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 №18.7**

Дисциплина: «Основы теории алгоритмов и структуры данных»  
Тема: Шаблоны классов

Вариант 14

Выполнил:

студент группы РИС-20-2б

Вичугов Алексей Дмитриевич

Проверила:

доцент кафедры ИТАС

Полякова Ольга Андреевна

Пермь, 2021

**Цель работы**

1. Создание консольного приложения, состоящего из нескольких файлов.

2. Реализация шаблона класса-контейнера.

**Постановка задачи**

1. Определить шаблон класса-контейнера.

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

3. Перегрузить необходимые операции.

4. Инстанцировать шаблон для стандартных типов данных.

5. Реализовать пользовательский класс .

6. Перегрузить для пользовательского класса операции ввода-вывода.

7. Перегрузить операции, необходимые для выполнения операций контейнерного класса.

8. Инстанцировать шаблон для пользовательского класса.

9. Написать тестирующую программу, демонстрирующую выполнение указанных заданий.

10. Задание варианта: Класс-контейнер- спрсок с ключевыми значениями типа int. Реализовать операции доступа по индексу, определения размера списка и умножения всех элементов списка на число. Пользовательский класс – пара. Пара должна быт представлена двумя полями: типа int для первого числа и типа double для второго числа. Первое число при выводе на экран должно быть отделено от второго числа двоеточием.

**Анализ задачи**

1. Описание класса:

struct I{

int val;

I \*sled;

};

struct D{

double val;

D \*sled;

};

struct C{

char val;

C \*sled;

};

class Para{

public:

double val;

int cel;

double drob;

Para \*sled;

void Complete(){cel=val;drob=val-cel;Show();}

void Show(){cout << cel << ":" << drob << " ";}

};

template <class T> class Lister{

public:

T \*first;

int len;

Lister(){len=0;first=NULL;}

Lister(T \*f, int l){first=f;len=l;}

Lister(Lister &l){first=l.first;len=l.len;}

~Lister(){}

double operator [](int index){

bool b=0;

T \*cur=first;

if(len>0){

for (int i=0;i<index;i++){

if (cur->sled!=NULL){

cur=cur->sled;

}

else b=1;

}

}

else b=1;

if(b) return 0;

else return cur->val;

}

int operator ()(){

return len;

}

void operator \*(int n){

if(len>0){

T \*cur=first;

while (cur->sled!=NULL){

cur->val\*=n;

cur=cur->sled;

}

cur->val\*=n;

}

}

void Create(int r){

if (r>0){

len=1;

first = new T;

T \*cur, \*nov;

cur=first;

cur->sled=NULL;

cout << "1: ";

cin >> cur->val;

for (int i=1;i<r;i++){

len++;

nov = new T;

nov->sled=NULL;

cur->sled=nov;

cur=nov;

cout << i+1 << ": ";

cin >> cur->val;

}

}

}

void Show(){

T \*c=first;

while (c->sled!=NULL){

cout << c->val << " ";

c=c->sled;

}

cout << c->val << endl;

}

void PShow(){

T \*c=first;

while (c->sled!=NULL){

c->Complete();

c=c->sled;

}

c->Complete();

}

};

4. Функция main():

main(){

Lister <I> Ilist;

Lister <D> Dlist;

Lister <C> Clist;

Lister <Para> Plist;

Ilist.Create(5);

Dlist.Create(4);

Clist.Create(3);

Plist.Create(2);

cout << Ilist[4] << " " << Dlist[3] << " " << Clist[2] << " " << Plist[1] << endl;

Ilist();

Dlist();

Clist();

Plist();

Ilist\*3;

Dlist\*2;

Plist\*5;

Ilist.Show();

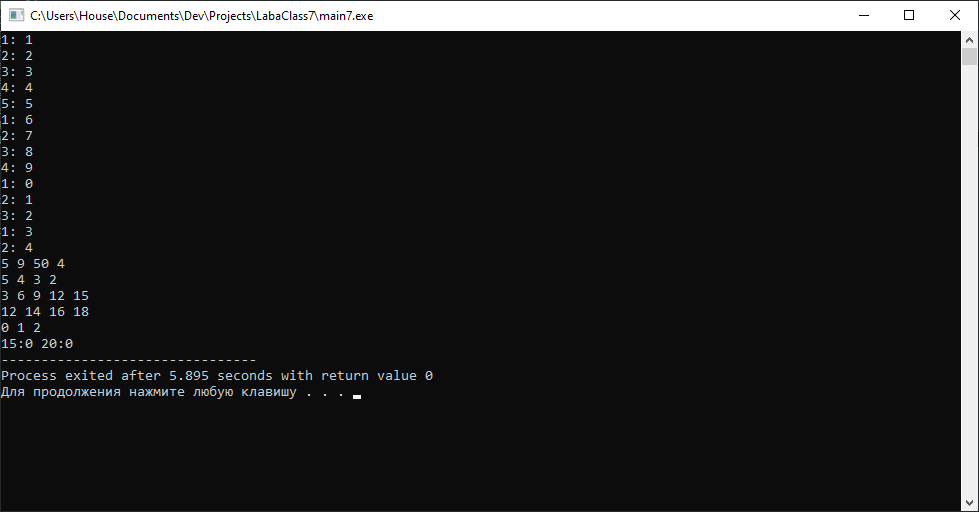
Dlist.Show();

Clist.Show();

Plist.PShow();

}

**Результаты работы программы**

****

**Ответы на вопросы**

1. Шаблоны используются, чтобы не создавать множество однотипных функций или классов для разных типов данных. Шаблоны позволяют сэкономить время написания кода и уменьшить его размер.
2. template <параметры\_шаблона> тип\_функции имя\_функции {}
3. template <параметры\_шаблона> class имя\_класса {};
4. Это параметры неопределённого типа, которыми оперирует функция.
5. При вызове функции компилятор определяет тип значения, переданного в функцию в качестве аргумента и устанавливает тип данных параметру шаблона функции. Габлоны функций не могут быть виртуальными.
6. Template <class T> (T – имя параметризуемого типа).
7. Можно.
8. Шаблоны классов могут содержать статические элементы, дружественные функции и классы. Шаблоны могут быть производными как от шаблонов, так и от обычных классов, а также являться базовыми и для шаблонов и для обычных классов.
9. Нет.
10. Не все.
11. Да.
12. С использованием синтаксиса объявления шаблона перед функцией.
13. Процесс генерации компилятором определения конкретного класса по шаблону класса и аргументам шаблона.
14. При создании экземпляра класса с указанием типа данных в качестве параметра.