**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: Шавандрин Федор Михайлович

Группа: 80-208

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2020

1. Постановка задачи

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
2. Класс IPAdress

a) Конструкторы

б) Геттеры и сеттеры

в) Функция печати адреса

г) Свойства класса

1. а) Перегрузка операторов сложения, вычитания, побитовой конъюнкции

б) Перегрузка операторов сравнения

в) Функция принадлежности адреса к подсети по адресу подсети и маске

г) Функция суммы и разности двух адресов

д) Функция ввода адреса

1. Главная функция

а) Вывод меню

б) Выбор пункта меню

в) Ввод данных

г) Подсчёт результатов с помощью функций.

д) Вывод результата

1. Набор тестов

|  |  |
| --- | --- |
| Первый адрес | Второй адрес |
| 192.168.0.1 | 168.158.14.26 |
| 192.54.26.0 | 192.54.4.8 |

Адрес подсети 192.168.0.0

Маска 255.255.0.0.

1. Результаты выполнения тестов

MENU:

1. Enter the IP addresses.

2. Print first IP address.

3. Print second IP address.

4. Print a sum of IP addresses.

5. Print a difference of IP addresses.

6. Check if IP address belongs to the subnet with the address of subnet and mask.

Choose a menu item -> 1

Enter first IP address -> 192 168 0 1

Enter second IP address -> 168 158 14 26

MENU:

1. Enter the IP addresses.

2. Print first IP address.

3. Print second IP address.

4. Print a sum of IP addresses.

5. Print a difference of IP addresses.

6. Check if IP address belongs to the subnet with the address of subnet and mask.

Choose a menu item -> 4

IP address is : 255.255.14.27

MENU:

1. Enter the IP addresses.

2. Print first IP address.

3. Print second IP address.

4. Print a sum of IP addresses.

5. Print a difference of IP addresses.

6. Check if IP address belongs to the subnet with the address of subnet and mask.

Choose a menu item -> 5

IP address is : 24.10.0.0

MENU:

1. Enter the IP addresses.

2. Print first IP address.

3. Print second IP address.

4. Print a sum of IP addresses.

5. Print a difference of IP addresses.

6. Check if IP address belongs to the subnet with the address of subnet and mask.

Choose a menu item -> 6

Enter the address of subnet -> 192 168 0 0

Enter the mask -> 255 255 0 0

1. Check if first IP address belongs to the subnet with adress of subnet and mask.

2. Check if second IP address belongs to the subnet with adress of subnet and mask.

Type here -> 1

First address belongs to subnet.

MENU:

1. Enter the IP addresses.

2. Print first IP address.

3. Print second IP address.

4. Print a sum of IP addresses.

5. Print a difference of IP addresses.

6. Check if IP address belongs to the subnet with the address of subnet and mask.

Choose a menu item -> 6

The address of subnet and mask are entered.

1. Check if first IP address belongs to the subnet with adress of subnet and mask.

2. Check if second IP address belongs to the subnet with adress of subnet and mask.

Type here -> 2

Second address doesn't belong to subnet.

-----------------------------------------------------------------------------------------

MENU:

1. Enter the IP addresses.

2. Print first IP address.

3. Print second IP address.

4. Print a sum of IP addresses.

5. Print a difference of IP addresses.

6. Check if IP address belongs to the subnet with the address of subnet and mask.

Choose a menu item -> 1

Enter first IP address -> 192 54 26 0

Enter second IP address -> 192 54 4 8

MENU:

1. Enter the IP addresses.

2. Print first IP address.

3. Print second IP address.

4. Print a sum of IP addresses.

5. Print a difference of IP addresses.

6. Check if IP address belongs to the subnet with the address of subnet and mask.

Choose a menu item -> 4

IP address is : 255.108.30.8

MENU:

1. Enter the IP addresses.

2. Print first IP address.

3. Print second IP address.

4. Print a sum of IP addresses.

5. Print a difference of IP addresses.

6. Check if IP address belongs to the subnet with the address of subnet and mask.

Choose a menu item -> 5

IP address is : 0.0.22.0

MENU:

1. Enter the IP addresses.

2. Print first IP address.

3. Print second IP address.

4. Print a sum of IP addresses.

5. Print a difference of IP addresses.

6. Check if IP address belongs to the subnet with the address of subnet and mask.

Choose a menu item -> 6

Enter the address of subnet -> 192 168 0 0

Enter the mask -> 255 255 0 0

1. Check if first IP address belongs to the subnet with adress of subnet and mask.

2. Check if second IP address belongs to the subnet with adress of subnet and mask.

Type here -> 1

First address doesn't belong to subnet.

