**Московский авиационный институт**

**(Национальный исследовательский университет)**

Институт: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Объектно-ориентированное программирование»

**Лабораторная работа № 2**

Тема: Перегрузка операторов в С++

Студент: Хренникова Ангелина Сергеевна

Группа: М8О-208Б-19

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2020

**Cодержание**

1. Постановка задачи
2. Описание программы
3. Набор тестов
4. Результаты выполнения тестов
5. Листинг программы
6. Вывод
7. Список литературы

**Постановка задачи**

Цель:

* Изучить механизмы перегрузки операторов;
* Изучить механизмы работы с пользовательскими литералами.

Задача: Разработать программу на языке C++ согласно варианту задания. Программа на C++ должна собираться с помощью системы сборки CMake. Программа должна получать данные из стандартного ввода и выводить данные в стандартный вывод.

Вариант(№19): 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).

Операцию сравнения равенства реализовать в виде перегрузки оператора.

Операцию нахождения «по соседству» реализовать в виде перегрузки оператора &. Необходимо реализовать пользовательский литерал для работы с константами типа Address.

**Описание программы**

Класс Address, определенный в программе, содержит четыре поля: City - название города, Street - название улицы, House - номер дома, Door - номер квартиры.

В методах класса содержится конструктор для инициализации полей объекта, конструктор копирования, деструктор, функции, возвращающие значение улицы и города, функция сравнения строк, функция, проверяющая принадлежность адреса к улице и городу, а также перегруженные операторы для сравнения двух адресов и проверки на соседство.

Отдельно описан пользовательский литерал, который преобразует адрес, записанный одной строкой, в объект класса и выводит его.

Меню, с помощью которого пользователь может выбрать необходимое действие (сравнить адреса, проверить на соседства, проверить на принадлежность к улице и городу, проверить работу пользовательского литерала, выйти из программы).

Главная функция запрашивает у пользователя два адреса, которые помещаются в объекты класса, выводит меню, после чего пользователь должен выбрать один из пунктов. В каждом из пунктов с помощью описанных функций производятся необходимые действия и выводится ответ. При неверном выборе пункта меню, выводится сообщение об ошибке. Если не был выбран пункт о выходе из программы, запрашивается повторный выбор пункта меню.

**Набор тестов**

Входными данными являются два адреса, где город, улица, дом и квартира записываются отдельно, и пункты меню.

Входные данные теста 1: Moscow Durova 4 14 moscow Durova 6 7 1 2 3 4 5

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Город | Улица | Дом | Квартира |
| Адрес 1 | Moscow | Durova | 4 | 14 |
| Адрес 2 | moscow | Durova | 6 | 7 |

Вывод программы:

|  |  |
| --- | --- |
| Равенство адресов | Есть |
| Принадлежность к городу и улице | Есть |
| Соседство адресов | Есть |

Входные данные теста 2: Orenburg Kirova 4 17 Omsk Lenina 17 4 1 2 3 4 5

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Город | Улица | Дом | Квартира |
| Адрес 1 | Orenburg | Kirova | 4 | 17 |
| Адрес 2 | Omsk | Lenina | 17 | 4 |

Вывод программы:

|  |  |
| --- | --- |
| Равенство адресов | Отсутствует |
| Принадлежность к городу и улице | Отсутствует |
| Соседство адресов | Отсутствует |

Входные данные теста 3: Moscow Arbat 178 4 Moscow Arbat 5 102 1 2 3 4 5

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Город | Улица | Дом | Квартира |
| Адрес 1 | Moscow | Arbat | 178 | 4 |
| Адрес 2 | Moscow | Arbat | 5 | 102 |

Вывод программы:

|  |  |
| --- | --- |
| Равенство адресов | Отсутствует |
| Принадлежность к городу и улице | Есть |
| Соседство адресов | Отсутствует |

Работа пользовательского литерала:

Входные данные: Moscow,Arbat,1,1(строка символов)

Вывод программы:

|  |  |
| --- | --- |
| Город | Moscow |
| Улица | Arbat |
| Дом | 1 |
| Квартира | 1 |

