Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования
«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ имени Н.Э.БАУМАНА
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

Факультет: Информатика и системы управления

Кафедра: Теоретическая информатика и компьютерные технологии

Лабораторная работа №10

Реализация итераторов на языке C++

Вариант №2

Выполнил
студент группы ИУ9-21Б
Лисов Алексей

Москва, 2023

1 Условие

В ходе лабораторной работы нужно разработать программу, перегружающую итератор. Целочисленная матрица размера m × n с перегруженной операцией «[ ]», обеспечивающей обращение к элементам матрицы, и однонаправленным итератором по наибольшим общим делителям элементов каждой строки. При изменении наибольшего общего делителя каждый элемент строки делится на старый делитель, а затем умножается на новый делитель. Условие задачи, исходный код и пример работы программы необходимо прислать в формате LaTeX.

2 Код решения

Файл main.cpp

```
#include <iostream>
#include <numeric>
#include <vector>
#include <assert.h>

using namespace std;

class GCDIterator {
public:
    GCDIterator(int* row, int n) : row_(row), n_(n), gcd_(0) {
        update_gcd();
    }

    GCDIterator& operator++() {
        row_ += n_;
        update_gcd();
        return *this;
    }

    int operator*() const {
        return gcd_;
    }

    bool operator==(const GCDIterator& other) const {
        return row_ == other.row_;
    }
```

```
    bool operator!=(const GCDIterator& other) const {
        return !(*this == other);
    }

private:
    void update_gcd() {
        gcd_ = abs(row_[0]);
        for (int i = 1; i < n_; i++) {
            gcd_ = gcd(gcd_, abs(row_[i]));
        }
    }

    int* row_;
    int n_;
    int gcd_;
};

class Matrix {
public:
    Matrix(int m, int n) : data_(m * n), m_(m), n_(n) {}

    int* operator[](int i) {
        return &data_[i * n_];
    }

    GCDIterator begin() {
        return GCDIterator(&data_[0], n_);
    }

    GCDIterator end() {
        return GCDIterator(&data_[m_ * n_], n_);
    }

private:
    vector<int> data_;
    int m_, n_;
};

int main() {
```

```
    Matrix mat(3, 4);
    mat[0][0] = 6; mat[0][1] = 9; mat[0][2] = 21; mat[0][3] = 33;
    mat[1][0] = 15; mat[1][1] = 18; mat[1][2] = 27; mat[1][3] = 30;
    mat[2][0] = 8; mat[2][1] = 12; mat[2][2] = 24; mat[2][3] = 36;

    vector<int> a;
    for (auto it = mat.begin(); it != mat.end(); ++it) {
        int row_gcd = *it;
        //cout << "Row gcd: " << row_gcd << '\n';
        a.push_back(row_gcd);
    }

    vector<int> ans = {3,3,4};
    assert(a == ans);

    mat[0][0] = 5;
    mat[1][0] = 21;
    //cout << "After changes:\n";
    while (a.size()) {
        a.pop_back();
    }
    for (auto it = mat.begin(); it != mat.end(); ++it) {
        int row_gcd = *it;
        //cout << "Row gcd: " << row_gcd << '\n';
        a.push_back(row_gcd);
    }

    ans = {1,3,4};
    assert(a == ans);

    return 0;
}
```

3 Скриншоты

```cpp
#include <iostream>
#include <numeric>
#include <vector>
#include <assert.h>

using namespace std;

class GCDIterator {
public:
    GCDIterator(int* row, int n) : row_(row), n_(n), gcd_(0) {
        update_gcd();
    }

    GCDIterator& operator++() {
        row_ += n_;
        update_gcd();
        return *this;
    }

    int operator*() const {
        return gcd_;
    }
```

Рис. 1: Вывод программы

```
        a.push_back(row_gcd);
    }

    vector<int> ans = {3,3,4};
    assert(a == ans);

    mat[0][0] = 5;
    mat[1][0] = 21;
    //cout << "After changes:\n";
    while (a.size()) {
        a.pop_back();
    }
    for (auto it : GCDIterator = mat.begin(); it != mat.end(); ++it) {
        int row_gcd = *it;
        //cout << "Row gcd: " << row_gcd << '\n';
        a.push_back(row_gcd);
    }

    ans = {1,3,4};
    assert(a == ans);

    return 0;
}
```

Рис. 2: Вывод программы

```
C:\Users\Aleksey\CLionProjects\untitled13\cmake-build-debug\untitled13.e

Process finished with exit code 0
```

Рис. 3: Вывод программы