Міністерство освіти і науки України

Національний університет „Львівська політехніка”

Кафедра “Електронних обчислювальних машин”



**Звіт з лабораторної роботи №5**

на тему:

“ПЕРЕВАНТАЖЕННЯ ОПЕРАТОРІВ”

**Виконав:**

Ст. гр. КІ-15

Зектер В.О.

**Перевірив:**

Викладач

Козак Н.Б.

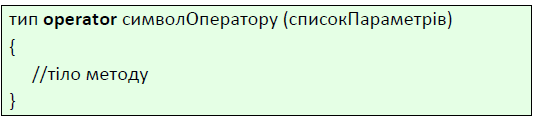
Львів – 2020

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

**Теоретичні відомості:**

**Перевантаження операторів**

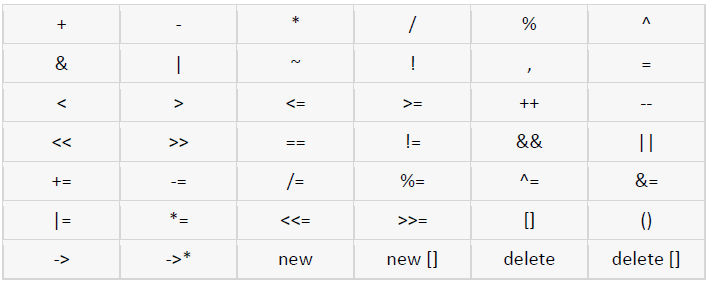
Кожному оператору мова С++ ставить у відповідність ім'я функції, що складається з ключового слова operator, власне оператору та аргументів відповідних типів:



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.

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.

Оператори, які можна перевантажити:



Оператори, які не можна перевантажити:

- sizeof

- . (селектор елемента структури або класу)

- \* (оператор доступу до елементу за вказівником)

- :: (оператор дозволу видимості)

- ?: (тернарний оператор)

- typeid

- const\_cast

- dynamic\_cast

- reinterpret\_cast

- static\_cast

- # і ## (символи препроцесору)

***Завдання:***

Розширити функціональність розроблених у 4 лабораторній роботі класів за допомогою операторів, що задані варіантом та оператора присвоювання. Конкретні функції операторів реалізувати на власний розсуд (крім оператора присвоювання). Організувати виведення та введення даних за допомогою класів-потоків сin, cout та перевантажених операторів вводу/виводу. Написати програму, яка демонструє роботу з об'єктами цього класу.



***Код програми:***

#include <iostream>

#include <string>

#include <ctime>

using namespace std;

class CTank {

private:

int boyekomplect, toplivo, popadannya;

int imovirnist\_popadannya;

public:

// Operator's

CTank& operator = (const CTank& other) {

this->boyekomplect = other.boyekomplect;

this->toplivo = other.toplivo;

this->popadannya = other.popadannya;

this->imovirnist\_popadannya = other.imovirnist\_popadannya;

return \*this;

}

bool operator == (const CTank& other)

{

return this->boyekomplect == other.boyekomplect && this->toplivo == other.toplivo && this->popadannya == other.popadannya && this->imovirnist\_popadannya == other.imovirnist\_popadannya;

}

CTank operator += (const CTank& other)

{

this->boyekomplect = other.boyekomplect + 1;

this->toplivo = other.toplivo + 1;

this->popadannya = other.popadannya + 1;

return \*this;

}

CTank operator -= (const CTank& other)

{

this->boyekomplect = other.boyekomplect - 1;

this->toplivo = other.toplivo - 1;

this->popadannya = other.popadannya - 1;

return \*this;

}

bool operator >(const CTank& other)

{

return (this->boyekomplect > other.boyekomplect && this->toplivo > other.toplivo && this->popadannya > other.popadannya && this->imovirnist\_popadannya > other.imovirnist\_popadannya);

}

CTank(int Vboyekomplect, int Vtoplivo, int Vpopadannya, int Vimovirnist\_popadannya) {

boyekomplect = Vboyekomplect;

toplivo = Vtoplivo;

popadannya = Vpopadannya;

imovirnist\_popadannya = Vimovirnist\_popadannya;

}

//Setters

void Set\_boyekomplect(int Sboyekomplect)

{

boyekomplect = Sboyekomplect;

}

void Set\_toplivo(int Stoplivo)

{

toplivo = Stoplivo;

}

void Set\_popadannya(int Spopadannya)

{

popadannya = Spopadannya;

}

void Set\_imovirnist\_popadannya(int Simovirnist\_popadannya)

{

imovirnist\_popadannya = Simovirnist\_popadannya;

}

//Getters

int Get\_boyekomplect()

{

return boyekomplect;

}

int Get\_tolivo()

{

return toplivo;

}

int Get\_popadannya()

{

return popadannya;

}

int Get\_imovirnist\_popadannya()

{

return imovirnist\_popadannya;

}

//Shooting

void Shooting()

{

int vistrel;

srand(time(NULL));

cout << "To shoot press 1" << endl;

int choice;

cin >> choice;

if (choice == 1) {

vistrel = rand() % 2;

if (vistrel == 1) {

popadannya++;

cout << "The target is hit.Number of hits: " << popadannya << endl;

}

else {

cout << "You missed" << endl;

}

}

}

//Moving

void Moving()

{

cout << "To start moving press 2" << endl;

int choicee;

cin >> choicee;

if (choicee == 2) {

toplivo = toplivo - 10;

cout << "The amount of fuel decrease. Fuel = " << toplivo << endl;

}

}

void showdata() {

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "Number of ammunition: " << boyekomplect << endl;

cout << "Amount of fuel: " << toplivo << endl;

cout << "Number of hits: " << popadannya << endl;

cout << "Hit probability: " << imovirnist\_popadannya << endl;

}

};

int main() {

CTank tank1(50, 500, 15, 50);

tank1.showdata();

tank1.Shooting();

tank1.Moving();

tank1.showdata();

CTank tank2(75, 720, 20, 51);

tank2.showdata();

tank2.Shooting();

tank2.Moving();

tank2.showdata();

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl << endl;

cout << "Operator ==" << endl;

cout << "Before operator ==" << endl;

tank1.showdata();

cout << endl<<endl;

tank1 = tank2;

cout << "After operator ==" << endl;

tank1.showdata();

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl<<endl;

cout << "operator += " << endl;

cout << "Before operatore" << endl;

tank2.showdata();

cout << "After operator " << endl;

tank2 += tank2;

tank2.showdata();

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl<<endl;

cout << "operator -= " << endl;

cout << "Before operatore" << endl;

tank2.showdata();

cout << "After operator " << endl;

tank2 -= tank2;

tank2.showdata();

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl << endl;

cout << "operator > " << endl;

cout << "tank 2 bigger than tank 1" << endl;

return 0;

}

***Резульатат виконання:***





***Висновок:***на цій лабораторній роботі я познайомився з перевантаженням операторів.