МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

#### Факультет информационных технологий и компьютерной безопасности

(факультет)

### Кафедра Систем автоматизированного проектирования и информационных систем

# ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 9

по дисциплине Объектно-ориентированное программирование

Тема: «Регулярные выражения. Шаблоны»

Выполнили студенты группы ИСТ-222 Е.Г. Гладнева

Подпись, дата Инициалы, фамилия

Е.В. Журавлёва

Подпись, дата Инициалы, фамилия

## Принял Б.Н. Тишуков

Подпись, дата Инициалы, фамилия

Защищена \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

дата

2023

Лабораторная работа № 9

«Регулярные выражения. Шаблоны»

1 Цель работы

Целью лабораторной работы является изучение способов использования регулярных выражений и шаблонов в языке программирования C++.

1. Задание на лабораторную работу

Задание 1. Считать текст из файла и вывести его на экран монитора, заменив цифры словами, например, «0» на слово «ноль»; «1» на слово «один» и т. д.

Задание 2.1. Реализовать шаблонную версию класса Array. Добавить возможность работать с данными любого типа.

Задание 2.2. Реализовать класс Array не полагаясь на то, что для типаn T определен оператор присваивания

Задание 2.3. Реализовать шаблонную структуру ValueHolder с одним типовым параметром T, унаследованную от интерфейса ICloneable.

Задание 2.4. Реализовать функцию копирования элементов copy\_n из массива источника типа U\* в целевой массив типа T\*, где T и U произвольные типы, для которых определено преобразование из U в T.

Задание 2.5. Реализовать шаблонную функцию minimum, которая находит минимальный элемент, который хранится в экземпляре шаблонного класса Array, при этом типовой параметр шаблона Array может быть произвольным.

Задание 2.6. Необходимо написать шаблонную функцию flatten, которая принимает на вход "многомерный" Array неизвестной заранее глубины вложенности и выводит в поток out через пробел все элементы, хранящиеся на самом нижнем уровне.

Задание 2.7. Реализовать класс Any (интересно, что не шаблонный), который позволяет хранить значения любого типа!

Задание 2.8. Необходимо реализовать шаблон SameType с одним статическим константным полем. Шаблон принимает два типовых параметра, и если два типовых параметра шаблона являются одним и тем же типом, то статическое поле value должно хранить значение true, в противном случае – значение false.

1. Ход выполнения

Задание 1. Открываем файл old.txt, затем в цикле проверяем каждую строку, получаемую с помощью функции getline. Внутри цикла используем функцию regex\_replace из библиотеки регулярных выражений, заменяющую «1» на «один» и «0» на «ноль». Выводим строки на консоль независимо от того, была ли произведена замена.

Задание 2.1. Создаём конструктор, который который создает Array размера size, заполненный значениями value типа T. Следующий конструктор создаёт копию переданного объекта Array. В деструкторе с помощью delete удаляем массив myArray. При переопределении оператора присваивания используем ту же логику, что при в конструкторе копирования. Создаём две версии доступа по индексу – по значению и по указателю. В метод size добавляем возвращение значения n. Создаём класс Student для проверки работоспособности класса.

Задание 2.2. Создаём класс Array, выделяем память, так как не существует конструктора по умолчанию. Далее проходимся по массиву и вызываем конструктор. У типа T есть конструктор с параметром и нет оператора присваивания. Создаём цикл for и выводим элементы. Точно по той же схеме создаём конструктор копирования. Далее убираем то, что было в нашем что-то вроде деструктора.

Задание 2.3. Создаём структуру ValueHolder, наследующуюся от ICloneable и содержащую одно поле data\_ типа T. Также определяем метод clone, возвращающий указатель на копию объекта, на котором он был вызван.

Задание 2.4. Создаём функцию copy\_n, принимающую указатель на целевой массив, указатель на массив-источник и кол-во элементов, которые нужно скопировать. В цикле от 0 до size присваиваем целевому массиву значение из массива-источника, приведённого к типу T.

Задание 2.5. Создаём класс Array, выделяем память, создаём деструктор. Создаём шаблон, в котором выделаем память, проходимся по массиву и записываем в переменную min текущий элемент, если он соответствует условию. В основной функции вызываем функцию minimum.

Задание 2.6. Возьмём класс Array из задания 2.1. Создадим функцию flatten, принимающую Array с типами T и поток вывода out. В этой функции каждый элемент массива выводится в поток out. Добавим перегрузку функции, где первым параметром будет массив массивов Array< Array<T>. Внутри функции вызывается исходная функция flatten для каждого массива.

Задание 2.7. Создаём структуру ICloneable. Далее создаём класс Any так, чтобы конструктор можно было вызвать без параметров. В классе Any создаём шаблонный конструктор от одного параметра, чтобы можно было создавать объекты типа Any. Выделенные ресурсы освобождаем с помощью деструктора. Создаём оператор присваивания и шаблонный оператор присваивания. Используем шаблонный метод dynamic\_cast. Обращаемся в основной функции к классу Any.

