**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Объектно-ориентированное программирование»

**Лабораторная работа № 1**

Тема: Простые классы на языке С++

Студент: Баранников Степан

Группа: 80-201

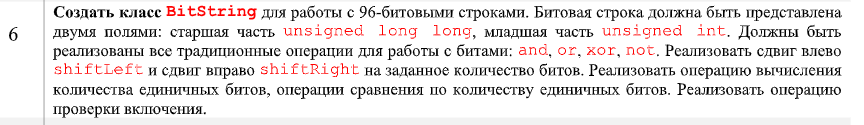
Преподаватель: Чернышов Л.Н.

Дата: 22.09.2019

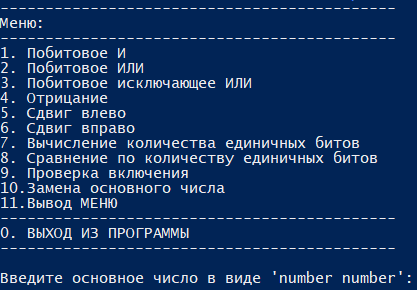
Оценка:

Москва, 2019

1. Постановка задачи

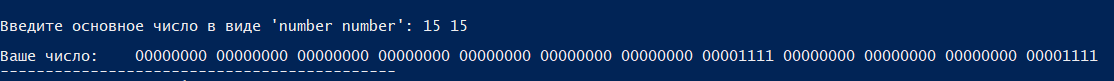


1. Описание программы

В программе реализован класс BitString, владеющий всеми вышеперечисленными логическими операциями. При запуске программы создается три объекта типа BitString, которые с помощью конструктора получают значение битовой строки ноль. Далее программа просит пользователя задать основную битовую строку в виде “Dec Dec”, где Dec - десятичное число, и вызывает функцию setData($value$), которая устанавливает значение данной битовой строки, с которой далее можно будет проводить различные операции с помощью интерактивного меню. Также помимо основного задания реализован вывод битовой строки в машинном виде в двоичной системе счисления.

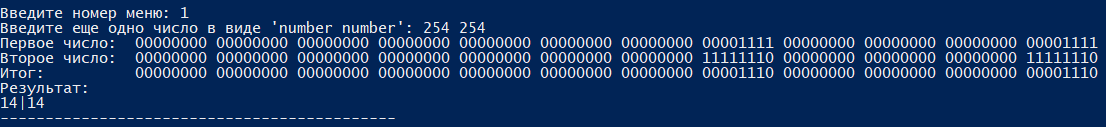
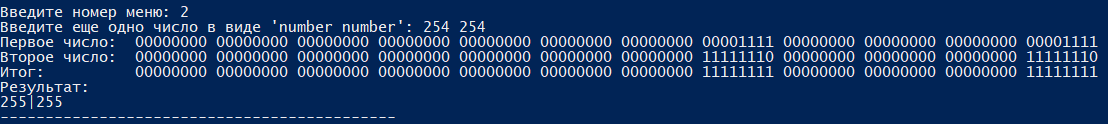
Ссылка на github: <https://github.com/mamongo/oop_exercize_01>

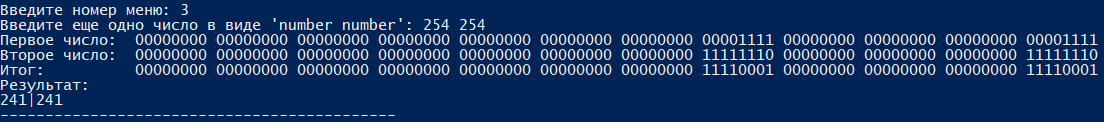
1. Набор testcases

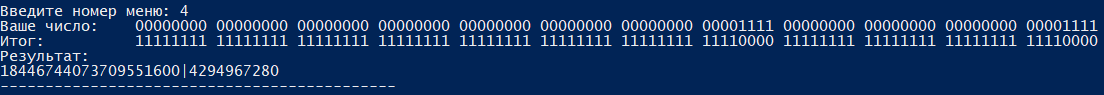
Вводится основное число 15 15

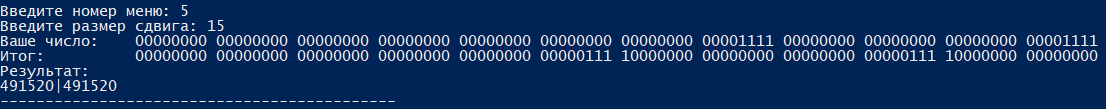
Далее работаем с меню:

