**Лабораторна робота №2**

## Тема: Перевантаження операцій.

***Мета роботи:*** Одержати практичні навички реалізації класів на С++ та C#.

**Варіант 24**

**Задача 1(4)**

**Умова:**

В клас Money додати:

Перевантаження:

-операції ++ (--): одночасно збільшує (зменшує) значення полів first і second;

-операції !: повертає значення true, якщо поле second не нульове, інакше false;

-операції бінарний +: додає до значення поля second значення скаляра;

-перетворення типу Money в string (і навпаки).

**Код:**

1)Опис класу, Money.h

#ifndef MONEY\_H

#define MONEY\_H

#include <iostream>

#include <string>

using namespace std;

class Money {

private:

int first;

int second;

Money(int f, int s) : first(f), second(s) {}

public:

Money() = delete;

static Money input();

Money& operator++();

Money operator++(int);

Money& operator--();

Money operator--(int);

bool operator!() const;

Money operator+(int scalar) const;

explicit operator string() const;

void print() const;

};

#endif

2)Money.cpp

#include "Money.h"

static bool isAllDigits(const string& s) {

if (s.empty()) return false;

for (size\_t i = 0; i < s.size(); ++i) {

char c = s[i];

if (c < '0' || c > '9') return false;

}

return true;

}

static string trimBoth(const string& s) {

int i = 0;

int j = (int)s.size() - 1;

while (i <= j && (s[i] == ' ' || s[i] == '\t' || s[i] == '\r')) ++i;

while (j >= i && (s[j] == ' ' || s[j] == '\t' || s[j] == '\r')) --j;

if (i > j) return "";

return s.substr(i, j - i + 1);

}

static bool parseNonNegInt(const string& s, int& out) {

try {

int value = stoi(s);

if (value < 0) return false;

out = value;

return true;

}

catch (...) {

return false;

}

}

Money Money::input() {

string line;

int grn = 0;

int kop = 0;

while (true) {

cout << "Введіть гривні (тільки цифри, >= 0): ";

if (!getline(cin, line)) {

cout << "Помилка вводу. Спробуйте ще раз.\n";

cin.clear();

continue;

}

line = trimBoth(line);

if (!parseNonNegInt(line, grn)) {

cout << "Невірний формат. Введіть лише цифри без мінуса.\n";

continue;

}

break;

}

while (true) {

cout << "Введіть копійки (0..99): ";

if (!getline(cin, line)) {

cout << "Помилка вводу. Спробуйте ще раз.\n";

cin.clear();

continue;

}

line = trimBoth(line);

if (!parseNonNegInt(line, kop)) {

cout << "Невірний формат. Введіть лише цифри від 0 до 99.\n";

continue;

}

if (kop < 0 || kop > 99) {

cout << "Копійки мають бути в межах 0..99.\n";

continue;

}

break;

}

return Money(grn, kop);

}

Money& Money::operator++() {

++first;

++second;

return \*this;

}

Money Money::operator++(int) {

Money tmp = \*this;

++(\*this);

return tmp;

}

Money& Money::operator--() {

--first;

--second;

return \*this;

}

Money Money::operator--(int) {

Money tmp = \*this;

--(\*this);

return tmp;

}

bool Money::operator!() const {

return second != 0;

}

Money Money::operator+(int scalar) const {

return Money(first, second + scalar);

}

explicit Money::operator string() const {

return to\_string(first) + " грн " + to\_string(second) + " коп";

}

void Money::print() const {

cout << first << " грн " << second << " коп" << endl;

}

3)main.cpp

#include "Money.h"

#include <iostream>

using namespace std;

int main() {

setlocale(0, "ukr");

cout << "Створюємо Money тільки через введення користувача!\n";

Money m1 = Money::input();

cout << "Ви ввели: ";

m1.print();

++m1;

cout << "Після ++: ";

m1.print();

m1--;

cout << "Після --: ";

m1.print();

cout << "Перевірка чи копійки не = 0: " << (!m1 ? "true" : "false") << "\n";

cout << "Додамо до копійок +25:\n";

Money m2 = m1 + 25;

m2.print();

string s = static\_cast<string>(m2);

cout << "Як рядок (через змінну): " << s << "\n";

}

РЕЗУЛЬТАТ ПРОГРАМИ