Пермский Национальный Исследовательский  
Политехнический Университет

**Лабораторная работа № 6**

Информатика

за 2 семестр

Вариант № 1

Выполнил:

Студент группы РИС 20-1-бз

Акиев В.В.

20-ЭТФ-641

Проверила:

Доцент кафедры ИТАС

Полякова О.А.

2021

**АТД. Контейнеры**

1. **Цель задания**.
   1. Создание консольного приложения, состоящего из нескольких файлов в системе программированияVisual Studio.
   2. Реализация класса-контейнера.
2. **Задание.**

Класс-контейнер ВЕКТОР с элементами тип аint. Реализовать операции:

[] – доступа по индексу;

() – определение размера вектора;

+ число–добавляет константу ко всем элементам вектора;

++- переход к следующему элементу (с помощью класса-итератора).

1. **Файл Лабораторная\_6.cpp**

#include "Vector.h"

#include <iostream>

using namespace std;

void main()

{

Vector a(5);

cout << a << "\n";

cin >> a;

cout << a << "\n";

a[2] = 100;

cout << a << "\n";

Vector b(10);

cout << b << "\n";

b = a;

cout << b << "\n";

Vector c(10);

c = b + 100;

cout << c << "\n";

cout << "\nthe length of a=" << a() << endl;

cout << \*(a.first())<<endl;

Iterator i = a.first();

++i;

cout << \*i << endl;

for(i=a.first(); i!=a.last(); ++i) cout << \*i << endl;

for (i = a.first(); i != a.last(); i++) cout << \*i << endl;

}

1. **ФайлVector.cpp**

#include "Vector.h"

Vector::Vector(int s, int k)

{

size = s;

data = new int[size];

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

data[i] = k;

beg.elem = &data[0];

end.elem = &data[size];

}

Vector::Vector(const Vector& a)

{

size = a.size;

data = new int[size];

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

data[i] = a.data[i];

beg = a.beg;

end = a.end;

}

Vector::~Vector()

{

delete[] data;

data = 0;

}

int& Vector::operator[](int index)

{

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

else cout << "\nError! Index>size";

}

Vector Vector::operator+(const int k)

{

Vector temp(size);

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

temp.data[i] += data[i] + k;

return temp;

}

int Vector::operator()()

{

return size;

}

Vector& Vector::operator=(const Vector& a)

{

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

size = a.size;

if (data != 0) delete[]data;

data = new int[size];

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

data[i] = a.data[i];

beg = a.beg;

end = a.end;

return \*this;

}

istream& operator>>(istream& in, Vector& a)

{

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

in >> a.data[i];

return in;

}

ostream& operator<<(ostream& out, const Vector& a)

{

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

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

return out;

}

1. **ФайлVector.h**

#pragma once

#include <iostream>

using namespace std;

class Iterator

{

friend class Vector;

public:

Iterator() { elem = 0; }

Iterator(const Iterator& it) { elem = it.elem; }

bool operator == (const Iterator& it) { return elem == it.elem; }

bool operator != (const Iterator& it) { return elem != it.elem; }

void operator++ () { ++elem; }

void operator ++(int) { elem++; }

void operator --() { --elem; }

int& operator \*() const { return \*elem; }

private:

int\* elem;

};

class Vector

{

public:

//конструкторы с параметрами

//s - длина, k - значение каждого элемента

Vector(int s, int k = 0);

//клонирование

Vector(const Vector& a);

//деструктор

~Vector();

//перегрузка присваивания

Vector& operator= (const Vector& a);

//перегрузка обращения по индексу

int& operator[] (int index);

//перегрузка добавления

Vector operator+(const int k);

//возвращение длины вектора

int operator()();

Iterator first() { return beg; };

Iterator last() { return end; }

//перегрузка ввода вывода

friend istream& operator>>(istream& in, Vector& a);

friend ostream& operator<<(ostream& out, const Vector&a);

private:

int size;

