Министерство образования РФ

Пермский национальный исследовательский политехнический университет

Кафедра ИТАС

ЛАБОРАТОРНАЯ РАБОТА № 7

ПО ИНФОРМАТИКЕ ЗА II СЕМЕСТР

Вариант 15

Выполнил студент:

Сташков Арсений Дмитриевич

Группа РИС-20-1бз

Шифр 20-ЭТФ-644

Проверила:

Доцент кафедры ИТАС

Полякова Ольга Андреевна

ПЕРМЬ 2022

# Лабораторная работа №7 "Шаблоны классов"

**Вариант №15**

**Цель:** 1) Создание консольного приложения, состоящего из нескольких файлов в системе программирования Visual Studio.

2) Реализация шаблона класса-контейнера.

**Задача:** 1. Определить шаблон класса-контейнера (см. лабораторную работу №6).

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

присваивания.

3. Перегрузить операции, указанные в варианте.

4. Инстанцировать шаблон для стандартных типов данных (шф Ноа, доче).

5. Написать тестирующую программу, иллюстрирующую выполнение операций для контейнера, содержащего элементы стандартных типов данных.

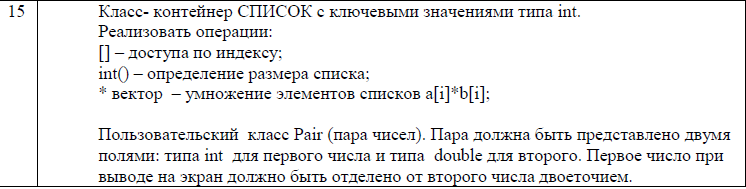
6. Реализовать пользовательский класс (см. лабораторную работу №3).

7. Перегрузить для пользовательского класса операции ввода-вывода.

8. Перегрузить операции необходимые для выполнения операций контейнерного класса.

9. Инстанцировать шаблон для пользовательского класса.

10. Написать тестирующую программу, иллюстрирующую выполнение операций для контейнера, содержащего элементы пользовательского класса.



**Решение задания:**

Файл Pair.h:

#pragma once

#include <iostream>

using namespace std;

struct Pair {

Pair(void);

Pair(int, double);

Pair(const Pair&);

Pair& operator = (const Pair&);

friend ostream& operator << (ostream& out, const Pair&);

friend istream& operator >> (istream in, Pair&);

virtual ~Pair(void) {};

int first;

double second;

};

Файл Pair.cpp:

#include "Pair.h"

#include <iostream>

using namespace std;

Pair::Pair(void) { first = second = 0; }

Pair::Pair(int F, double S) { first = F, second = S; }

Pair::Pair(const Pair& t) { first = t.first; second = t.second; }

Pair& Pair::operator = (const Pair& t)

{ first = t.first; second = t.second; return \*this; }

ostream& operator << (ostream& out, const Pair& t)

{ out << t.first << " : " << t.second; return out; }

istream& operator >> (istream& in, Pair& t) {

cout << "\nПервое число? ";

in >> t.first;

cout << "\nВторое число? ";

in >> t.second;

return in;

}

Файл Listing.cpp

#pragma once

#include "Pair.h"

#include <iostream>

using namespace std;

// T - параметр шаблона

template <class T>

struct Listing {

// Конструктор с параметрами: выделяет память под s элементов и заполняет их значением k

Listing(int s, T k);

// Конструктор с параметрами

Listing(const Listing<T>& a);

// Деструктор

~Listing();

// Присваивание

Listing& operator = (const Listing<T>& a);

// Доступ по индексу

T& operator[] (int index);

// Извлечение длины списка

int operator () ();

// Перегруженные операции ввода - вывода

friend ostream& operator << <> (ostream& out, const Listing<T>& a);

friend istream& operator >> <> (istream& in, Listing<T>& a);

private:

int size;

T\* data;

};

template <class T>

Listing<T>::Listing(int s, T k) {

size = s;

data = new T[size];

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

data[i] = k;

}

template <class T>

Listing<T>::Listing(const Listing& a) {

size = a.size;

data = new T[size];

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

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

}

template <class T>

Listing<T>::~Listing() {

delete[] data;

data = 0;

}

template <class T>

Listing<T>& Listing<T>::operator = (const Listing<T>& a) {

if (this == &a)

return \*this;

size = a.size;

if (data != 0)

delete[] data;

data = new T[size];

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

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

return \*this;

}

template <class T>

T& Listing<T>::operator[] (int index) {

if (index < size)

return data[index];

else

cout << "\nОшибка! Индекс больше, чем длина списка";

}

// Переопределение оператора вызова функции

template <class T>

int Listing<T>::operator() () { return size; }

template <class T>

ostream& operator << (ostream& out, const Listing<T>& a) {

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

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

return out;

}

template <class T>

istream& operator >> (istream& in, Listing<T>& a) {

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

in >> a.data[i];

return in;

}

Файл Шаблоны классов.cpp

#include "Listing.h"

#include "Pair.h"

#include <iostream>

#include <Windows.h>

using namespace std;

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

cout << "Введите 5 целых чисел: \n";

Listing<int>A(5, 0);

cin >> A;

cout << "\nИнициализация, ввод и вывод значений: " << A << endl;

Listing <int>B(10, 5);

cout << "\nИнициализация и вывод заранее заданных значений: " << B << endl;

B = A;

cout << "\nПрисваивание значение A переменной B: " << B << endl;

cout << "\nДоступ по индексу [2]: " << A[2] << endl;

cout << "\nРазмер списка " << A() << " символов" << endl;

cout << "\nУмножение элементов списков по индексам [0] и [2]: " << A[0] \* A[2];

}

**Результат:**

Введите 5 целых чисел:

1

2

3

4

5

Инициализация, ввод и вывод значений: 1 2 3 4 5

Инициализация и вывод заранее заданных значений: 5 5 5 5 5 5 5 5 5 5

Присваивание значение A переменной B: 1 2 3 4 5

Доступ по индексу [2]: 3

Размер списка 5 символов

Умножение элементов списков по индексам [0] и [2]: 3