Задание 2.8. Создаём структуру SameType с одним статическим константным параметром value, по умолчанию имеющим значение false. Эта структура принимает два шаблонных параметра. Перегружаем структуру, создавая ещё одну с таким же именем, но передаём одни шаблонный параметр, из которого создаётся первая структура, при этом поле value теперь имеет значение true. Таким образом, если при создании структуры типы одинаковые, то будет выводиться true, а если разные, то false.

1. Вывод

В ходе выполнения лабораторной работы мы научились обрабатывать исключения в языке программирования C++.

Приложение А

Листинг программного кода языка C++ (Задание 1)

#include <iostream>

#include <fstream>

#include <string>

#include <regex>

using namespace std;

int main()

{

setlocale(LC\_ALL, "RUS");

ifstream file("old.txt");

string s;

while (getline(file, s))

{

s = regex\_replace(s, regex(R"(1)"), "один");

s = regex\_replace(s, regex(R"(0)"), "ноль");

cout << s << endl;

}

return 0;

}

Приложение B

Листинг программного кода на языке C++ (Задание 2.1)

#include <iostream>

#include <cstddef>

using namespace std;

template <typename T>

class Array

{

private:

T\* myArray;

size\_t n;

public:

explicit Array(size\_t size = 0, const T& value = T())

{

myArray = new T[size];

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

{

myArray[i] = value;

cout << myArray[i] << endl;

}

n = size;

}

Array(const Array& mas, size\_t size)

{

cout << "Копирование..." << endl;

myArray = new T[size];

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

{

myArray[i] = mas[i];

cout << myArray[i] << endl;

}

n = size;

}

~Array()

{

delete[] myArray;

}

Array& operator=(const Array& mas)

{

cout << "Присваивание..." << endl;

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

{

myArray[i] = mas[i];

cout << myArray[i] << endl;

}

return \*this;

}

T& operator[](size\_t idx)

{

return myArray[idx];

}

const T& operator[](size\_t idx) const

{

return \*(myArray + idx);

}

size\_t size() const { return n; }

};

class Student

{

public:

string name;

int number;

Student() {};

Student(string name, int number) : name{ name }, number{ number }

{

cout << name << " " << number << endl;

}

};

ostream& operator << (std::ostream& os, const Student& person)

{

return os << person.name << " " << person.number;

}

int main()

{

setlocale(LC\_ALL, "RUS");

Student ann("Ann", 1);

Student john("John", 2);

Array<Student> arr (3, ann);

Array<Student> arr1(3, john);

return 0;

}

Приложение C

Листинг программного кода на языке C++ (Задание 2.2)

#include <iostream>

#include <cstddef>

using namespace std;

template <typename T>

class Array

{

public:

Array(size\_t size, const T& value = T())

{

size\_ = size;

for (size\_t i = 0; i != size\_; ++i)

new (data\_ + i) T(value);

for (size\_t i = 0; i != size\_; ++i)

{

cout << i << endl;

}

}

Array()

{

size\_ = 0;

data\_ = 0;

}

Array(const Array& obj)

{

size\_ = obj.size();

data\_ = (T\*) new char[size\_ \* sizeof(T)];

for (size\_t i = 0; i != size\_; ++i)

new (data\_ + i) T(obj[i]);

for (size\_t i = 0; i != size\_; ++i)

{

cout << i << endl;

}

}

Array& operator=(const Array& obj)

{

if (this != &obj)

{

mem\_free();

size\_ = obj.size();

data\_ = (T\*) new char[size\_ \* sizeof(T)];

for (size\_t i = 0; i != size\_; ++i)

new (data\_ + i) T(obj[i]);

}

return \*this;

}

size\_t size() const

{

return size\_;

}

T& operator[](size\_t i)

{

return \*(data\_ + i);

}

const T& operator[](size\_t i) const

{

return \*(data\_ + i);

}

~Array()

{

mem\_free();

}

void mem\_free()

{

for (size\_t i = 0; i != size\_; ++i)

data\_[i].~T();

delete[](char\*) data\_;

}

private:

size\_t size\_;

T\* data\_;

};

class Class

{

public:

Class(int) {}

};

int main()

{

Array<Class> a(5, 10);

setlocale(LC\_ALL, "RUS");

}

Приложение D

Листинг программного кода на языке C++ (Задание 2.3)

#include <iostream>

using namespace std;

struct ICloneable

{

virtual ICloneable\* clone() const = 0;

virtual ~ICloneable() { }

};

template <typename T>

struct ValueHolder : ICloneable {

T data\_;

ValueHolder(T value) : data\_(value) {};

ValueHolder\* clone() const

{

return new ValueHolder(\*this);

}

~ValueHolder() {}

};

int main() {

ValueHolder<int>\* a = new ValueHolder<int>(651);

ValueHolder<int>\* b = a->clone();

cout << a->data\_ << " " << b->data\_ << " " << endl;

return 0;

