Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«**Пермский национальный исследовательский политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЕТ**

Дисциплина: «Информатика»

"Классы и объекты. Атд.Контейнеры."

Семестр 2

Выполнил работу

Студент группы РИС-22-1Б

Бадртдинов Т.З

Проверил

Доцент кафедры ИТАС

Полякова О.А.

Г. Пермь-2023

# Постановка задачи

1. Определить класс-контейнер.

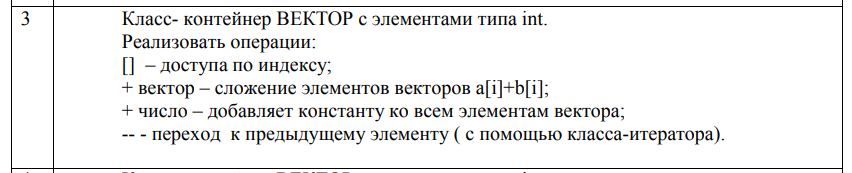
2. Реализовать конструкторы, деструктор, операции ввода-вывода, операцию присваивания.

3. Перегрузить операции, указанные в варианте.

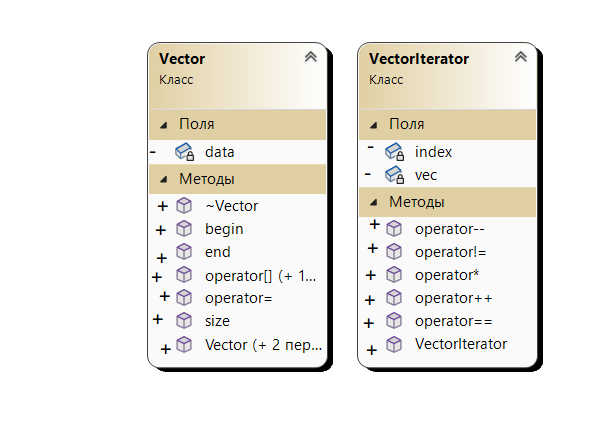
4. Реализовать класс-итератор. Реализовать с его помощью операции последовательного доступа.

5. Написать тестирующую программу, иллюстрирующую выполнение операций.

Вариант 3



Описание класса



# Код программы(vector.h)

#include <vector>

class VectorIterator;

class Vector {

public:

Vector();

explicit Vector(int size) : data(size) {}

Vector(const Vector& other) : data(other.data) {}

Vector& operator=(const Vector& other) {

data = other.data;

return \*this;

}

~Vector() {};

int& operator[](int index) {

return data[index];

}

const int& operator[](int index) const {

return data[index];

}

friend Vector operator+(const Vector& a, const Vector& b) {

Vector result(a.size());

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

result[i] = a[i] + b[i];

}

return result;

}

friend Vector operator+(const Vector& v, int x) {

Vector result(v.size());

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

result[i] = v[i] + x;

}

return result;

}

VectorIterator begin();

VectorIterator end();

int size() const {

return data.size();

}

private:

std::vector<int> data;

};

class VectorIterator {

public:

explicit VectorIterator(Vector& vec, int index) : vec(vec), index(index) {}

VectorIterator& operator++() {

++index;

return \*this;

}

VectorIterator& operator--() {

--index;

return \*this;

}

bool operator==(const VectorIterator& other) const {

return index == other.index;

}

bool operator!=(const VectorIterator& other) const {

return !(\*this == other);

}

int& operator\*() const {

return vec[index];

}

private:

Vector& vec;

int index;

};

VectorIterator Vector::begin() {

return VectorIterator(\*this, 0);

}

VectorIterator Vector::end() {

return VectorIterator(\*this, size());

}

std::ostream& operator<<(std::ostream& os, const Vector& v) {

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

os << v[i] << ' ';

}

return os;

# }

# Код программы(main)

#include <iostream>

#include "vector.h"

int main() {

Vector v1(5);

Vector v2(5);

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

v1[i] = i;

v2[i] = i \* i;

}

std::cout << "v1 = " << v1 << std::endl;

std::cout << "v2 = " << v2 << std::endl;

Vector v3 = v1 + v2;

std::cout << "v3 = " << v3 << std::endl;

Vector v4 = v1 + 10;

std::cout << "v4 = " << v4 << std::endl;

int a;

VectorIterator it = v1.begin();

a = \*it;

std::cout << "v1 before increment: " << a << std::endl;

++it;

a = \*it;

std::cout << "v1 after increment: " << a << std::endl;

--it;

a = \*it;

std::cout << "v1 after decrement: " << a << std::endl;

system("pause");

return 0;

}

# Работы программы

