**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Объектно-ориентированное программирование»

**Лабораторная работа № 2**

Тема: Перегрузка операторов в С++

Студент: Баранников Степан Алексеевич

Группа: 80-201

Преподаватель: Чернышов Л.Н.

Дата: 25.09.2019

Оценка:

Москва, 2019

1. Постановка задачи

|  |  |
| --- | --- |
| 18 | **Создать класс IPAddress** для работы с адресом в интернет. Класс состоит из четырех чисел unsigned char (a,b,c,d). Реализовать арифметические операции сложения, вычитания, а также операции сравнения (для сравнение на больше/меньше считать что левые байты главнее т.е. вначале сравниваются первые байты, потом вторые и т.д.). Так же реализовать функцию, которая будет определять принадлежность адреса к подсети по адресу подсети (a1,b1,c1,d1) и битовой маске подсети (a2,b2,c2,d2). Например, адрес 192.168.1.30 принадлежит подсети 192.168.0.0 с маской 255.255.0.0.  Операции сложения, вычитания и сравнения необходимо реализовать в виде перегрузки операторов.  Необходимо реализовать пользовательский литерал для работы с константами типа IPAddress. |

1. Описание программы

В программе реализован класс IPAddress, имеющий все вышеперечисленные методы работы с данным классом данных. Операции сложения, вычитания, а также сравнения двух ip адресов реализованы с перегрузкой операторов +, -, >, <, ==. Так же перегружен оператор потокового вывода <<, чтобы объекты данного класса можно было без труда выводить на экран и, помимо этого, перегружен оператор присваивания =, чтобы было можно более свободно и комфортно оперировать объектами данного класса.  
 пример:   
 IPaddress a,b,c;  
 a = b + c; a = b - c; a = b&c; cout << a; bool check = a < b;  
Так же реализован пользовательский литерал “” \_ip, который преобразует данные в структуру union ipv4, которая обладает переменными двух типов: 4 элемента в типе bytes типа unsigned char, т.е bytes.a, bytes.b, bytes.c, bytes.d и элемент типа unsigned int размером 32 бит, чтобы хранить ip в более удобном для операций формате. Union ipv4 же, в свою очередь, самостоятельно преобразует один формат данных в другой.

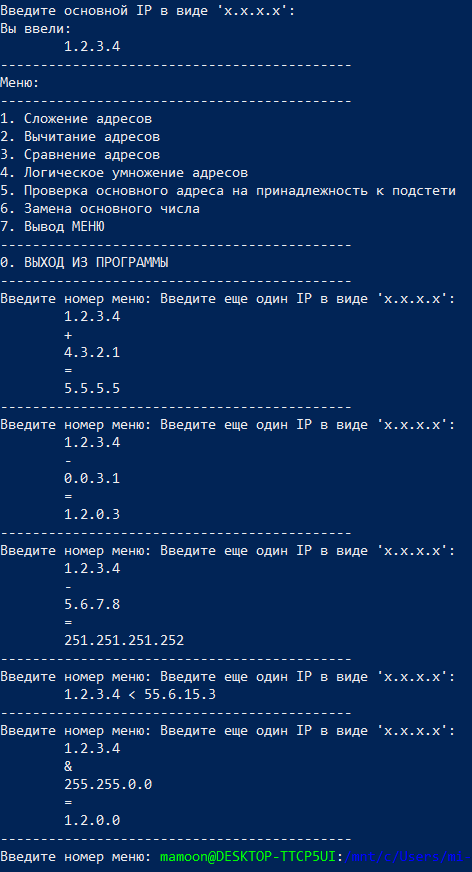
пример:  
ipv4 ip="192.168.1.30"\_ip;  
a = ip; //Объекту “a” типа IPAddress было присвоено поле значений типа ipv4