**Результаты выполнения тестов**

Тест 1:

Input first address city: Moscow

Input first address street: Durova

Input first address house number: 4

Input first address door number: 14

Input second address city: moscow

Input second address street: Durova

Input second address house number: 6

Input second address door number: 7

Menu:

1. Check if addresses are equal.

2. Check if the address belongs to the street and city.

3. Check if two addresses are in the neighborhood.

4. Test user-defined literal.

5. Exit.

Select menu item:

1

Equal? NO

Select menu item:

2

Same city and street? YES

Select menu item:

3

Neighbors? YES

Select menu item:

4

Checking the work of the user-defined literal for the address written in one line.

The literal must split the string into multiple class fields.

String format: "City,Street,House,Door".

String: Moscow,Arbat,1,1

City: Moscow

Street: Arbat

House: 1

Door: 1

Select menu item:

5

Тест 2:

Input first address city: Orenburg

Input first address street: Kirova

Input first address house number: 4

Input first address door number: 17

Input second address city: Omsk

Input second address street: Lenina

Input second address house number: 17

Input second address door number: 4

Menu:

1. Check if addresses are equal.

2. Check if the address belongs to the street and city.

3. Check if two addresses are in the neighborhood.

4. Test user-defined literal.

5. Exit.

Select menu item:

1

Equal? NO

Select menu item:

2

Same city and street? NO

Select menu item:

3

Neighbors? NO

Select menu item:

4

Checking the work of the user-defined literal for the address written in one line.

The literal must split the string into multiple class fields.

String format: "City,Street,House,Door".

String: Moscow,Arbat,1,1

City: Moscow

Street: Arbat

House: 1

Door: 1

Select menu item:

5

Тест 3:

Input first address city: Moscow

Input first address street: Arbat

Input first address house number: 178

Input first address door number: 4

Input second address city: Moscow

Input second address street: Arbat

Input second address house number: 5

Input second address door number: 102

Menu:

1. Check if addresses are equal.

2. Check if the address belongs to the street and city.

3. Check if two addresses are in the neighborhood.

4. Test user-defined literal.

5. Exit.

Select menu item:

1

Equal? NO

Select menu item:

2

Same city and street? YES

Select menu item:

3

Neighbors? NO

Select menu item:

4

Checking the work of the user-defined literal for the address written in one line.

The literal must split the string into multiple class fields.

String format: "City,Street,House,Door".

String: Moscow,Arbat,1,1

City: Moscow

Street: Arbat

House: 1

Door: 1

Select menu item:

5

**Листинг программы**

/\* Хренникова Ангелина, М8О-208Б-19, 19 вариант

\*

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).

Операцию сравнения равенства реализовать в виде перегрузки оператора. Операцию нахождения «по соседству» реализовать

в виде перегрузки оператора &.

Необходимо реализовать пользовательский литерал для работы с константами типа Address.

\*/

#include <iostream>

#include <cctype>

#include <string>

#include <algorithm>

using namespace std;

class Address { //класс адреса

private:

string City, Street; //город, улица

long long House, Door; //дом, квартира

public:

Address() : City(), Street(), House(), Door() {};

Address(const string& city, const string& street, const long long& house, const long long& door)

: City(city), Street(street), House(house), Door(door) {};

~Address() {};

string GetCity() {

return City;

}

string GetStreet() {

return Street;

}

static bool CompareString(const string& lhs, const string& rhs) { //Сравнение строк

if (lhs.size() == rhs.size()) {

for (int i = 0; i < lhs.size(); ++i) {

if (tolower(lhs[i]) != tolower(rhs[i])) {

return false;

}

}

return true;

}

else {

return false;

}

}

static bool IsInCity(const Address& address, const string& city, const string& street) { //Принадлежность к городу и улице

return CompareString(address.City, city) && CompareString(address.Street, street);

}

friend bool operator== (const Address& lhs, const Address& rhs) {

if (CompareString(lhs.City, rhs.City) && CompareString(lhs.Street, rhs.Street)

