**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: Савров Никита

Группа: 80-207

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2019

1. Постановка задачи

Создать класс Address для работы с адресами домов. Адрес должен состоять из строк с названием города и

улицы и чисел с номером дома и квартиры. Реализовать операции сравнения адресов, а также операции

проверки принадлежности адреса к улице и городу. В операциях не должен учитываться регистр строки. Так

же необходимо сделать операцию, которая возвращает истину если два адреса находятся по соседству (на

одной улице в одном городе и дома стоят подряд).

1. Описание программы

Для выполнения программы создадим класс Address содержащий поля city,street,house,apartment. Для присвоения значений создадим функцию Set в которую будем передавать нужные значения. Для сравнения двух адресов напишем перегрузку оператора ==, в котором будем сравнивать каждое поле. Так же напишем функции is\_city и is\_street для проверки принадлежности адреса к городу и улице. Для выполнения последнего пункта задания напишем функцию is\_neighbour для проверки являются ли адреса соседними. В функции main инстанциируем 2 объекта класса Address и будем вызывать к ним нужные функции.

1. Набор testcases

test\_01

1

MoscoW lenina 10 11

2

MOScow leninA 10 11

3

4

Leningrad 1

5

lenina 2

6

0

test\_02

1

MOScow lenina 10 11

2

Leningrad mira 11 12

3

4

Moscow 1

5

lenina 2

6

0

test\_03

1

MosCOw lenina 10 13

2

Moscow lenina 11 14

3

4

Stalingrad 1

5

mira 2

6

0

1. Результаты выполнения тестов.

test\_01

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

1

MoscoW lenina 10 11

2

MOScow leninA 10 11

3

4

Leningrad 1

5

lenina 2

6

0

Enter Address 1

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

Enter Address 2

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

equal

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

enter city and number address

this address not locate in Leningrad

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

enter sreet and number address

this address locate in lenina

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

this addresses are not neighbour

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

test\_02

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

1

MOScow lenina 10 11

2

Leningrad mira 11 12

3

4

Moscow 1

5

lenina 2

6

0

Enter Address 1

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

Enter Address 2

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

not equal

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

enter city and number address

this address locate in Moscow

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

enter street and number address

this address locate in lenina

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

this addresses are not neighbour

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

test\_03

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

1

MosCOw lenina 10 13

2

Moscow lenina 11 14

3

4

Stalingrad 1

5

mira 2

6

0

Enter Address 1

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

Enter Address 2

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

not equal

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

enter city and number address

this address not locate in Stalingrad

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

enter sreet and number address

this address not locate in mira

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

this addresses are neighbour

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

1 - input address 1

2 - input address 2

3 - compare addresses

4 - this address locate in this city

5 - this address locate on this street

6 - are this addresses neighbour

0 - exit

1. Листинг программы

// Лабораторная работа №1 Савров Никита М80-207Б-18

// Создать класс Address для работы с адресами домов. Адрес должен состоять из строк с названием города и

// улицы и чисел с номером дома и квартиры. Реализовать операции сравнения адресов, а также операции

// проверки принадлежности адреса к улице и городу. В операциях не должен учитываться регистр строки. Так

// же необходимо сделать операцию, которая возвращает истину если два адреса находятся по соседству (на

// одной улице в одном городе и дома стоят подряд).

#include <iostream>

#include <string>

#include <algorithm>

#include <cmath>

using namespace std;

class Address{

private:

string city;

string street;

int house;

int apartment;

public:

Address(){

city = "Moscow";

street = "Lenina";

house = 10;

apartment = 10;

}

void Set(string s1, string s2, int a1, int a2){

city = s1;

street = s2;

house = a1;

apartment = a2;

}

bool operator ==(Address a){

transform(a.city.begin(), a.city.end(), a.city.begin(), (int (\*)(int))tolower);

transform(this->city.begin(), this->city.end(), this->city.begin(), (int (\*)(int))tolower);

transform(a.street.begin(), a.street.end(), a.street.begin(), (int (\*)(int))tolower);

transform(this->street.begin(), this->street.end(), this->street.begin(), (int (\*)(int))tolower);

return a.city == this->city && a.street == this->street && a.house == this->house && a.apartment == this->apartment;