MENU:

1. Enter the IP addresses.

2. Print first IP address.

3. Print second IP address.

4. Print a sum of IP addresses.

5. Print a difference of IP addresses.

6. Check if IP address belongs to the subnet with the address of subnet and mask.

Choose a menu item -> 6

The address of subnet and mask are entered.

1. Check if first IP address belongs to the subnet with adress of subnet and mask.

2. Check if second IP address belongs to the subnet with adress of subnet and mask.

Type here -> 2

Second address doesn't belong to subnet.

1. Листинг программы

/\*

Created by Шавандрин Фёдор

На github Adi6e

Создать класс IPAddress для работы с адресом в интернет

\*/

#include <iostream>

using namespace std;

#define uc unsigned char

class IPAddress{

public:

IPAddress(){

a\_ = 0;

b\_ = 0;

c\_ = 0;

d\_ = 0;

}//конструктор по умолчанию

IPAddress(int valueA, int valueB, int valueC, int valueD){

a\_ = valueA;

b\_ = valueB;

c\_ = valueC;

d\_ = valueD;

}//конструктор с параметрами

void SetA(int valueA){

if (valueA > 255){

valueA = 255;

a\_ = valueA;

}

if (valueA < 0){

valueA = 0;

a\_ = valueA;

}

a\_ = valueA;

}//сеттер

void SetB(int valueB){

if (valueB > 255){

valueB = 255;

b\_ = valueB;

}

if (valueB < 0){

valueB = 0;

b\_ = valueB;

}

b\_ = valueB;

}//сеттер

void SetC(int valueC){

if (valueC > 255){

valueC = 255;

c\_ = valueC;

}

if (valueC < 0){

valueC = 0;

c\_ = valueC;

}

c\_ = valueC;

}//сеттер

void SetD(int valueD){

if (valueD > 255){

valueD = 255;

d\_ = valueD;

}

if (valueD < 0){

valueD = 0;

d\_ = valueD;

}

d\_ = valueD;

}//сеттер

uc GetA(){

return a\_;

}

uc GetB(){

return b\_;

}

uc GetC(){

return c\_;

}

uc GetD(){

return d\_;

}

void PrintIP(){

cout << "IP address is : " << (int)a\_ << "." << (int)b\_ << "." << (int)c\_ << "." << (int)d\_ << endl;

}

private:

uc a\_;

uc b\_;

uc c\_;

uc d\_;

};

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

int x1 = lhs.GetA(), x2 = rhs.GetA(), y1 = lhs.GetB(), y2 = rhs.GetB(), z1 = lhs.GetC(), z2 = rhs.GetC(), k1 = lhs.GetD(), k2 = rhs.GetD();

IPAddress temp;

if (x1 + x2 > 255){

temp.SetA(255);

}

else temp.SetA(x1 + x2);

if (y1 + y2 > 255){

temp.SetB(255);

}

else temp.SetB(y1 + y2);

if (z1 + z2 > 255){

temp.SetC(255);

}

else temp.SetC(z1 + z2);

if (k1 + k2 > 255){

temp.SetD(255);

}

else temp.SetD(k1 + k2);

return temp;

}//оператор сложения двух адресов

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

int x1 = lhs.GetA(), x2 = rhs.GetA(), y1 = lhs.GetB(), y2 = rhs.GetB(), z1 = lhs.GetC(), z2 = rhs.GetC(), k1 = lhs.GetD(), k2 = rhs.GetD();

IPAddress temp;

if (x1 - x2 < 0){

temp.SetA(0);

}

else temp.SetA(x1 - x2);

if (y1 - y2 < 0){

temp.SetB(0);

}

else temp.SetB(y1 - y2);

if (z1 - z2 < 0){

temp.SetC(0);

}

else temp.SetC(z1 - z2);

if (k1 - k2 < 0){

temp.SetD(0);

}

else temp.SetD(k1 - k2);

return temp;

}//оператор вычитания двух адресов

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

IPAddress temp;

temp.SetA(lhs.GetA() & rhs.GetA());

temp.SetB(lhs.GetB() & rhs.GetB());

temp.SetC(lhs.GetC() & rhs.GetC());

temp.SetD(lhs.GetD() & rhs.GetD());

return temp;

}

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

return (lhs.GetA() == rhs.GetA()) && (lhs.GetB() == rhs.GetB()) && (lhs.GetC() == rhs.GetC()) && (lhs.GetD() == rhs.GetD());