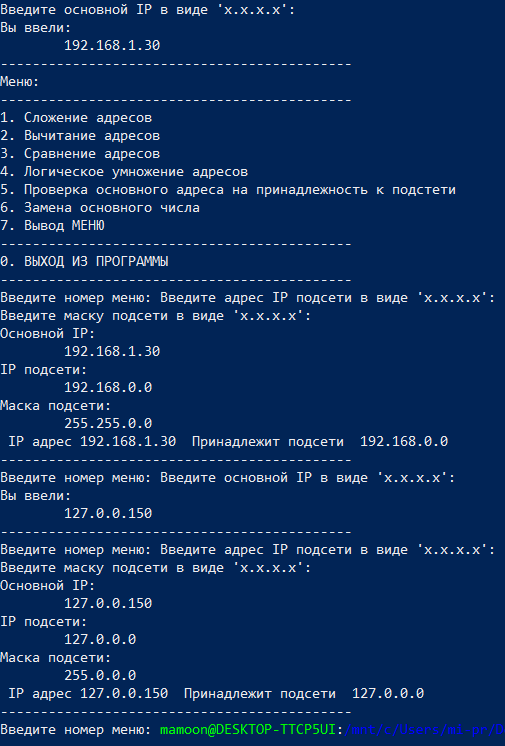
1. Набор testcases

test\_01.txt  
1.2.3.4  
1  
4.3.2.1  
2  
0.0.3.1  
2  
5.6.7.8  
3  
55.6.15.3  
4  
255.255.0.0  
0

test\_02.txt  
192.168.1.30  
5  
192.168.0.0  
255.255.0.0  
6  
127.0.0.150  
5  
127.0.0.0  
255.0.0.0  
0

1. Результаты выполнения тестов.

