# Міністерство Освіти і Науки України

# Національний технічний університет

«Харківський політехнічний інститут»

Кафедра «Програмної інженерії та інформаційних технологій управління»

**Лабораторна робота № 1**

Створення та використання класів C++

**Виконав:**

Студент групи КН-36А

Кулик В.В.

**Перевірив:** Смолін П.А.

Харків 2017

**Завдання**

**1.1** Клас для представлення простого дробу

Створити клас для представлення простого дробу. Реалізувати конструктори, функцію скорочення дробу, а також перевантажити операції +, -, \*, /, введення та виведення. Здійснити демонстрацію можливостей класу в функції main().

**1.2** Клас для представлення двовимірного масиву

Розробити клас для представлення двовимірного масиву (матриці) цілих чисел довільних розмірів. Створити конструктори та деструктор, перевантажити операції додавання, віднімання і множення (згідно з правилами роботи з матрицями), звертання за індексом, введення з потоку та виведення в потік. Створити власні класи винятків та генерувати відповідні об'єкти-винятки, якщо неможливо виконати ту чи іншу операцію.

Створити окрему функцію, яка отримує посилання на матрицю і виконує над масивом дії, вказані в таблиці. Функція не повинна бути методом класу або дружньою функцією



У функції main() здійснити тестування всіх можливостей класу з перехопленням можливих винятків, а також розв'язати індивідуальну задачу.

**1.3** Підрахунок суми введених значень

Створити клас з одним закритим елементом даних цілого типу, геттером і конструктором з одним параметром. В цьому ж класі створити закрите статичне поле, яке зберігає суму цілих елементів даних всіх раніше створених об'єктів. Під час кожного виклику конструктора до статичного поля повинно додаватися нове значення. Статична публічна функція цього ж класу повинна повертати цю суму.

У функції main() створити декілька об'єктів і вивести отриману суму.

**Хід роботи**

**1.1** Створив клас для представлення простого дробу. Реалізував конструктори, функцію скорочення дробу, а також реалізував перевантаження операції +, -, \*, /, введення та виведення. Здійснив демонстрацію можливостей класу в функції main().

*Код програми 1.1:*

#include <iostream>

using std::cin;

using std::cout;

using std::endl;

using std::istream;

using std::ostream;

class Fraction {

friend istream& operator >> (istream& in, Fraction& fr) { return in >>fr.numerator >> fr.denominator; }

friend ostream& operator<<(ostream& out, const Fraction& fr) { return out << "numerator=" << fr.numerator << " denominator=" << fr.denominator; }

friend Fraction operator\*(Fraction fr1, Fraction fr2) { return Fraction(fr1.numerator \* fr2.numerator, fr1.denominator \* fr2.denominator); }

friend Fraction operator\* (Fraction fr, int val) { return Fraction(fr.numerator \* val, fr.denominator) ; }

friend Fraction operator/(Fraction fr1, Fraction fr2) { return Fraction(fr1.numerator \* fr2.denominator, fr1.denominator \* fr2.numerator); }

friend Fraction operator/ (Fraction fr, int val) { return Fraction(fr.numerator, fr.denominator \* val); }

friend Fraction operator+(Fraction fr1, Fraction fr2)

{

int new\_den = fr1.denominator\*fr2.denominator;

int new\_num = (new\_den / fr1.denominator\*fr1.numerator) + (new\_den / fr2.denominator\*fr2.numerator);

return Fraction(new\_num, new\_den);

}

friend Fraction operator-(Fraction fr1, Fraction fr2)

{

int new\_den = fr1.denominator\*fr2.denominator;

int new\_num = (new\_den / fr1.denominator\*fr1.numerator) - (new\_den / fr2.denominator\*fr2.numerator);

return Fraction(new\_num, new\_den);

}

private:

int numerator;

int denominator;

public:

Fraction() { numerator = denominator = 0; }

Fraction(int numerator, int denominator);

void Socr(int a, int b);

double getX() { return numerator; }

void setX(int numerator) { this->numerator = numerator; }

double getY() { return denominator; }

void setY(int denominator) { this->denominator = denominator; }

};

Fraction::Fraction(int numerator, int denominator)

{

if (denominator == 0)

throw std::runtime\_error("zero division error");

Socr(numerator, denominator);

}

void Fraction::Socr(int a, int b)

{

int x = (a < b) ? a : b;

for (int i = 2; i <= x; i++)

if (a%i == 0 && b%i == 0)

{

a /= i;

b /= i;

}

numerator = a;

denominator = b;

}

void main()

{

Fraction fr1, fr2;

cout << "Input first fraction: ";

cin >> fr1;

cout << "Input second fraction: ";

cin >> fr2;

cout << fr1 \* fr2 << endl;

cout << fr1 \* 4 << endl;

cout << fr1 / fr2 << endl;

cout << fr1 / 4 << endl;

cout << fr1 + fr2 << endl;

cout << fr1 - fr2 << endl;

}

**1.2** Індивідуальне завдання

*Код програми 1.2:*

#include <iostream>

using std::cout;

using std::endl;

using std::cin;

using std::istream;

using std::ostream;

class IntArray

{

    friend ostream& operator<<(ostream& out, const IntArray& a)

