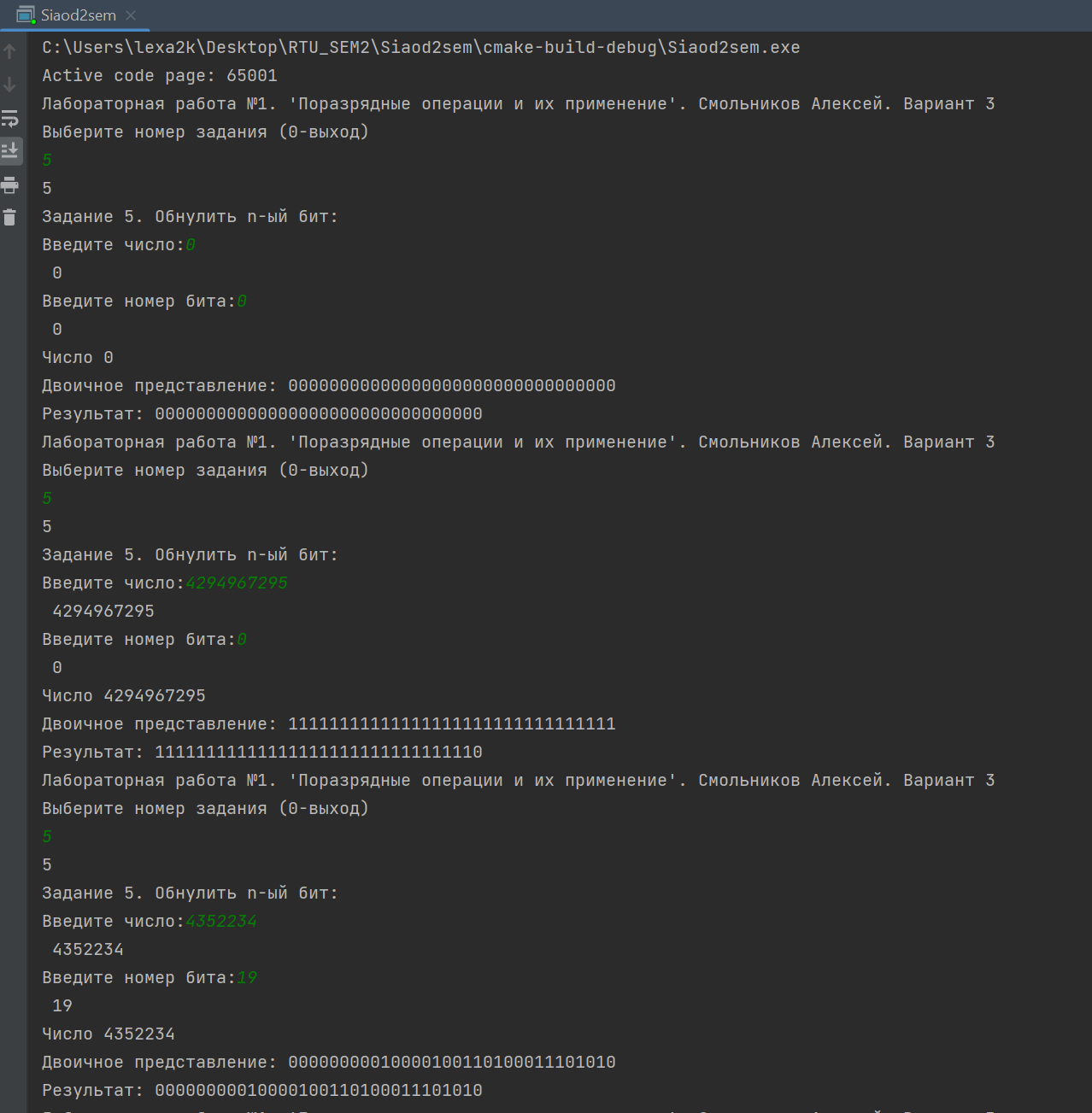


Рисунок 5. Тестирование задания 4

Рисунок 6. Тестирование задания 5

Из результатов выполнения программы видно:

1. Функция вывода двоичного значения числа работает корректно
2. Функции, выполняющие задания 1-5 работают корректно

# **Выводы**

В результате выполнения работы я:

1. Освоил алгоритмы работы с поразрядными операциями и их реализацию на языке программирования C++
2. Научился программировать автоматическое тестирование простых программ

# **Исходный код программы**

|  |
| --- |
| #include <iostream>  #include <string>  using namespace std;  //Функция вывода на экран с помощью поразрядных операций (упр. 1-5)  void coutp(unsigned int x)  {  int n=sizeof(int)\*8;  unsigned maska=1<<(n-1);  for(int i=0; i<n; i++)  {  cout<<(x&maska ? '1':'0');  x<<=1;  }  }  //Функция установки четных битов в единицу с помощью поразрядных операций (упр. 1)  unsigned int set\_even\_1(unsigned int inp){  return inp|0b10101010101010101010101010101010;  }  //Функция установки 11, 9, 7 битов в единицу с помощью поразрядных операций (упр. 2)  unsigned int set\_11\_9\_7\_to\_1(unsigned int inp){  return inp|0b101010000000;  }  //Функция умножения на 16 с помощью поразрядных операций (упр. 3)  unsigned int multiply\_by\_16(unsigned int inp){  return inp<<4;  }  //Функция деления на 16 с помощью поразрядных операций (упр. 4)  unsigned int divide\_by\_16(unsigned int inp){  return inp>>4;  }  //Функция установки n-ого бита в 0 с помощью поразрядных операций и маски 1 (упр. 5)  unsigned int zero\_nth\_bit(unsigned int inp, unsigned int n){  return inp & ~(1<<n);  }  int main() {  system("chcp 65001");  //create switch-case menu for 5 elements where 0 calls exit  int choice=1;  unsigned int inp;  while(choice!=0){  cout<<"Лабораторная работа №1. 'Поразрядные операции и их применение'. Смольников Алексей. Вариант 3"<<endl;  cout<<"Выберите номер задания (0-выход)"<<endl;  cin>>choice;  switch (choice) {  case 1:  cout<<"Задание 1. Установить четные биты переменной целого типа в 1:"<<endl;  cout<<"Число 16"<<endl;  cout<<"Двоичное представление: ";  coutp(0x10);  cout<<endl;  cout<<"Результат: ";  coutp(set\_even\_1(0x10));  cout<<endl;  break;  case 2:  cout<<"Задание 2. Установить биты 11, 9, 7 в 1:"<<endl;  cout<<"Введите число: ";  cin>>inp;  cout<<"Двоичное представление: ";  coutp(inp);  cout<<endl;  cout<<"Результат: ";  coutp(set\_11\_9\_7\_to\_1(inp));  cout<<endl;  break;  case 3:  cout<<"Задание 3. Умножить число на 16:"<<endl;  cout<<"Введите число: ";  cin>>inp;  cout<<"Двоичное представление: ";  coutp(inp);  cout<<endl;  cout<<"Результат: ";  coutp(multiply\_by\_16(inp));  cout<<endl;  break;  case 4:  cout<<"Задание 4. Разделить число на 16:"<<endl;  cout<<"Введите число: ";  cin>>inp;  cout<<"Двоичное представление: ";  coutp(inp);  cout<<endl;  cout<<"Результат: ";  coutp(divide\_by\_16(inp));  cout<<endl;  break;  case 5:  cout<<"Задание 5. Обнулить n-ый бит:"<<endl;  cout<<"Введите число: ";  cin>>inp;  cout<<"Введите номер бита: ";  unsigned int n;  cin>>n;  cout<<"Число "<<inp<<endl;  cout<<"Двоичное представление: ";  coutp(inp);  cout<<endl;  cout<<"Результат: ";  coutp(zero\_nth\_bit(inp, n));  cout<<endl;  break;  default:  break;  }  }  return 0;  } |