1) Побитовое И: вводится еще одно число 254 254

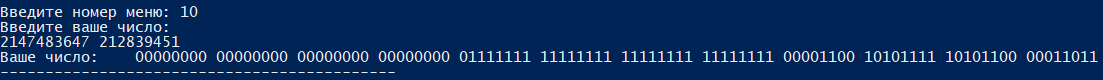
Далее 2) Побитовое ИЛИ: вводится число 254 254

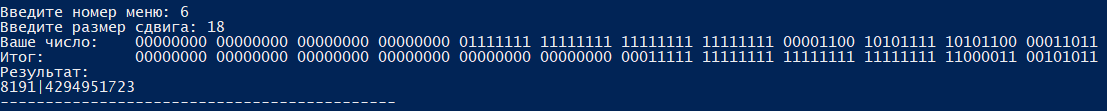
Далее 3) Побитовое исключающее ИЛИ: вводится число 254 254

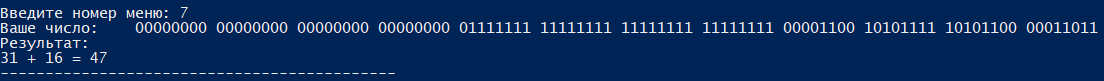
Далее 4) Отрицание: ничего не вводится, идет работа с основным числом

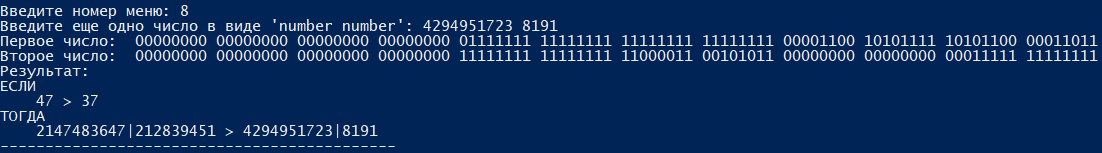
Далее 5) Побитовый сдвиг влево: вводится размер сдвига

Далее для наглядности сменим основное число:

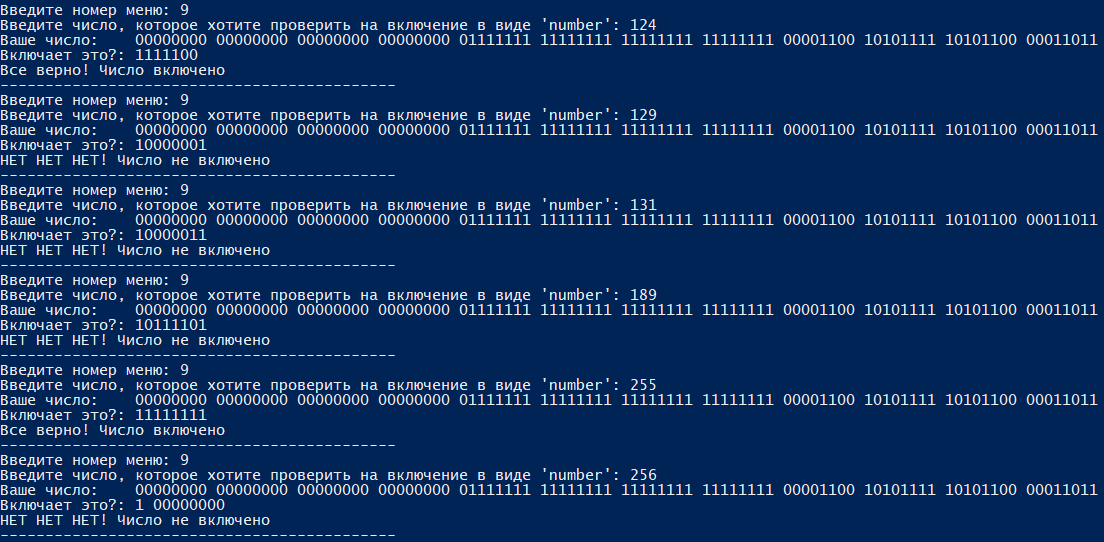
10) вводится число 2147483647 212839451

Далее 6) Побитовый сдвиг вправо: вводится размер сдвига 18

Далее 7) Вычисления количества единичных битов: ничего не вводится, идет работа с основным числом