    {

        for (int i = 0; i < a.m; i++)

        {

            for (int j = 0; j < a.n; j++)

            {

                out << a.pa[i][j] << ' ';

            }

            cout << "\n";

        }

        return out;

    }

    friend istream& operator >> (istream& in, IntArray& a)

    {

        for (int i = 0; i < a.m; i++)

        {

            for (int j = 0; j < a.n; j++)

            {

                in >> a.pa[i][j];

            }

        }

        return in;

    }

    friend int operator\*(IntArray& v1, IntArray& v2)

    {

        if (v1.n == v2.m)

        {

            IntArray a = IntArray(v1.m, v2.n);

            for (int i = 0; i < v1.m; i++)

            {

                for (int j = 0; j < v2.n; j++)

                {

                    int sum = 0;

                    for (int k = 0; k < v1.n; k++)

                    {

                        sum += v1.pa[i][k] \* v2.pa[k][j];

                    }

                    a.pa[i][j] = sum;

                }

            }

            cout << a;

            return 0;

        }

        else

        {

            return 1;

        }

    }

    friend int operator+(IntArray& v1, IntArray& v2)

    {

        if (v1.m == v2.m && v1.n == v2.n)

        {

            IntArray a = IntArray(v1.m, v1.n);

            for (int i = 0; i < v1.m; i++)

            {

                for (int j = 0; j < v1.n; j++)

                {

                    a.pa[i][j] = v1.pa[i][j] + v2.pa[i][j];

                }

            }

            cout << a;

            return 0;

        }

        else

        {

            return 1;

        }

    }

    friend int operator-(IntArray& v1, IntArray& v2)

    {

        if (v1.m == v2.m && v1.n == v2.n)

        {

            IntArray a = IntArray(v1.m, v1.n);

            for (int i = 0; i < v1.m; i++)

            {

                for (int j = 0; j < v1.n; j++)

                {

                    a.pa[i][j] = v1.pa[i][j] - v2.pa[i][j];

                }

            }

            cout << a;

            return 0;

        }

        else

        {

            return 1;

        }

    }

private:

    int \*\*pa;

    int m;

    int n;

public:

    class OutOfBounds

    {

        int index;

    public:

        OutOfBounds(int i) : index(i) { }

        int getIndex() const { return index; }

    };

    IntArray() { pa = 0; m = 0; n = 0; }

    IntArray(int j, int k);

    IntArray(IntArray& arr);

    ~IntArray() { if (pa) delete[] pa; };

    int\* operator[](int index1);

    int getSize() const { return n\*m; }

    int getM() const { return m; }

    int getN() const { return n; }

};

IntArray::IntArray(int j, int k)

{

    pa = new int\*[m = j];

    for (int i = 0; i < m; i++)

    {

        pa[i] = new int[n = k];

    }

}

IntArray::IntArray(IntArray& arr)

{

    m = arr.m;

    n = arr.n;

    pa = new int\*[m];

    for (int i = 0; i < m; i++)

    {

        pa[i] = new int[n];

    }

    for (int i = 0; i < m; i++)

    {

        for (int j = 0; j < n; j++)

        {

            pa[i][j] = arr.pa[i][j];

        }

    }

}

int\* IntArray::operator[](int index1)

{

    return pa[index1];

}

int fun(IntArray& f)

{

    for (int i = 0; i < f.getM(); i++)

    {

        for (int j = 0; j < f.getN(); j++)

        {

            if (f[i][j] < 0)

            {

                f[i][j] = f[i][j] \* f[i][j];

            }

        }

    }

    cout << f << endl;

    return 0;

}

void main()

{

    IntArray a(4, 4);

    IntArray b(4, 4);

    cin >> a;

    cin >> b;

    cout << "/////////////////" << endl;

    cout << a << endl;

    cout << "/////////////////" << endl;

    cout << b[1][1] << endl;

    cout << "////////////////////////////" << endl;

    cout << "a\*b: " << a\*b << endl;

    cout << "/////////////////" << endl;

    cout << "a+b: " << a + b << endl;

    cout << "/////////////////" << endl;

    cout << "a-b: " << a - b << endl;

    cout << "/////////////////" << endl;

    cout << "KV:" << fun(a) << endl;

    cout << "/////////////////" << endl;

    system("pause");

}

**1.3** Написав програму яка підраховує суму введених значень.

*Код програми 1.3:*

#include <iostream>

using std::cout;

using std::endl;

class ObjectSum

{

private:

int x;

static int sum;

public:

static int getSum()

{

return sum;

}

ObjectSum(int y)

{

x = y;

sum += x;

}

ObjectSum()

{

x = 0;

sum = 0;

}

~ObjectSum()

{

sum--;

}

};

int ObjectSum::sum = 0;

void main()

{

ObjectSum c1;

cout << c1.getSum() << endl;

ObjectSum \*p1 = &c1;

cout << p1->getSum() << endl;

ObjectSum \*p2 = new ObjectSum(4);

cout << p2->getSum() << endl;

delete p2;

cout << p2->getSum() << endl;

system("pause");

}

***Висновок:***

*У даній роботі я за допомогою мови програмування С++ та використовуючи попередній досвід я розробив різного роду програми, під час розробки яких я навчився створювати та використовувати класи у C++.*