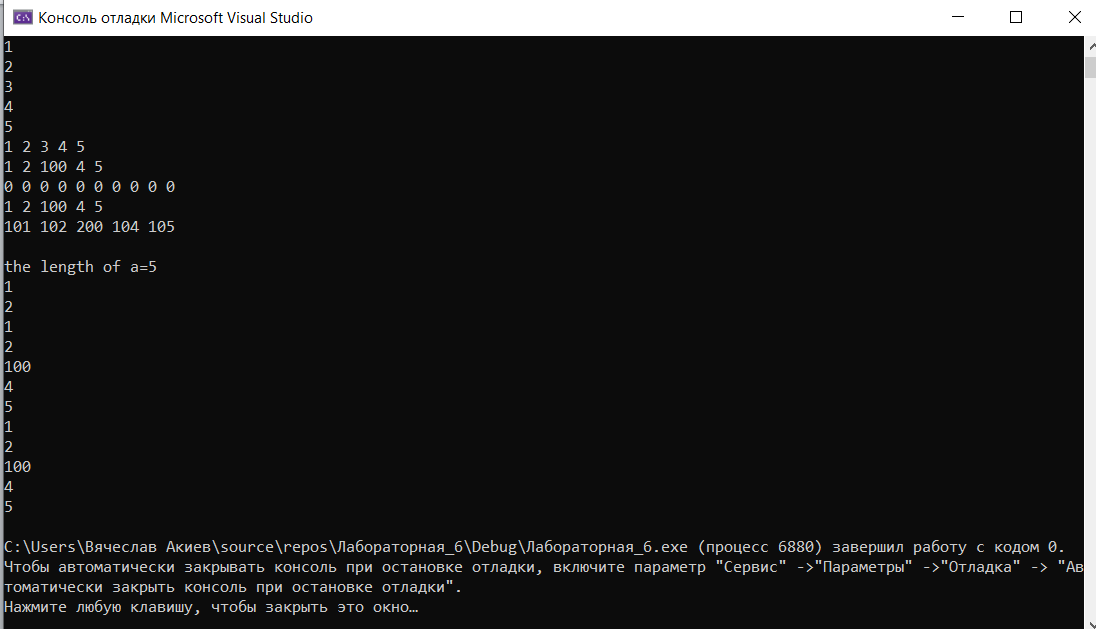
int\* data;

Iterator beg;

Iterator end;

};

1. **Результат работы программы**

****

1. **Контрольные вопросы.** 
   1. **Что такое абстрактный тип данных? Привести примерыАТД.**

Тип данных, определяемы только через операции, которые могут выполняться над соответствующими объектами. апример атд можно считать использование функции для разных классов которые предком которых является 1 и тот же абстрактный класс.

* 1. **Привести примеры абстракции через параметризацию.**

Использование 1 и той-же функции для разных переменных

* 1. **Привести примеры абстракции через спецификацию.**

Использование 1 и той-же функции для разных классов из 1

* 1. **Что такое контейнер? Привести примеры.**

Контейнер - набор однотипных элементов. Частный случай массив.

* 1. **Какие группы операций выделяют в контейнерах?**

Операции доступа к элементам, которые обеспечивают и операцию замены значений элементов

Операции добавления удаления

Операция поиска

Операция объединения контейнеров

Специальные операции, зависящие от вида контейнера

* 1. **Какие виды доступа к элементам контейнера существуют? Привести примеры.**

Прямой доступ - доступ по индексу (a[10] - 10 элемент массива a)

Ассоциативный доступ - доступ по идентификатору который выступает в роли индекса (a["abc"] передаёт элемент с именем abc )

Последовательный доступ - от элемента к элементу при помощи различный функций (a.first() - передаёт первый элемент)

* 1. **Что такое итератор?**

Итератор – объект, который обеспечивает последовательный доступ к элементам контейнера.

* 1. **Каким образом может быть реализован итератор?**

Итератор можно реализовать как класс, представляющий такой же набор операций. Итератор реализуется как класс, который имеет такой же интерфейс.

* 1. **Каким образом можно организовать объединение контейнеров?**

Можно реализовать как объединения множеств в новый контейнер попадают только те элементы которые есть хотя бы в 1 контейнере.

* 1. **Какой доступ к элементам предоставляет контейнер, состоящий из элементов**

**«ключ-значение»?**

Предоставляет доступ к значению элемента по ключу (ассоциативный контейнер)

* 1. **Как называется контейнер, в котором вставка и удаление элементов выполняется на одном конце контейнера?**

Lastinfirstout

* 1. **Какой из объектов (a,b,c,d) является контейнером?**

D

* 1. **Какой из объектов (a,b,c,d)не является контейнером?**

C

* 1. **Контейнер реализован как динамический массив, в нем определена операция доступ по индексу. Каким будет доступ к элементам контейнера?**

Прямой доступ

* 1. **Контейнер реализован как линейный список. Каким будет доступ к элементам контейнера?**

Последовательный доступ