Далее 8) Сравнение по количеству единичных битов: вводится число 4294951723 8191

Далее 9) Проверка включения: вводится число, которое нужно проверить на включение в основное. Для лучшей проверки вводятся по очереди числа 124, 129, 131, 189, 255, 256



1. Результаты выполнения тестов. (для наглядности совмещено с 3 пунктом)
2. Листинг программы

|  |
| --- |
| //Barannikov Stepan M8o-201b  #include <iostream>  using namespace std;  struct am{  int old, young, sum; }; struct bs{  unsigned long long old;  unsigned int young; };  class BitString{ private:  bs data;  int\* bindata;  public:  BitString(bs a = {0,0}){  setData(a);  }   bs getData(){  return data;  }  int\* getBindata(){  return bindata;  }   void setData(bs a){  data.young = a.young;  data.old = a.old;  bindata = BSToArr(data);  }  void setData(BitString &a){  setData(a.getData());  }   am BitsAmount(bs a){  bs buf = a;  am count;  count.young = count.old = 0;  for (int i=0; i<32; i++){  if ((buf.young >> i) % 2 == 1)  count.young++;  }  for (int i=0; i<64; i++){  if ((buf.old >> i) % 2 == 1)  count.old++;  }  count.sum = count.old + count.young;  return count;  }   int\* BSToArr(bs a){  int i,j;  int \*asave = new int[96];  int \*young = new int[32];  int \*old = new int[64];  for(i=0; i<32; i++){  young[i] = (a.young >> i) % 2;  }  //PrintBS(young, 32);  for(j=0; j<64; j++){  old[j] = (a.old >> j) % 2;  }  //PrintBS(old, 64);  for(i=0; i<32; i++){  asave[i] = young[i];  }  j = 0;  for(i=32; i<96; i++){  asave[i] = old[j];  j++;  }  return asave;  }   void PrintBS(){  int \*a = bindata;  int l = 96-1;  for(int i=l; i>=0; i--){  cout << a[i];  if(i%8==0)  cout << " ";  }  cout << endl;  }  void PrintBS(int \*a, int l = 96){  for(int i=l-1; i>=0; i--){  cout << a[i];  if(i%8==0)  cout << " ";  }  cout << endl;  }   BitString& And(BitString &b){  BitString \*sol = new (BitString);  sol->data.old = data.old & b.data.old;  sol->data.young = data.young & b.data.young;  return \*sol;  }  BitString& Or(BitString &b){  BitString \*sol = new (BitString);  sol->data.old = data.old | b.data.old;  sol->data.young = data.young | b.data.young;  return \*sol;  }  BitString& Xor(BitString &b){  BitString \*sol = new (BitString);  sol->data.old = data.old ^ b.data.old;  sol->data.young = data.young ^ b.data.young;  return \*sol;  }  bs Not(){  bs sol;  sol.old = ~data.old;  sol.young = ~data.young;  return sol;  }   bs ShiftLeft(int n){  bs sol;  if (n<32){  bs sol;  sol.old = data.old << n;  sol.old = sol.old | (data.young >> (32-n));  sol.young = data.young << n;   }  else{  sol.old = data.old << 32;  sol.old = sol.old | data.young;  sol.old = sol.old << (n-32);  sol.young = 0;  }  return sol;  }  bs ShiftRight(int n){  bs sol;  if(n < 32){  sol.young = data.young >> n;  //unsigned long long trans = (data.old << (64-n)) >> 32;  sol.young = sol.young | ((data.old << (64-n)) >> 32);  sol.old = data.old >> n;  }  else{  sol.old = data.old >> (n-32);  sol.young = 0;  sol.young = sol.young | sol.old;  sol.old = data.old >> 32;  }  return sol;  }   am BitsAmount(){  bs buf = data;  am count;  count.young = count.old = 0;  for (int i=0; i<32; i++){  if ((buf.young >> i) % 2 == 1)  count.young++;  }  for (int i=0; i<64; i++){  if ((buf.old >> i) % 2 == 1)  count.old++;  }  count.sum = count.old + count.young;  return count;  }  bool Includes(unsigned long long b){  am aam = BitsAmount(data);  