**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: Базаргармаев Нима

Группа: 80-201

Преподаватель: Чернышов Л.Н.

Дата: 04.09.2019.

Оценка:

Москва, 2019

1. Постановка задачи:

Вариант 18.

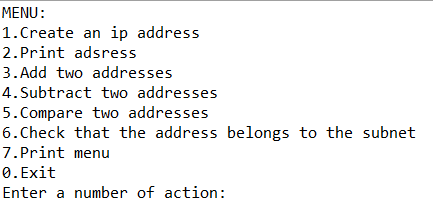
Создать класс IPAddress для работы с адресом в интернет. Класс состоит из четырех чисел unsigned char (a,b,c,d). Реализовать арифметические операции сложения, вычитания, а также операции сравнения

(для сравнение на больше/меньше считать что левые байты главнее т.е. в начале сравниваются первые байты, потом вторые и т.д.). Также 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.

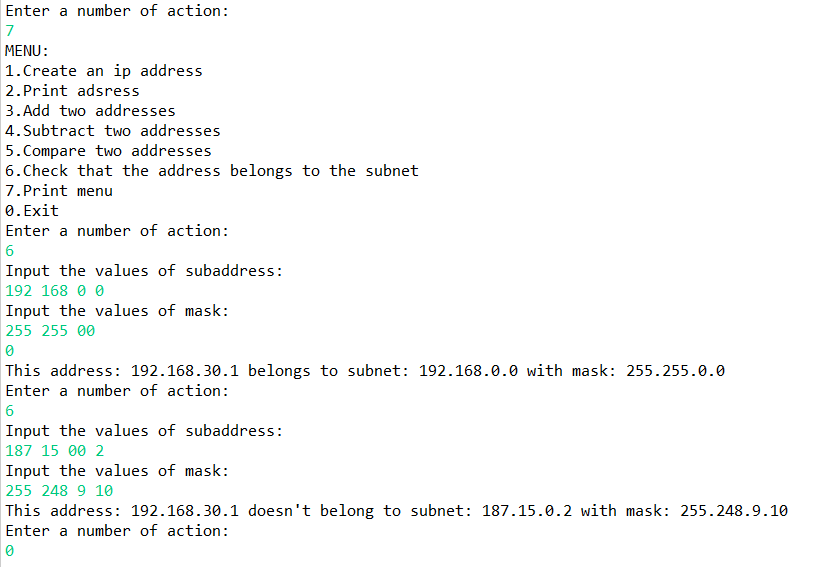
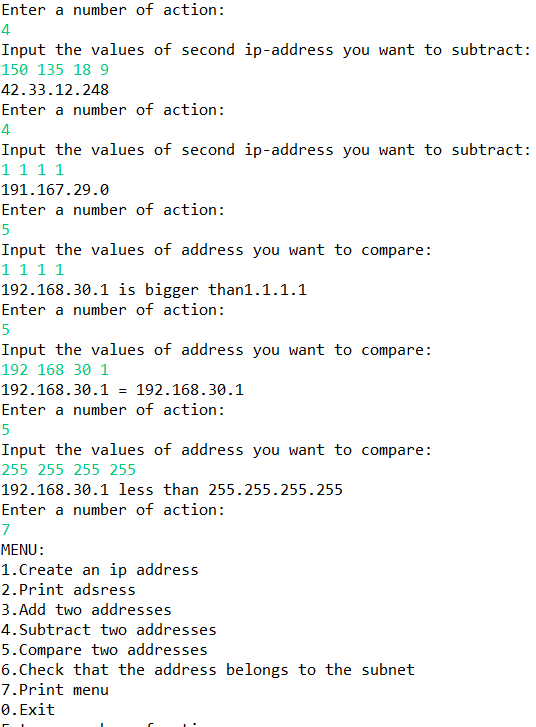
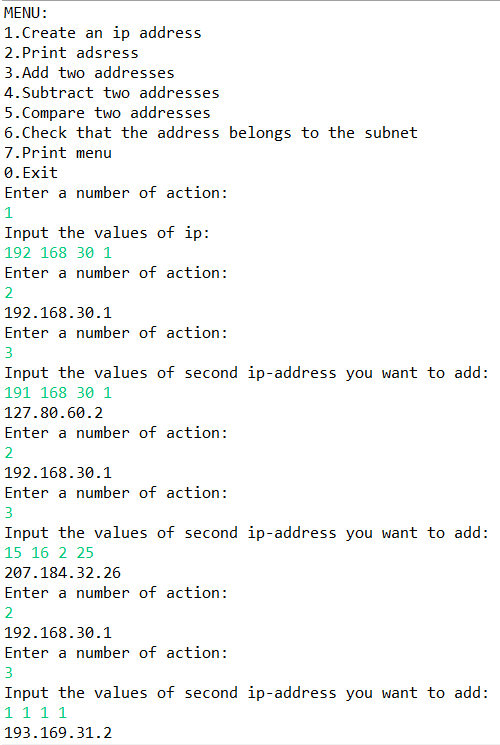
1. Описание программы:

В программе реализован класс IPAddress, владеющий всеми вышеперечисленными логическими операциями. При запуске программы создается два объекта типа IPAddress, которые с помощью конструктора по умолчанию заполняются нулями. Далее программа просит ввести пользователя число в диапазоне от 0 до 7. И в зависимости от введенного числа будут производиться операции, требуемые в задании. В программе перегружено большинство арифметических операций и оператор вывода в поток.

Меню:



1. Набор testcases



1. Результаты выполнения тестов.

Результаты выполнения тестов совмещены с набором тестов, поэтому см. выше.

1. Листинг программы

#include <iostream>

using namespace std;

class IPAddress {

public:

// конструктор по умолчанию создает адрес 000.000.0.0

IPAddress() {

a = '0';

b = '0';

c = '0';

d = '0';

}

// конструктор, задающий значения

IPAddress(unsigned char a1, unsigned char b1, unsigned char c1, unsigned char d1) {

a = a1;

b = b1;

c = c1;

d = d1;

}

void CreateAddress(const int& a1, const int& b1, const int& c1, const int& d1) {

a = a1;

b = b1;

c = c1;

d = d1;

}

// печать адреса в формате XXX.YYY.ZZ.KK

friend ostream& operator<<(ostream& stream, const IPAddress& ip);

// арифметические операции с адресами и операции сравнения

friend IPAddress operator+(IPAddress& lhs, IPAddress& rhs);

friend IPAddress operator-(IPAddress& lhs, IPAddress& rhs);

friend bool operator<(const IPAddress& lhs, const IPAddress& rhs);

friend bool operator==(const IPAddress& lhs, const IPAddress& rhs);

// проверка принадлежности к подсети по адресу и маске

friend void CheckAddress(const IPAddress& ip);

private:

unsigned char a, b, c, d;

};

ostream& operator<<(ostream& stream, const IPAddress& ip) {

stream << static\_cast<int>(ip.a) << "."

<< static\_cast<int>(ip.b) << "."

<< static\_cast<int>(ip.c) << "."

<< static\_cast<int>(ip.d);

return stream;

}

IPAddress operator+(IPAddress& lhs, IPAddress& rhs) {

IPAddress result;

result.a = static\_cast<int>(lhs.a) + static\_cast<int>(rhs.a);

result.b = static\_cast<int>(lhs.b) + static\_cast<int>(rhs.b);

result.c = static\_cast<int>(lhs.c) + static\_cast<int>(rhs.c);

result.d = static\_cast<int>(lhs.d) + static\_cast<int>(rhs.d);

return result;

}

IPAddress operator-(IPAddress& lhs, IPAddress& rhs) {

IPAddress result;

result.a = static\_cast<int>(lhs.a) - static\_cast<int>(rhs.a);

result.b = static\_cast<int>(lhs.b) - static\_cast<int>(rhs.b);

result.c = static\_cast<int>(lhs.c) - static\_cast<int>(rhs.c);

result.d = static\_cast<int>(lhs.d) - static\_cast<int>(rhs.d);

return result;

