МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Тихоокеанский государственный университет»

Кафедра «Программное обеспечение вычислительной техники и автоматизированных систем»

Перегрузка операторов

Лабораторная работа №4

по дисциплине «Объекно-ориентированное программирование»

Выполнил студент Чекулаев В. Ю.

Факультет, группа ФКФН, ПО(аб)-81

Проверил Федосеев А. А.

Хабаровск – 2020г.

1. **Постановка задачи**

На основе лабораторной работы № 3, перегрузить операторы для улучшения работы с созданной структурой данных. Согласно варианта реализовать набор перегруженных операторов. Для обработки ошибочных ситуаций использовать механизм исключений.

Вариант 6 : правый верхний треугольник квадратной матрицы. Размеры указываются в конструкторе. Операторы:

* [] - чтение и запись элемента по индексу

1. **Содержание заголовочного файла класса-шаблона**

*#ifndef RIGHTTRIANGLE\_H*

#define RIGHTTRIANGLE\_H

#include<cstddef>

#include<iostream>

template<class T>

class RightTriangle{

private:

class SubClass{

public:

SubClass(T\* pointer, size\_t i, size\_t n);

~SubClass(){};

T& operator[](size\_t index) const;

private:

T\* m\_ptr;

size\_t m\_index;

size\_t m\_n;

};

public:

RightTriangle(size\_t n = 0);

~RightTriangle(){ if(m\_triangle){ delete[] m\_triangle; } }

T get\_at(size\_t i, size\_t j) const;

void set\_at(size\_t i, size\_t j,const T& elem);

RightTriangle<T>::SubClass& operator[](size\_t index) const;

private:

size\_t m\_n;

T\* m\_triangle;

};

template<class T>

RightTriangle<T>::RightTriangle(size\_t n) : m\_n(n), m\_triangle(nullptr){

try{

if(n <= 1) throw(1);

u\_int amount(0);

while(n){

amount += n;

--n;

}

m\_triangle = new T[amount]();

}

catch(int){

std::cout << "RightTriangle<T>::RightTriangle(int n)::Uncorrect size" << "\n";

exit(1);

}

catch(std::bad\_alloc){

std::cout << "RightTriangle<T>::RightTriangle(int n)::Failed to allocate memory" << "\n";

exit(1);

}

}

template<class T>

RightTriangle<T>::SubClass::SubClass(T\* pointer, size\_t i, size\_t n) : m\_ptr(pointer), m\_index(i), m\_n(n){}

template<class T>

T& RightTriangle<T>::SubClass::operator[](size\_t index) const{

try {

if(m\_index > index || (m\_index >= m\_n && index >= m\_n)) throw(1);

return m\_ptr[index];

}

catch (int) {

std::cout << "RightTriangle<T>::operator[]::Index out of range\n";

exit(1);

}

}

template<class T>

typename RightTriangle<T>::SubClass& RightTriangle<T>::operator[](size\_t index)const{

T\* ptr = m\_triangle;

size\_t i = index;

size\_t n(m\_n);

--n;

while(i){

ptr += n;

--n; --i;

}

SubClass\* obj = new SubClass(ptr, index, m\_n);

return \*obj;

}

#endif *//* *RIGHTTRIANGLE\_H*

1. **Список исключительных ситуаций при работе класса-шаблона:**

* Неверное задание размеров матрицы в конструкторе
* Невозможность выделения достаточного количества памяти для хранения матрицы
* Неверное задание индексов при чтении матрицы
* Неверное задание индексов при записи значения в матрицу

1. **Текст одного из методов класса-шаблона, реализующего поставленную задачу**

template<class T>

typename RightTriangle<T>::SubClass& RightTriangle<T>::operator[](size\_t index)const{

T\* ptr = m\_triangle;

size\_t i = index;

size\_t n(m\_n);

--n;

while(i){

ptr += n;

--n; --i;

}

SubClass\* obj = new SubClass(ptr, index, m\_n);

return \*obj;

}

**5. Пример работы класса‑шаблона для двух разных типов**

Содержание файла main.cpp:

#include<iostream>

#include<cstddef>

#include"righttriangle.h"

class TestClass{

public:

TestClass(int num = 0) : m\_num(num){}

~TestClass(){}

int get() const{

return m\_num;

}

private:

int m\_num;

};

int main(){

size\_t n = 5;

RightTriangle<double> obj1(n);

obj1[2][3] = 8;

obj1[0][2] = 2;

obj1[4][4] = 5;

obj1[0][0] = 3;

obj1[2][3] = 7;

std::cout << "obj1:" << "\n";

for(size\_t i = 0; i < n; ++i){

for(size\_t j = 0; j < n; ++j){

if(i <= j){

std::cout << obj1[i][j];

} else{

std::cout << " ";

}

}

std::cout << "\n";

}

n = 8;

RightTriangle<TestClass> obj2(n);

obj2[2][6] = TestClass(8);

obj2[0][0] = TestClass(4);

obj2[3][5] = TestClass(1);

obj2[6][6] = TestClass(7);

obj2[5][6] = TestClass(3);

obj2[2][3] = TestClass(2);

std::cout << "\nobj2:" << "\n";

for(size\_t i = 0; i < n; ++i){

for(size\_t j = 0; j < n; ++j){

if(i <= j){

std::cout << obj2[i][j].get();

} else{

std::cout << " ";

}

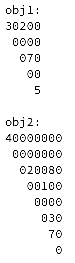
}

std::cout << "\n";

}

}

Вывод программы:



**6. Пример работы класса‑шаблона по обработке исключительных ситуаций**

Изменение в файле main.cpp:

Строка 24: «obj1[0][0] = 3;» изменено на «obj1[2][0] = 3;»

Вывод программы:

