|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | 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-03-21 | Хречко С.В. |
| 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 №11. Условие задания:

|  |  |
| --- | --- |
| Упражнение 1 | С 5-ого четыре слева |
| Упражнение 2 | 5-ый, 7-ой справа |
| Упражнение 3 | 32 |
| Упражнение 4 | 32 |
| Упражнение 5 | Обнулить n-ый бит, используя маску 1 |

# **Решение**

Побитовые операции – это операции, работающие напрямую с битами данных.

|  |  |  |
| --- | --- | --- |
| x<<n | Сдвиг влево двоичного кода (умножение на 2n) | unsigned int x=7; x=x<<2;  результат 0х0000001С |
| x>>n | Сдвиг вправо двоичного кода (деление на 2n) | unsigned int x=28; x=x>>2;  результат =0х00000007 |
| x & maska | Поразрядное И (применяется для записи в указанный разряд 0) | Правило выполнения операции  111  & 100  = 100  Установить в двоичном коде переменной ***х*** только 9-ый справа бит в 0  unsigned short int x=0xAEFF;  unsigned short int maska=0xFDFF;  x=x & maska результат 0xACFF |
| X | maska | Поразрядное ИЛИ (применяется для записи в указанный разряд 1) | Правило выполнения операции  111  | 100  111  Установить в двоичном коде переменной х 9-ый справа бит в 1  unsigned short int x=0xACFF;  unsigned short int maska=0x0200;  x=x | maska результат 0xAEFF; |
| X ^ maska | Исключающее ИЛИ для поразрядных операций.  Используется для проверки соответствующих битов двух переменных, если они имеют разные значения, то результат 1, а если равны, то 0. | Правило выполнения операции  1111  ^ 0001  = 1110  unsigned int x=0xF, a=1;  a=x^a;  Результат: в переменной ***а*** значение  0х0000000Е |
| ~ | Инверсия (0 заменяет на 1, а 1 на 0) | x=0x0F;  ~x;  результат 0хF0 |

Каждое задание настолько маленькое и тривиальное, что было принято решение выделять функции на каждое отдельное задание и его выполнение от выбора конкретного задания пользователем, до его завершения. В таком случае (а данной конкретной работе) код получается проще для восприятия.

Первое упражнение: по условию данные задавались мной, потому они заданы в коде. Функция применяет требуемую побитовую операцию.

|  |
| --- |
| void firstTask()  {      cout << "first task" << endl;      unsigned int x = 0xa7;      unsigned int mask = 0b111100000;      coutp(x);      coutp(x | mask);  } |

Второе упражнение: Для обнуления требуемых битов, нужно сделать конъюнкцию с маской, в которой на требуемых позициях стоят нули, а на остальных единицы, для этого используется отрицание, чтобы не писать большое число единиц в начале числа.

|  |
| --- |
| void secondTask()  {      cout << "second task" << endl;      unsigned int x;      cout << "input x" << endl;      cin >> x;      unsigned int mask = 0b10100000;      mask = ~mask;      coutp(x);      coutp(x & mask);  } |

Третье упражнение: Для умножения на 32 нужно совершить сдвиг влево на 5 битов.

|  |
| --- |
| void thirdTask()  {      cout << "third task" << endl;      unsigned int x;      cout << "input x" << endl;      cin >> x;      cout << (x << 5);  } |

Четвертое упражнение: Для деления на 32 нужно совершить сдвиг вправо на 5 битов.

|  |
| --- |
| void fourthTask()  {      cout << "fourth task" << endl;      unsigned int x;      cout << "input x" << endl;      cin >> x;      cout << (x >> 5);  } |

Пятое упражнение: выполняется по аналогии со вторым, но нужно создать маску из единицы сдвинув влево на n.

|  |
| --- |
| void fifthTask()  {      cout << "fifth task" << endl;      unsigned int x;      cout << "input x" << endl;      cin >> x;      unsigned int mask = 1;      coutp(x);      cout << "input n" << endl;      int n;      cin >> n;        mask = mask << n;      coutp(x & (~mask));  } |

Функция coutp выводит на экран переданное число в битовом виде, количество выводимых битов определяется n.

|  |
| --- |
| void coutp(unsigned int x)  {      int n=sizeof(int)\*8;      unsigned maska=(1<<(n-1));      for(int i=1; i<=n;i++)      {          cout<<((x&maska)>>(n-i));          maska=maska>>1;      }      cout << endl;  } |

Пользовательский интерфейс предлагает выбрать какое упражнение выполнять (рис. 1).

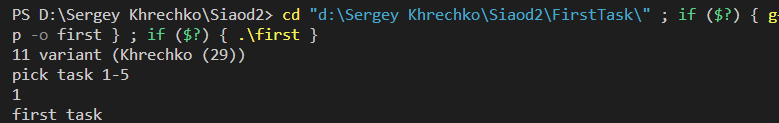


Рисунок 1. Интерфейс программы

# **Тестирование**

Тестирование проводится вручную в связи с простотой упражнений. Результаты тестирования приведены на рисунках далее (рис. 2-6).