}

bool operator<(const IPAddress& lhs, const IPAddress& rhs) {

if (lhs.a < rhs.a) {

return true;

} else if ((lhs.a == rhs.a) && (lhs.b < rhs.b)) {

return true;

} else if ((lhs.a == rhs.a) && (lhs.b == rhs.b) && (lhs.c < rhs.c)) {

return true;

} else if ((lhs.a == rhs.a) && (lhs.b == rhs.b) && (lhs.c == rhs.c) && (lhs.d < rhs.d)) {

return true;

}

return false;

}

bool operator==(const IPAddress& lhs, const IPAddress& rhs) {

if (lhs.a == rhs.a && lhs.b == rhs.b && lhs.c == rhs.c && lhs.d == rhs.d) {

return true;

}

return false;

}

// функция, определяющая принадлежность адреса к подсети и его маску

void CheckAddress(const IPAddress& ip) {

int a, b, c, d;

cout << "Input the values of subaddress: " << endl;

cin >> a >> b >> c >> d;

IPAddress subaddress(a, b, c, d); // подсеть адреса

int a1, b1, c1, d1;

cout << "Input the values of mask: " << endl;

cin >> a1 >> b1 >> c1 >> d1;

IPAddress mask(a1, b1, c1, d1); // маска подсети

if (ip.a == a && ip.b == b) {

cout << "This address: " << ip << " belongs to subnet: " << subaddress

<< " with mask: " << mask << endl;

} else {

cout << "This address: " << ip << " doesn't belong to subnet: " << subaddress

<< " with mask: " << mask << endl;

}

}

int main() {

IPAddress ip, tmp; // основной ip-адрес и временный

int option = 7;

bool flag = true;

while(flag) {

switch (option) {

case 0:

flag = false;

break;

case 1: {

int a, b, c, d;

cout << "Input the values of ip: " << endl;

cin >> a >> b >> c >> d;

ip.CreateAddress(a, b, c, d);

break;

}

case 2:

cout << ip << endl;

break;

case 3: {

cout << "Input the values of second ip-address you want to add: " << endl;

int a, b, c, d;

cin >> a >> b >> c >> d;

tmp.CreateAddress(a, b, c, d);

cout << ip + tmp << endl;

break;

}

case 4: {

cout << "Input the values of second ip-address you want to subtract: " << endl;

int a, b, c, d;

cin >> a >> b >> c >> d;

tmp.CreateAddress(a, b, c, d);

cout << ip - tmp << endl;

break;

}

case 5: {

cout << "Input the values of address you want to compare: " << endl;

int a, b, c, d;

cin >> a >> b >> c >> d;

tmp.CreateAddress(a, b, c, d);

if (ip < tmp) {

cout << ip << " less than " << tmp << endl;

} else if (ip == tmp) {

cout << ip << " = " << tmp << endl;

} else {

cout << ip << " is bigger than" << tmp << endl;

}

break;

}

case 6:

CheckAddress(ip);

break;

case 7:

cout << "MENU: " << endl

<< "1.Create an ip address" << endl

<< "2.Print adsress" << endl

<< "3.Add two addresses" << endl

<< "4.Subtract two addresses" << endl

<< "5.Compare two addresses" << endl

<< "6.Check that the address belongs to the subnet" << endl

<< "7.Print menu" << endl

<< "0.Exit" << endl;

break;

}

if (flag) {

cout << "Enter a number of action: " << endl;

cin >> option;

}

}

return 0;

}

**Вывод:**

Научился создавать простейшие классы в С++ и работать с ними.

## Список литературы

1. Перегрузка операторов в C++ [Электронный ресурс]  
   URL: <https://habr.com/ru/post/132014/>
2. 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>