./exercize\_02 < test\_01.txt  


./exercize\_02 < test\_02.txt  


1. Листинг программы

|  |
| --- |
| //Barannikov Stepan M8o-201b /\* \* ======================== \* Создать класс IPAddress для работы с адресом в интернет. \* Класс состоит из четырех чисел unsigned char (a,b,c,d). \* Реализовать арифметические операции сложения, вычитания, \* а также операции сравнения (для сравнение на больше/меньше считать, \* что левые байты главнее т.е. вначале сравниваются первые байты, потом вторые и т.д.). \* Так же реализовать функцию, которая будет определять принадлежность адреса к подсети \* по адресу подсети (a1,b1,c1,d1) и битовой маске подсети (a2,b2,c2,d2). \* Например, адрес 192.168.1.30 принадлежит подсети 192.168.0.0 с маской 255.255.0.0. \*  \* Операции сложения, вычитания и сравнения необходимо реализовать в виде перегрузки операторов. \*  \* Необходимо реализовать пользовательский литерал для работы с константами типа IPAddress. \* ======================== \*/ //Использование пользовательского литерала находится в конце программы (закоментировано)  #include <iostream> using namespace std; #include <sys/socket.h> #include <netinet/in.h> #include <arpa/inet.h>  struct ipv4bytes {  unsigned char d,c,b,a; }; union ipv4 {  ipv4bytes bytes;  unsigned int number; }; ipv4 operator "" \_ip(const char\* saddr, size\_t size){  ipv4 result, reverse;  in\_addr num\_ip;  inet\_aton(saddr, &num\_ip);  reverse.number = num\_ip.s\_addr;  result.bytes.a = reverse.bytes.d;  result.bytes.b = reverse.bytes.c;  result.bytes.c = reverse.bytes.b;  result.bytes.d = reverse.bytes.a;  return result; }  class IPAddress{ private:  ipv4 data; public:  IPAddress(){  data.bytes = {0,0,0,0};  }  IPAddress(ipv4 x){  setData(x);  }  ipv4 getData(){  return data;  }  void setData(ipv4 x){  data = x;  }  void setData(IPAddress &x){  setData(x.getData());  }   IPAddress& operator= (const IPAddress &x){  setData(x.data);  return \*this;  }  IPAddress& operator& (const IPAddress &y){  return And(y);  }  IPAddress& operator- (const IPAddress &y){  return minus(y);  }  IPAddress& operator+ (const IPAddress &y){  return plus(y);  }  bool operator< (IPAddress &y){  bool l = less(data, y.data);  return l;  }  bool operator> (IPAddress &y){  bool m = more(data, y.data);  return m;  }  bool operator== (IPAddress &y){  bool eq = equal(data, y.data);  return eq;  }  friend ostream& operator<< (ostream& stream, const IPAddress &x){  in\_addr numeric\_ip;  ipv4 save;  save.bytes.a = x.data.bytes.d;  save.bytes.b = x.data.bytes.c;  save.bytes.c = x.data.bytes.b;  save.bytes.d = x.data.bytes.a;  numeric\_ip.s\_addr = save.number;  stream << inet\_ntoa(numeric\_ip);  return stream;  }   IPAddress &plus(const IPAddress &y){  IPAddress \*sol = new(IPAddress);  sol->data.number = data.number + y.data.number;  return \*sol;  }  IPAddress &minus(const IPAddress &y){  IPAddress \*sol = new(IPAddress);  sol->data.number = data.number - y.data.number;  return \*sol;  }  IPAddress &And(const IPAddress &y){  IPAddress \*sol = new(IPAddress);  sol->data.number = data.number & y.data.number;  return \*sol;  }   //sravneniye  bool less(ipv4 x, ipv4 y){  return (x.number < y.number);  }  bool more(ipv4 x, ipv4 y){  return (x.number > y.number);  }  bool equal(ipv4 x, ipv4 y){  return (x.number == y.number);  }  bool includes(IPAddress &netw,const IPAddress &mask){  IPAddress x = data;  if((x & mask) == (netw & mask))  return true;  else  return false;  } };   int main() {  IPAddress x,y,mask;  ipv4 ip1, ip2, ip3;  int a,b,c,d;  int k = -1;  cout << "Введите основной IP в виде 'x.x.x.x':\n";  scanf("%d.%d.%d.%d", &a, &b, &c, &d);  ip1 = {(unsigned char)d,(unsigned char)c,(unsigned char)b,(unsigned char)a};  x = ip1;  cout << "Вы ввели:\n\t" << x << endl;  cout <<  "--------------------------------------------\n" <<  "Меню:\n" <<  "--------------------------------------------\n" <<  "1. Сложение адресов\n" <<  "2. Вычитание адресов\n" <<  "3. Сравнение адресов\n" <<  "4. Логическое умножение адресов\n" <<  "5. Проверка основного адреса на принадлежность к подстети\n" <<  "6. Замена основного числа\n" <<  "7. Вывод МЕНЮ\n" <<  "--------------------------------------------\n" <<  "0. ВЫХОД ИЗ ПРОГРАММЫ\n" <<  "--------------------------------------------\n";  while(k!