}

Приложение E

Листинг программного кода на языке C++ (Задание 2.4)

#include <cstddef>

template <typename T, typename U>

void copy\_n(T\* array, U\* s\_array, size\_t size) {

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

array[i] = (T)s\_array[i];

}

}

int main() {

int a[] = { 0, 4, 9, 4, 6 };

double b[5] = {};

copy\_n(a, b, 5);

return 0;

}

Приложение F

Листинг программного кода на языке C++ (Задание 2.5)

#include <iostream>

#include <cstddef>

using namespace std;

template <typename T>

class Array

{

public:

explicit Array(size\_t size = 0, const T& value = T());

Array(const Array& other);

~Array();

Array& operator=(Array other);

void swap(Array& other);

size\_t size() const;

T& operator[](size\_t idx);

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

private:

size\_t size\_;

T\* data\_;

};

class Class

{

public:

Class(int) {}

};

template<typename T, typename Cmp>T minimum(Array<T> a, Cmp less) {

T min = a[0];

size\_t l = a.size();

for (size\_t i = 1; i < l; ++i)

if (less(a[i], min))

min = a[i];

return min;

}

int main()

{

Array<int> ints(3);

Array<int> less(2);

less[0] = 10;

less[1] = 2;

ints[0] = 10;

ints[1] = 2;

ints[2] = 15;

int min = minimum(ints, less);

}

Приложение G

Листинг программного кода на языке C++ (Задание 2.6)

#include <iostream>

using namespace std;

template <typename T>

class Array

{

private:

T\* myArray;

size\_t n;

public:

explicit Array(size\_t size = 0, const T& value = T())

{

myArray = new T[size];

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

{

myArray[i] = value;

}

n = size;

}

~Array()

{

delete[] myArray;

}

const T& operator[](int idx) const

{

return myArray[idx];

}

size\_t size() const { return n; }

};

template <typename T>

void flatten(const Array<T>& array, ostream& out)

{

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

out << array[i] << " ";

}

}

template <typename T>

void flatten(const Array< Array<T> >& array, ostream& out)

{

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

flatten(array[i], out);

}

}

int main26() {

Array<int> arr(2, 0);

flatten(arr, cout);

Array< Array<int> > arr2(4, arr);

flatten(arr2, cout);

return 0;

}

Приложение H

Листинг программного кода на языке C++ (Задание 2.7)

#include <iostream>

using namespace std;

struct ICloneable

{

virtual ICloneable\* clone() const = 0;

virtual ~ICloneable() { }

};

template <typename T>

struct ValueHolder : ICloneable {

ValueHolder(T value) : data\_(value) {};

ValueHolder\* clone() const {

return new ValueHolder(\*this);

}

~ValueHolder() {}

T data\_;

};

class Any

{

public:

Any() : data\_(0) {}

template < class T >

Any(T data) : data\_(new ValueHolder<T>(data)) {}

~Any() {

delete data\_;

}

Any(Any const& obj) : data\_(obj.data\_->clone()) {}

template <typename T>

Any& operator=(const T& obj) {

// if (this != &obj) {

delete data\_;

data\_ = new ValueHolder<T>(obj);

// }

return \*this;

}

Any& operator=(const Any& obj) {

if (this != &obj) {

delete data\_;

data\_ = obj.data\_->clone();

}

return \*this;

}

template <typename T>

T\* cast() {

if (data\_) {

ValueHolder<T>\* res = dynamic\_cast<ValueHolder<T> \*>(data\_);

if (res) {

return &(res->data\_);

}

}

return 0;

}

ICloneable\* data\_;

};

int main()

{

setlocale(LC\_ALL, "RUS");

Any empty;

Any i(10);

cout << "[1] i=" << i.cast<int>() << endl;

Any copy(i);

cout << "[2] copy=" << copy.cast<int>() << endl;

empty = copy;

cout << "[3] empty=" << empty.cast<int>() << endl;

empty = 0;

cout << "[4] empty=" << empty.cast<int>() << endl;

int\* iptr = i.cast<int>();

cout << "[5] \*iptr=" << iptr << endl;

char\* cptr = i.cast<char>();

cout << "[6] \*cptr=" << cptr << endl;

Any empty2;

int\* p = empty2.cast<int>();

cout << "[7] \*p=" << p << endl;

Any a = 20;

cout << "[8] a=" << a.cast<int>() << endl;

a = 0;

cout << "[9] a=" << a.cast<int>() << endl;

a = 'w';

cout << "[10] a=" << a.cast<char>() << endl;

return 0;

}

Приложение I

Листинг программного кода на языке C++ (Задание 2.8)

#include <iostream>

using namespace std;

template <typename T, typename V>

struct SameType

{

static const bool value = false;

};

template <typename W>

struct SameType<W, W>

{

static const bool value = true;

};

int main()

{

SameType<int, int> s1;

cout << s1.value << endl;

SameType<int, double> s2;

cout << s2.value;

return 0;

}