bs bconv;  bconv.old = b, bconv.young = 0;  am bam = BitsAmount(bconv);  if(aam.sum < bam.sum){  return false;  }  else{  int i,j,k;  int \*asave;  int \*bsave;  bool check;  asave = BSToArr(data);  unsigned long long bbuf = b;  int n = 0;  while(bbuf != 0){  bbuf = bbuf / 2;  n++;  }  if(n != 0)  bsave = new int[n];  else{  return false;  }  for(j=0; j<n; j++){  bsave[j] = (b >> j) % 2;  }  PrintBS(bsave, n);  for(i=0; i<96; i++){  if(asave[i] == bsave[j]){  k = i;  for(j=1; j<n; j++){  if(asave[k] == bsave[j]){  check = true;  k++;  }  else{  check = false;  break;  }  }  if(check)  break;  }  else{  check = false;  }  }  return check;  }  }  };  int main(){  BitString a,b,c;  bs ad, bd, cd;  am aam, bam;  int it;  int k = -1, n;  bool check;  cout <<  "--------------------------------------------\n" <<  "Меню:\n" <<  "--------------------------------------------\n" <<  "1. Побитовое И\n" <<  "2. Побитовое ИЛИ\n" <<  "3. Побитовое исключающее ИЛИ\n" <<  "4. Отрицание\n" <<  "5. Сдвиг влево\n" <<  "6. Сдвиг вправо\n" <<  "7. Вычисление количества единичных битов\n" <<  "8. Сравнение по количеству единичных битов\n" <<  "9. Проверка включения\n" <<  "10.Замена основного числа\n" <<  "11.Вывод МЕНЮ\n" <<  "--------------------------------------------\n" <<  "0. ВЫХОД ИЗ ПРОГРАММЫ\n" <<  "--------------------------------------------\n";  cout << "\nВведите основное число в виде 'number number': ";  cin >> ad.old >> ad.young;  a.setData(ad);  cout << "\nВаше число: ";  a.PrintBS();  cout << "--------------------------------------------\n";  while(k!=0){  cout << "Введите номер меню: ";  cin >> k;  switch(k){  case(1):  cout << "Введите еще одно число в виде 'number number': ";  cin >> bd.old >> bd.young;  b.setData(bd);  cout << "Первое число: ";  a.PrintBS();  cout << "Второе число: ";  b.PrintBS();  c.setData(a.And(b));  cout << "Итог: ";  c.PrintBS();  cout << "Результат:" << endl << c.getData().old << "|" << c.getData().young;  cout << "\n--------------------------------------------\n";  break;  case(2):  cout << "Введите еще одно число в виде 'number number': ";  cin >> bd.old >> bd.young;  b.setData(bd);  cout << "Первое число: ";  a.PrintBS();  cout << "Второе число: ";  b.PrintBS();  c.setData(a.Or(b));  cout << "Итог: ";  c.PrintBS();  cout << "Результат:" << endl << c.getData().old << "|" << c.getData().young;  cout << "\n--------------------------------------------\n";  break;  case(3):  cout << "Введите еще одно число в виде 'number number': ";  cin >> bd.old >> bd.young;  b.setData(bd);  cout << "Первое число: ";  a.PrintBS();  cout << "Второе число: ";  b.PrintBS();  c.setData(a.Xor(b));  cout << "Итог: ";  c.PrintBS();  cout << "Результат:" << endl << c.getData().old << "|" << c.getData().young;  cout << "\n--------------------------------------------\n";  break;  case(4):  cout << "Ваше число: ";  a.PrintBS();  c.setData(a.Not());  cout << "Итог: ";  c.PrintBS();  cout << "Результат:" << endl << c.getData().old << "|" << c.getData().young;  cout << "\n--------------------------------------------\n";  break;  case(5):  cout << "Введите размер сдвига: ";  cin >> n;  cout << "Ваше число: ";  a.PrintBS();  c.setData(a.ShiftLeft(n));  cout << "Итог: ";  c.PrintBS();  cout << "Результат:" << endl << c.getData().old << "|" << c.getData().young;  cout << "\n--------------------------------------------\n";  break;  