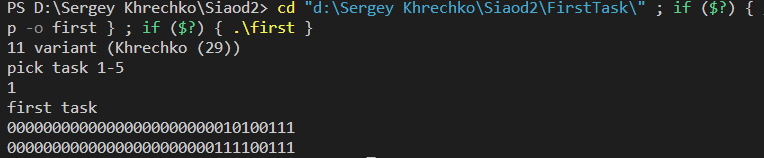


Рисунок 2. Тестирование упражнения 1

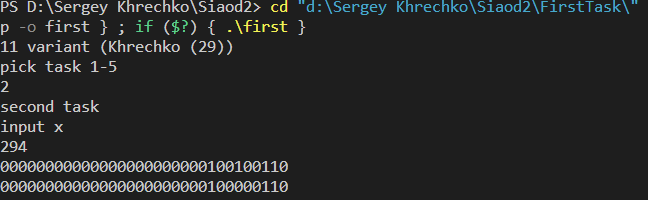


Рисунок 3. Тестирование упражнения 2

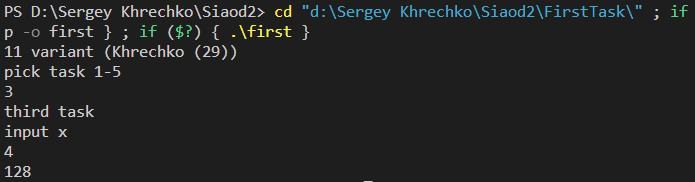


Рисунок 4. Тестирование упражнения 3

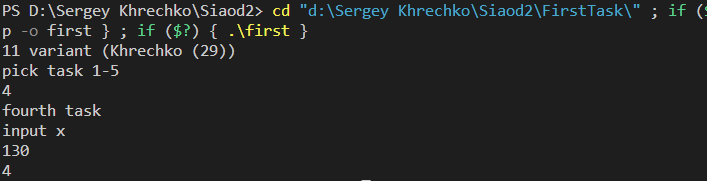


Рисунок 5. Тестирование упражнения 4

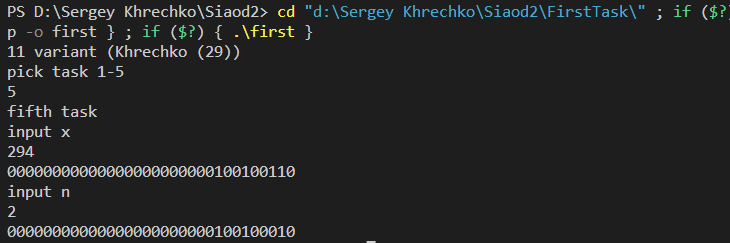


Рисунок 6. Тестирование упражнения 5

Из результатов выполнения программы видно:

1. Программа работает правильно и выполнены все упражнения.

# **Вывод**

В результате выполнения работы я:

1. Освоил алгоритмы работы с поразрядными операциями и их реализацию на языке программирования C++

# **Исходный код программы**

|  |
| --- |
| #include <iostream>  using namespace std;  void coutp(unsigned int x);  void firstTask();  void secondTask();  void thirdTask();  void fourthTask();  void fifthTask();  int main()  {      cout << "11 variant (Khrechko (29))" << endl;      cout << "pick task 1-5" << endl;      //I don't do protection against wront input      int pick;      cin >> pick;      switch (pick)      {      case 1:          firstTask();          break;      case 2:          secondTask();          break;      case 3:          thirdTask();          break;      case 4:          fourthTask();          break;      case 5:          fifthTask();          break;      default:          cout << "wrong input" << endl;          break;      }      return 0;  }  void firstTask()  {      cout << "first task" << endl;      unsigned int x = 0xa7;      unsigned int mask = 0b111100000;      coutp(x);      coutp(x | mask);  }  void secondTask()  {      cout << "second task" << endl;      unsigned int x;      cout << "input x" << endl;      cin >> x;      unsigned int mask = 0b10100000;      mask = ~mask;      coutp(x);      coutp(x & mask);  }  void thirdTask()  {      cout << "third task" << endl;      unsigned int x;      cout << "input x" << endl;      cin >> x;      cout << (x << 5);  }  void fourthTask()  {      cout << "fourth task" << endl;      unsigned int x;      cout << "input x" << endl;      cin >> x;      cout << (x >> 5);  }  void fifthTask()  {      cout << "fifth task" << endl;      unsigned int x;      cout << "input x" << endl;      cin >> x;      unsigned int mask = 1;      coutp(x);      cout << "input n" << endl;      int n;      cin >> n;        mask = mask << n;      coutp(x & (~mask));  }  void coutp(unsigned int x)  {      int n=sizeof(int)\*8;      unsigned maska=(1<<(n-1));      for(int i=1; i<=n;i++)      {          cout<<((x&maska)>>(n-i));          maska=maska>>1;      }      cout << endl;  } |