Міністерство освіти і науки України

Національний університет „Львівська політехніка”

Кафедра ЕОМ

**ЛАБОРАТОРНА РОБОТА № 5.**

**Перевантаження операторів**

Варіант 16

Виконав: ст.гр. КІ-15

Леземезюк Р.Т.

Прийняв: асистент

Козак Н.Б.

Львів – 2020

***Мета: познайомитися із перевантаженням операторів.***

***ТЕОРЕТИЧНІ ВІДОМОСТІ***

***Перевантаження операторів***

***Кожному оператору мова С++ ставить у відповідність ім'я функції, що складається з ключового слова operator, власне оператору та аргументів відповідних типів:***

***тип operator символОператору (списокПараметрів) { //тіло методу }***

***Щоб використовувати операцію над об'єктами класів, ця операція повинна бути перевантажена, але є два виключення. Операції присвоювання (=) і взяття адреси (&) створюються в класі автоматично за замовчуванням, тому їх можна використовувати без явного перевантаження. За замовчуванням операція присвоювання зводиться до побітового копіювання даних-елементів класу. Проте таке побітове копіювання небезпечне для класів з елементами, що вказують на динамічно виділені області пам'яті, масиви, рядки, оскільки в цьому випадку відбувається копіювання не даних (глибоке копіювання), а лише вказівників на дані (поверхневе копіювання). Для таких класів слід явно перевантажувати операцію присвоювання і здійснювати у ній глибоке копіювання. Операція адресації також може бути використана з об'єктами будь-яких класів без перевантаження. Вона просто повертає адресу об'єкта в пам'яті. Але операцію адресації можна також і перевантажувати. Перевантаження операцій підпорядковується наступним правилам:***

* ***При перевантаженні зберігаються кількість аргументів, пріоритети операцій та правила асоціації, що використовуються у стандартних типах даних;***
* ***Для стандартних типів даних операції не підлягають перевизначенню;***
* ***Перевантажена функція-оператор не може мати параметрів по замовчуванню, не успадковується та не може бути визначеною як static;***
* ***Функція-оператор може бути визначена трьома способами – метод класу, дружня функція або звичайна функція. В останніх двох випадках вона повинна приймати хоча б один аргумент, що має тип класу, вказівника або посилання на клас. При перевантаженні операцій ( ), [], -> та = функція перевантаження операції може бути оголошена лише як метод класу. Для інших операцій функції перевантаження операцій можуть не бути методами класу.***

**Код програми**

#include <iostream>

#include <iomanip>

#include <Windows.h>

#include <vector>

#include <string>

using namespace std;

class CTV

{

private:

int Volume, Resourse, NowChannel;

bool OnOff;

string Channels[6] = { "New Channel","ICTV","NlO TV","1+1","2+2","STB" };

public:

CTV()

{

Volume = 0;

Resourse = 10;

NowChannel = 0;

OnOff = false;

}

~CTV() {};

CTV(int Volume, int Resourse, int NowChannel, bool OnOff)

{

this->Volume = Volume;

this->Resourse = Resourse;

this->NowChannel = NowChannel;

this->OnOff = OnOff;

}

void Information()

{

if (OnOff == false)

{

cout << "TV i off, no info\n\n";

}

else

{

cout << "Channel: " << NowChannel << endl << Channels[NowChannel] << "\nVolume: " << Volume << "\nResourses: " << Resourse << endl << endl;

}

}

void ChangeVolume(int i)

{

if (i > 0)

{

if (Volume + i > 100)

{

Volume = 100;

}

else

{

Volume += i;

}

}

else

{

if (Volume + i < 0)

{

Volume = 0;

}

else

{

Volume -= i;

}

}

}

void ChangeChannel(int i)

{

if (i < 0 && i>5)

{

cout << "No such a channel\n\n";

}

else

{

NowChannel = i;

}

}

void Watch()

{

if (Resourse < 0)

{

cout << "You have no resourses\n\n";

}

else

{

Resourse--;

}

}

void Put(bool choose)

{

OnOff = choose;

}

CTV operator ++ (int)

{

ChangeVolume(1);

return \*this;

}

CTV& operator += (const CTV& TV)

{

ChangeVolume(TV.Volume);

return \*this;

}

bool operator < (CTV& TV)

{

return Volume < TV.Volume;

};

bool operator > (CTV& TV)

{

return Volume > TV.Volume;

};

};

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int choose, amount;

bool take;

CTV TV1, TV2;

do

{

cout << "Information(0)\nChange volume(1)\nChange volume TV1++(2)\nChange channel(3)\nWatch TV1(4)\nOnOff TV1(5)\nWatch TV2(6)\nOnOff TV2(7)\nTV1 += TV2(8)\nExit(9)\n";

cin >> choose;

switch (choose)

{

case 0:

system("cls");

TV1.Information();

cout << endl;

TV2.Information();

break;

case 1:

system("cls");

cout << "TV1 Enter amount: ";

cin >> amount;

TV1.ChangeVolume(amount);

cout << endl;

cout << "TV2 Enter amount: ";

cin >> amount;

TV2.ChangeVolume(amount);

break;

case 2:

system("cls");

TV1++;

break;

case 3:

system("cls");

cout << "TV1 Enter channel: ";

cin >> amount;

TV1.ChangeChannel(amount);

cout << endl;

cout << "TV2 Enter channel: ";

cin >> amount;

TV2.ChangeChannel(amount);

break;

case 4:

system("cls");

TV1.Watch();

break;

case 5:

system("cls");

cout << "OFF(0)\nON(1)\n";

cin >> take;

TV1.Put(take);

break;

case 6:

system("cls");

TV2.Watch();

break;

case 7:

system("cls");

cout << "OFF(0)\nON(1)\n";

cin >> take;

TV2.Put(take);

break;

case 8:

system("cls");

TV1 += TV2;

break;

case 9:

system("cls");

return 0;

default:

system("cls");

cout << "Enter corect!!!" << endl;

continue;

}

} while (true);

system("pause");

return 0;

**}**

**Результат.**