case(6):  cout << "Введите размер сдвига: ";  cin >> n;  c.setData(a.ShiftRight(n));  cout << "Ваше число: ";  a.PrintBS();  cout << "Итог: ";  c.PrintBS();  cout << "Результат:" << endl << c.getData().old << "|" << c.getData().young;  cout << "\n--------------------------------------------\n";  break;  case(7):  cout << "Ваше число: ";  a.PrintBS();  aam = a.BitsAmount();  cout << "Результат:" << endl << aam.old << " + " << aam.young << " = " << aam.sum;  cout << "\n--------------------------------------------\n";  break;  case(8):  cout << "Введите еще одно число в виде 'number number': ";  cin >> bd.old >> bd.young;  b.setData(bd);  cout << "Первое число: ";  a.PrintBS();  cout << "Второе число: ";  b.PrintBS();  aam = a.BitsAmount();  bam = b.BitsAmount();  if(aam.sum < bam.sum){  cout << "Результат:" << endl << "ЕСЛИ" << endl << " " << aam.sum << " < " << bam.sum << endl;  cout << "ТОГДА" << endl << " " << a.getData().old << "|" << a.getData().young << " < " << b.getData().old << "|" << b.getData().young;  }  else if(aam.sum > bam.sum){  cout << "Результат:" << endl << "ЕСЛИ" << endl << " " << aam.sum << " > " << bam.sum << endl;  cout << "ТОГДА" << endl << " " << a.getData().old << "|" << a.getData().young << " > " << b.getData().old << "|" << b.getData().young;  }  else{  cout << "Результат:" << endl << "ЕСЛИ" << endl << " " << aam.sum << " = " << bam.sum << endl;  cout << "ТОГДА" << endl << " " << a.getData().old << "|" << a.getData().young << " = " << b.getData().old << "|" << b.getData().young;  }  cout << "\n--------------------------------------------\n";  break;  case(9):  cout << "Введите число, которое хотите проверить на включение в виде 'number': ";  cin >> it;  cout << "Ваше число: ";  a.PrintBS();  cout << "Включает это?: ";  check = a.Includes(it);  if(check)  cout << "Все верно! Число включено" << endl;  else  cout << "НЕТ НЕТ НЕТ! Чиcло не включено" << endl;  cout << "--------------------------------------------\n";  break;  case(10):  cout << "Введите ваше число:\n";  cin >> ad.old >> ad.young;  a.setData(ad);  cout << "Ваше число: ";  a.PrintBS();  cout << "--------------------------------------------\n";  break;  case(11):  cout <<  "--------------------------------------------\n" <<  "Меню:\n" <<  "--------------------------------------------\n" <<  "1. Побитовое И\n" <<  "2. Побитовое ИЛИ\n" <<  "3. Побитовое исключающее ИЛИ\n" <<  "4. Отрицание\n" <<  "5. Сдвиг влево\n" <<  "6. Сдвиг вправо\n" <<  "7. Вычисление количества единичных битов\n" <<  "8. Сравнение по количеству единичных битов\n" <<  "9. Проверка включения\n" <<  "10.Замена основного числа\n" <<  "11.Вывод МЕНЮ\n" <<  "--------------------------------------------\n" <<  "0. ВЫХОД ИЗ ПРОГРАММЫ\n" <<  "--------------------------------------------\n";  break;  }  }  return 0; } |

1. Вывод:

Я научился создавать простейшие классы на языке C++, а также решать задачи с использованием классов.

## Список литературы

1. Введение в ООП  
   <https://ravesli.com/urok-112-vvedenie-v-oop/>
2. Классы. Объекты и методы  
   <https://ravesli.com/urok-113-klassy-obekty-i-metody-klassov/>
3. Спецификаторы доступа public и private  
   <https://ravesli.com/urok-114-spetsifikatory-dostupa-public-i-private/>
4. Инкапсуляция  
   <https://ravesli.com/urok-115-inkapsulyatsiya-gettery-i-settery/>
5. Конструкторы

<https://ravesli.com/urok-116-konstruktory/#toc-0>