# Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"

# Факультет інформатики та обчислювальної техніки Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 4 з дисципліни

«Основи програмування»

Варіант 13

Виконав студент Котков Тимур Максимович

(шифр, прізвище, ім'я, по батькові)

Перевірив Вітковська Ірина Іванівна

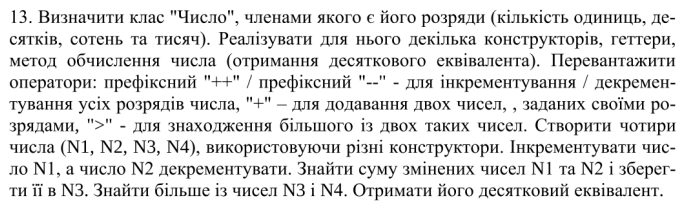
( прізвище, ім'я, по батькові)

Київ 2022

**Лабораторна робота №4**

**Перевантаження операторів**

**Мета –** визначити механізми створення класів та об’єктів зі створення перевантажених операторів.

**Індивідуальне завдання:**

**Постановка задачі**

Розробимо клас Number з декількома (чотирма) конструкторами, що приймають як параметри число, вектор або посилання на інший клас. Реалізуємо перевантаження операторів ++, --, +, >. Проводимо операції зі створенням, додаванням і порівнянням згідно умови, виводимо результат у консоль.

**Код на С++ :**

**Header.h :**

#pragma once

#include <cmath>

#include "Number.h"

using namespace std;

int vectorToNum(Number& num);

**Number.h :**

#pragma once

#include <iostream>

#include <vector>

#include <string>

#include <cmath>

using namespace std;

class Number {

private:

vector<int> getNumVector(int);

vector<int> numByDigit;

friend int vectorToNum(Number& num);

public:

Number(void);

Number(int num);

Number(vector<int>);

Number(const Number& num);

Number operator++();

Number operator--();

Number operator+(Number& obj);

Number operator>(Number& obj);

void setNum(int number);

void outputDigits();

};

**Source.cpp**

#include "Header.h"

#include "Number.h"

int vectorToNum(Number& num) {

int n = 0;

for (int i = 0; i < num.numByDigit.size(); i++)

{

n += num.numByDigit[i] \* pow(10, num.numByDigit.size() - i - 1);

}

return n;

}

**Number.cpp:**

#include "Number.h"

#include "Header.h"

Number::Number(void) {

this-> numByDigit = getNumVector(0);

}

Number::Number(int num) {

this->numByDigit = getNumVector(num);

}

Number::Number(vector<int> v) {

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

{

this->numByDigit.push\_back(v[i]);

}

}

Number::Number(const Number& obj) {

this->numByDigit = obj.numByDigit;

}

vector<int> Number::getNumVector(int number) {

vector<int> numByDigit;

int num\_copy = number;

for (int i = pow(10, trunc(log10(num\_copy))); i > 0; i /= 10)

{

numByDigit.push\_back(num\_copy / i);

num\_copy -= (num\_copy / i) \* i;

}

return numByDigit;

}

void Number::setNum(int number) {

this->numByDigit = getNumVector(number);

}

void Number::outputDigits() {

for (int i = 0; i < this->numByDigit.size(); i++)

{

cout << "10 ^ " << this->numByDigit.size() - i - 1 << ": " << this->numByDigit[i] << endl;

}

cout << "Full number: " << to\_string(vectorToNum(\*this)) << endl << endl;

}

Number Number::operator++()

{

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

{

numByDigit[i]++;

}

return \*this;

}

Number Number::operator--()

{

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

{

numByDigit[i]--;

}

return \*this;

}

Number Number::operator+(Number &obj) {

int diff = this->numByDigit.size() - obj.numByDigit.size();

for (int i = obj.numByDigit.size() - 1; i >= 0; i--)

{

numByDigit[i + diff] += obj.numByDigit[i];

if (numByDigit[i + diff] == 10) {

numByDigit[i + diff - 1] += 1;

numByDigit[i + diff] = 0;

}

}

return \*this;

}

Number Number::operator>(Number& obj) {

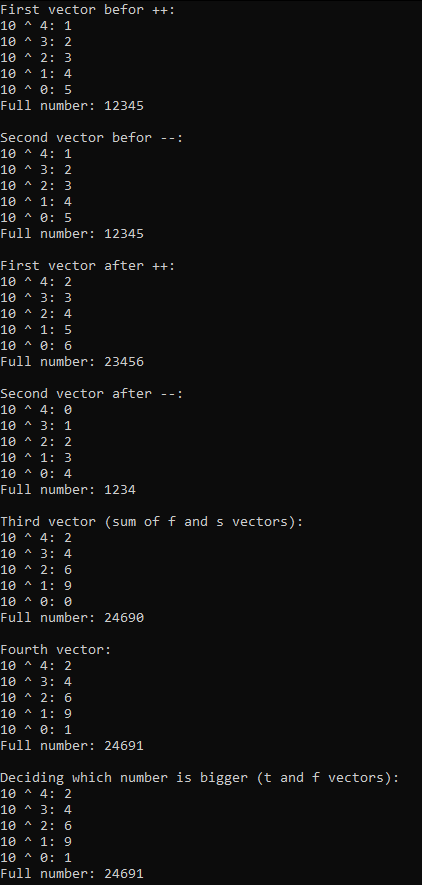
int a = vectorToNum((\*this)), b = vectorToNum(obj);

this->numByDigit = a > b ? getNumVector(a) : getNumVector(b);

return \*this;

}

**Скріншот результатів програми на C++ :**

****

**Висновок:**

Під час лабораторної роботи ми навчилися перевантажувати деякі оператори у класі (такі як ++, --, +, >). Також ознайомилися з friend-функціями у класі. Оскільки програма працює згідно з умовою, то можна сказати, що вона працює правильно.