}

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

return !((lhs.GetA() == rhs.GetA()) && (lhs.GetB() == rhs.GetB()) && (lhs.GetC() == rhs.GetC()) && (lhs.GetD() == rhs.GetD()));

}

bool operator>(IPAddress& lhs, IPAddress& rhs){

if (lhs.GetA() != rhs.GetA()) return lhs.GetA() > rhs.GetA();

if (lhs.GetB() != rhs.GetB()) return lhs.GetB() > rhs.GetB();

if (lhs.GetC() != rhs.GetC()) return lhs.GetC() > rhs.GetC();

if (lhs.GetD() != rhs.GetD()) return lhs.GetD() > rhs.GetD();

return false;

}

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

if (lhs.GetA() != rhs.GetA()) return lhs.GetA() < rhs.GetA();

if (lhs.GetB() != rhs.GetB()) return lhs.GetB() < rhs.GetB();

if (lhs.GetC() != rhs.GetC()) return lhs.GetC() < rhs.GetC();

if (lhs.GetD() != rhs.GetD()) return lhs.GetD() < rhs.GetD();

return false;

}

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

return lhs.GetA() >= rhs.GetA();

}

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

return lhs.GetA() <= rhs.GetA();

}

IPAddress SumIP(IPAddress& lhs, IPAddress& rhs){

return lhs + rhs;

}

IPAddress DifferenceIP(IPAddress& lhs, IPAddress& rhs){

return lhs - rhs;

}

void EnterIP(IPAddress& ip, int a, int b, int c, int d){

ip.SetA(a);

ip.SetB(b);

ip.SetC(c);

ip.SetD(d);

}

bool CheckIP(IPAddress& need\_to\_check, IPAddress& adress\_of\_subnet, IPAddress& mask){

IPAddress temp = need\_to\_check & mask;

return temp == adress\_of\_subnet;

}

int main(){

IPAddress first, second, need, sub, mask;

bool entered = false , chosen = false, start = true;

int choice;

while(start){

cout << "MENU:" << endl << "1. Enter the IP addresses." << endl << "2. Print first IP address." << endl << "3. Print second IP address." << endl << "4. Print a sum of IP addresses." << endl << "5. Print a difference of IP addresses." << endl << "6. Check if IP address belongs to the subnet with the address of subnet and mask." << endl << "Choose a menu item -> ";

cin >> choice;

switch (choice){

case 1:

if (entered){

cout << "The IP addresses are already entered." << endl << endl;

break;

}

entered = true;

int a,b,c,d;

cout << "Enter first IP address -> ";

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

EnterIP(first,a,b,c,d);

cout << "Enter second IP address -> ";

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

EnterIP(second,a,b,c,d);

cout << endl;

break;

case 2:

if (!entered){

cout << "First IP address is not entered." << endl << endl;

break;

}

first.PrintIP();

cout << endl;

break;

case 3:

if (!entered){

cout << "Second IP address is not entered." << endl << endl;

break;

}

second.PrintIP();

cout << endl;

break;

case 4:

if (!entered){

cout << "IP addresses are not entered." << endl << endl;

break;

}

SumIP(first,second).PrintIP();

cout << endl;

break;

case 5:

if (!entered){

cout << "IP addresses are not entered." << endl << endl;

break;

}

DifferenceIP(first,second).PrintIP();

cout << endl;

break;

case 6:

if (!entered){

cout << "IP addresses are not entered." << endl << endl;

break;

}

if (chosen){

cout << "The address of subnet and mask are entered." << endl;

}

else{

chosen = true;

cout << "Enter the address of subnet -> ";

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

EnterIP(sub,a,b,c,d);

cout << "Enter the mask -> ";

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

EnterIP(mask,a,b,c,d);

}

int key;

cout << "1. Check if first IP address belongs to the subnet with adress of subnet and mask." << endl << "2. Check if second IP address belongs to the subnet with adress of subnet and mask." << endl << "Type here -> ";

cin >> key;

if (key == 1){

if (CheckIP(first, sub,mask)){

cout << "First address belongs to subnet." << endl << endl;

break;

}

else{

cout << "First address doesn't belong to subnet." << endl << endl;

break;

}

}

if (key == 2){

if (CheckIP(second, sub,mask)){

cout << "Second address belongs to subnet." << endl << endl;

break;

}

else{

cout << "Second address doesn't belong to subnet." << endl << endl;

break;

}

}

else{

cout << "There is no such menu item." << endl << endl;

break;

}

break;

default:

cout << "There is no such menu item.";

start = false;

break;

}

}

return 0;

}