}

bool is\_city(string town){

transform(town.begin(), town.end(), town.begin(), (int (\*)(int))tolower);

transform(this->city.begin(), this->city.end(), this->city.begin(), (int (\*)(int))tolower);

return this->city == town;

}

bool is\_street(string stree){

transform(stree.begin(), stree.end(), stree.begin(), (int (\*)(int))tolower);

transform(this->street.begin(), this->street.end(), this->street.begin(), (int (\*)(int))tolower);

return this->street == stree;

}

bool is\_neighbour(Address a){

transform(a.city.begin(), a.city.end(), a.city.begin(), (int (\*)(int))tolower);

transform(this->city.begin(), this->city.end(), this->city.begin(), (int (\*)(int))tolower);

transform(a.street.begin(), a.street.end(), a.street.begin(), (int (\*)(int))tolower);

transform(this->street.begin(), this->street.end(), this->street.begin(), (int (\*)(int))tolower);

if (a.city == this->city && a.street == this->street){

if (abs(a.house - this->house) == 1){

return true;

}

}

return false;

}

};

int main(){

Address \*a = new Address();

Address \*b = new Address();

int item = 1;

string name1,name2,town,stree;

int adress;

int name3,name4;

cout << "1 - input address 1\n2 - input address 2\n3 - compare addresses\n4 - this address locate in this city\n5 - this address locate on this street\n6 - are this addresses neighbour\n0 - exit\n";

while (item != 0){

cin >> item;

switch (item) {

case 1:

cout << "Enter Address 1\n";

cin >> name1 >> name2 >> name3 >> name4;

a->Set(name1, name2, name3, name4);

break;

case 2:

cout << "Enter Address 2\n";

cin >> name1 >> name2 >> name3 >> name4;

b->Set(name1, name2, name3, name4);

break;

case 3:

if (\*a == \*b)

cout << "equal\n";

else

cout << "not equal\n";

break;

case 4:

cout << "enter city and number address\n";

cin >> town >> adress;

if (adress == 1){

if (a->is\_city(town))

cout << "this address locate in " << town << '\n';

else

cout << "this address not locate in " << town << '\n';

break;

}

if (adress == 2){

if (b->is\_city(town))

cout << "this address locate in " << town << '\n';

else

cout << "this address not locate in " << town << '\n';

break;

}

cout << "this number not exist\n";

break;

case 5:

cout << "enter street and number address\n";

cin >> stree >> adress;

if (adress == 1){

if (a->is\_street(stree))

cout << "this address locate in " << stree << '\n';

else

cout << "this address not locate in " << stree << '\n';

break;

}

if (adress == 2){

if (a->is\_street(stree))

cout << "this address locate in " << stree << '\n';

else

cout << "this address not locate in " << stree << '\n';

break;

}

cout << "this number not exist\n";

break;

case 6:

if (a->is\_neighbour(\*b)){

cout << "this addresses are neighbour\n";

}else

cout << "this addresses are not neighbour\n";

break;

}

cout << "1 - input address 1\n2 - input address 2\n3 - compare addresses\n4 - this address locate in this city\n5 - this address locate on this street\n6 - are this addresses neighbour\n0 - exit\n";

}

delete(a);

delete(b);

}

6. Вывод

Данная программа может быть полезна для работы с адресами. Так же ее можно улучшить добавив дополнительные функции вроде проверки нахождения кваритир в одном доме и прочих. Благодаря парадигме ООП мы можем представлять пользовательские типы данных в виде объектов что значительно упрощает программирование.

7. Литература

1.Бьерн Страуструп «Язык программирования С++»

2.Информационный портал [электронный ресурс] URL: [www.habr.com](http://www.habr.com/)

(дата обращения 14.09.2019)