&& (lhs.House == rhs.House) && (lhs.Door == rhs.Door))

return true;

else return false;

}

friend bool operator& (const Address& lhs, const Address& rhs) {

if (CompareString(lhs.City, rhs.City) && CompareString(lhs.Street, rhs.Street)

&& (abs((int)lhs.House - (int)rhs.House) == 2 or rhs.House == lhs.House) == true)

return true;

else return false;

}

};

const Address operator"" \_address(const char\* str, size\_t size) { //Пользовательский литерал, который разбивает строку с адресом на отдельные поля

int position = 0;

string city;

while (str[position] != ',') {

city += str[position];

position += 1;

}

position += 1;

string street;

while (str[position] != ',') {

street += str[position];

position += 1;

}

position += 1;

string houseStr;

while (str[position] != ',') {

houseStr += str[position];

position += 1;

}

long long house = stoi(houseStr);

position += 1;

string doorStr;

while (position != size) {

doorStr += str[position];

position += 1;

}

long long door = stoi(doorStr);

Address result(city, street, house, door);

cout << "City: " << city << "\nStreet: " << street << "\nHouse: " << house << "\nDoor: " << door << "\n";

return result;

}

void menu() { //Меню

cout << "Menu:\n";

cout << "1. Check if addresses are equal.\n";

cout << "2. Check if the address belongs to the street and city.\n";

cout << "3. Check if two addresses are in the neighborhood.\n";

cout << "4. Test user-defined literal.\n";

cout << "5. Exit.\n";

}

signed main() {

string city, street;

long long house = 0, door = 0;

cout << "Input first address city: "; //Ввод первого адреса

cin >> city;

cout << "Input first address street: ";

cin >> street;

cout << "Input first address house number: ";

cin >> house;

cout << "Input first address door number: ";

cin >> door;

Address a = Address(city, street, house, door);

house = 0, door = 0;

cout << "Input second address city: "; //Ввод второго адреса

cin >> city;

cout << "Input second address street: ";

cin >> street;

cout << "Input second address house number: ";

cin >> house;

cout << "Input second address door number: ";

cin >> door;

Address b = Address(city, street, house, door);

cout << endl;

if (house <= 0 || door <= 0) { //Ошибка при попытке ввода 0 номера

cout << "Invalid input! Program terminates with exit code -1\n";

return -1;

}

menu(); //Вывод меню

int op;

cout << "Select menu item:\n"; //Выбор пункта меню

cin >> op;

while (op >= 1 && op <= 4) {

if (op == 1) { //Равенство

cout << "Equal? " << (a == b ? "YES" : "NO") << "\n";

}

else if (op == 2) { //Принадлежность к городу и улице

cout << "Same city and street? " << (Address::IsInCity(a, b.GetCity(), b.GetStreet()) ? "YES" : "NO") << "\n";

}

else if (op == 3) { //Соседи

cout << "Neighbors? " << (a & b ? "YES" : "NO") << "\n";

}

else if (op == 4) { //Пользовательский литерал

cout << "Checking the work of the user-defined literal for the address written in one line.\n";

cout << "The literal must split the string into multiple class fields.\n";

cout << "String format: \"City,Street,House,Door\".\n";

cout << "String: " << "Moscow,Arbat,1,1\n";

"Moscow,Arbat,1,1"\_address;

}

else if (op == 5)

break;

else

cout << "Error!";

cout << "Select menu item:\n";

cin >> op;

}

return 0;

}

**Вывод**

Изучила механизмы перегрузки операторов, изучила механизмы работы с пользовательскими литералами на основе работы с классом Address для работы с адресами, задаваемыми городом, улицей, домом и квартирой.

**Список литературы**

1. Хабр - Пользовательские литералы в С++11/Электронный диплом/Режим доступа: <https://habr.com/ru/post/140357/> (дата обращения: 05.10.2020).
2. Р. Лафоре, Объектно - ориентированное программирование в С++, 4-е издание. -М.: Питер, 2004 - 924 с. (дата обращения: 05.10.2020).