=0){  cout << "Введите номер меню: ";  cin >> k;  switch(k){  case(1):  cout << "Введите еще один IP в виде 'x.x.x.x':\n";  scanf("%d.%d.%d.%d", &a, &b, &c, &d);  ip2 = {(unsigned char)d,(unsigned char)c,(unsigned char)b,(unsigned char)a};  y = ip2;  cout << "\t" << x << "\n\t+\n\t" << y << "\n\t=\n\t" << (x+y) << endl;  cout << "--------------------------------------------\n";  break;  case(2):  cout << "Введите еще один IP в виде 'x.x.x.x':\n";  scanf("%d.%d.%d.%d", &a, &b, &c, &d);  ip2 = {(unsigned char)d,(unsigned char)c,(unsigned char)b,(unsigned char)a};  y = ip2;  cout << "\t" << x << "\n\t-\n\t" << y << "\n\t=\n\t" << (x-y) << endl;  cout << "--------------------------------------------\n";  break;  case(3):  cout << "Введите еще один IP в виде 'x.x.x.x':\n";   scanf("%d.%d.%d.%d", &a, &b, &c, &d);  ip2 = {(unsigned char)d,(unsigned char)c,(unsigned char)b,(unsigned char)a};  y = ip2;  if(x<y)  cout << "\t" << x << " < " << y << endl;  else if(x>y)  cout << "\t" << x << " > " << y << endl;  else if(x==y)  cout << "\t" << x << " = " << y << endl;  cout << "--------------------------------------------\n";  break;  case(4):  cout << "Введите еще один IP в виде 'x.x.x.x':\n";  scanf("%d.%d.%d.%d", &a, &b, &c, &d);  ip2 = {(unsigned char)d,(unsigned char)c,(unsigned char)b,(unsigned char)a};  y = ip2;  cout << "\t" << x << "\n\t&\n\t" << y << "\n\t=\n\t" << (x&y) << endl;  cout << "--------------------------------------------\n";  break;  case(5):  cout << "Введите адрес IP подсети в виде 'x.x.x.x':\n";  scanf("%d.%d.%d.%d", &a, &b, &c, &d);  ip2 = {(unsigned char)d,(unsigned char)c,(unsigned char)b,(unsigned char)a};  y = ip2;  cout << "Введите маску подсети в виде 'x.x.x.x':\n";  scanf("%d.%d.%d.%d", &a, &b, &c, &d);  ip3 = {(unsigned char)d,(unsigned char)c,(unsigned char)b,(unsigned char)a};  mask = ip3;  cout << "Основной IP:\n\t" << x << "\nIP подсети:\n\t" << y << "\nМаска подсети:\n\t" << mask << endl;  if(!((y&mask) == y)){  cout << " Неправильные адрес подсети и маска!\n";  cout << "--------------------------------------------\n";  break;  }  else{  bool check = x.includes(y,mask);  if(check)  cout << " IP адрес " << x << " Принадлежит подсети " << y << endl;  else  cout << " IP адрес " << x << " НЕ принадлежит подсети " << y << endl;  cout << "--------------------------------------------\n";  break;  }  case(6):  cout << "Введите основной IP в виде 'x.x.x.x':\n";  scanf("%d.%d.%d.%d", &a, &b, &c, &d);  ip1 = {(unsigned char)d,(unsigned char)c,(unsigned char)b,(unsigned char)a};  x = ip1;  cout << "Вы ввели:\n\t" << x << endl;  cout << "--------------------------------------------\n";  break;  case(7):  cout <<  "--------------------------------------------\n" <<  "Меню:\n" <<  "--------------------------------------------\n" <<  "1. Сложение адресов\n" <<  "2. Вычитание адресов\n" <<  "3. Сравнение адресов\n" <<  "4. Логическое умножение адресов\n" <<  "5. Проверка основного адреса на принадлежность к подстети\n" <<  "6. Замена основного числа\n" <<  "7. Вывод МЕНЮ\n" <<  "--------------------------------------------\n" <<  "0. ВЫХОД ИЗ ПРОГРАММЫ\n" <<  "--------------------------------------------\n";  break;  }  }  /\*  ipv4 ip="192.168.1.30"\_ip;  cout << "Hello," << ip.number << " !\n";  cout << "Hi," << (int)ip1.bytes.a << "." << (int)ip1.bytes.b << "." << (int)ip1.bytes.c << "." << (int)ip1.bytes.d << endl; \*/   return 0; } |

1. Вывод:

Научился создавать пользовательские литералы, а также научился перегружать стандартные арифметические операторы и оператор стандартного ввода <<.

## Список литературы

1. Перегрузка операторов в C++ [Электронный ресурс]  
   URL: <https://habr.com/ru/post/132014/>
2. Пользовательские литералы в C++11 [Электронный ресурс]  
   URL: <https://habr.com/ru/post/140357/>
3. Библиотека для преобразования IP в виде строки в число и обратно [Электронный ресурс]  
   URL: <https://www.opennet.ru/man.shtml?topic=inet_aton&category=3&russian=0#lbAC>