Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

**по лабораторной работе №18.6**

Дисциплина: «информатика»

Тема: АТД. Контейнеры

Вариант 5

Выполнил работу

студент группы РИС-20-1б

Зверев А.Д.

Проверила

Доцент кафедры ИТАС

Полякова О.А.

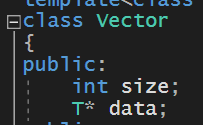
Пермь, 2021

**Постановка задачи**

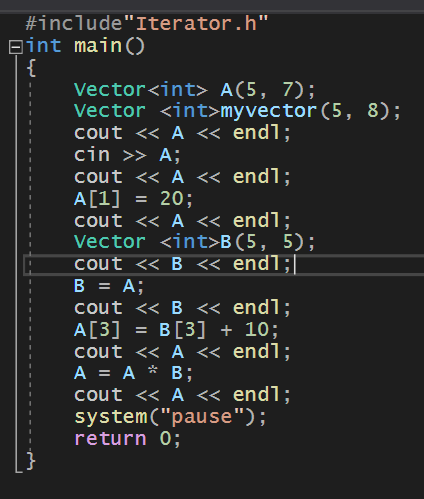
1. Определить класс—контейнер.
2. Реализовать конструкторы, деструктор, операции ввода—вывода, операцию присваивания.
3. Перегрузить операции, указанные в варианте.
4. Реализовать класс—итератор. Реализовать его с помощью операции последовательного доступа.
5. Написать тестирующую программу, иллюстрирующую выполнение операций.
6. Класс—контейнер ВЕКТОР с элементами типа int.
7. Реализовать операции:
8. []—доступа по индексу;
9. int()—определение размера вектора;
10. \* вектор—умножение элементов векторов a[i]\*b[i];
11. + n—переход вправо к элементу с номером n (с помощью класса—итератора).

**Анализ задачи**

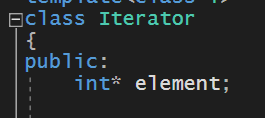
1. Для решения задачи необходимо…
   1. Создать класс Vector, где будут храниться данные, используемые внутри программы;



* 1. Написать функцию int main, где будут вызываться основные операции, производимые с векторами;



* 1. В заголовочном файле Iterator.h описать класс с нужными переменными;



1. В программе были использованы следующие типы данных:
   1. Тип данных int для хранения введенных данных;



**Блок—схема программы**



**Решение**

**Source.cpp**

#include"Iterator.h"

int main()

{

Vector<int> A(5, 7);

Vector <int>myvector(5, 8);

cout << A << endl;

cin >> A;

cout << A << endl;

A[1] = 20;

cout << A << endl;

Vector <int>B(5, 5);

cout << B << endl;

B = A;

cout << B << endl;

A[3] = B[3] + 10;

cout << A << endl;

A = A \* B;

cout << A << endl;

system("pause");

return 0;

}

**Vector.h**

#pragma once

#include<iostream>

using namespace std;

template<class T>

class Vector

{

public:

int size;

T\* data;

public:

Vector() :size(0), data(NULL) {}

Vector(int s, T element = 0)

{

system("chcp 1251>nul");

cout << "введите количеcтво элементов в контейнере: ";

cin >> size;

size = s;

data = new T[size];

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

{

data[i] = element;

}

}

Vector(const Vector& ob)

{

size = ob.size;

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

{

data[i] = ob.data[i];

}

}

Vector& operator=(const Vector& ob)

{

if (this == &ob)return \*this;

size = ob.size;

if (data != 0)

delete[]data;

data = new int[size];

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

{

data[i] = ob.data[i];

}

return \*this;

}

int& operator[](int index)

{

if (index < size)return data[index];

else

{

cout << "index > size " << endl;

return \*data;

}

}

Vector& operator+(int element) //\*

{

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

{

this->data = this->data[i] + &element;

}

return \*this;

}

int operator()()

{

return size;

}

Vector& operator\*(const Vector& ob)

{

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

ob.data[i] \*= ob.data[i];

return \*this;

}

friend istream& operator>>(istream& in, const Vector& ob)

{

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

in >> ob.data[i];

return in;

}

friend ostream& operator<<(ostream& out, const Vector& ob)

{

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

out << ob.data[i] << " ";

return out;

}

~Vector() { delete[]data; data = NULL; }

};

**Iterator.h**

#include"Vector.h"

template<class T>

class Iterator

{

public:

int\* element;

public:

Iterator() :element(NULL) {}

Iterator(Vector<T>& ob, int index)

{

element = ob.data[index];

}

Iterator& operator+(Vector<T>& ob)

{

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

{

this->element = element + ob.data[i];

}

return \*this;

}

Iterator& operator+(int intiger)

{

return \*this + intiger;

}

};

**Скриншоты**

