Министерство науки и высшего образования Российской Федерации

Федеральное государственное образовательное учреждение

высшего образования

«Алтайский государственный технический университет им. И. И. Ползунова»

Факультет информационных технологий

Кафедра прикладной математики

Отчет защищен с отметкой\_\_\_\_\_

Преподаватель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

« \_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2021г.

Отчет

по лабораторной работе №7

**«Некоторые конструкции языков C++, C# и Java»**

по дисциплине «Программирование - 3 семестр»

Студент группы ПИ-02 Немчинов В. И.

Преподаватель доцент, к.т.н. Троцкий В.

Барнаул 2021 г.

**Задание:**

**C++**

1) Продемонстрировать возврат значения из метода через указатель (\*) и через ссылку (&);

2) Продемонстрировать разумное использование оператора this;

3) Придумать и реализовать разумное использование дружественной функции;

4) Выполнить перегрузку операторов '+', '++' (два варианта, префиксный и постфиксный);

5) Заменить массивы char на std::string, продемонстрировать работу с этим классом;

**Java**

6) Продемонстрировать работу с массивом объектов;

7) Продемонстрировать возврат целочисленного значения из метода через вспомогательный класс;

8) Продемонстрировать разумное использование оператора this;

9) Продемонстрировать обработку строк (String);

**C#**

10) Для полей добавить свойства и продемонстрировать работу с ними;

11) Cоздать массив объектов и продемонстрировать работу с ним;

12) В отдельной ветке проекта заменить класс (class) на структуру (struct). Продемонстрировать различие между присваиванием объектов класса и структуры;

13) Продемонстрировать работу с массивом объектов;

14) Продемонстрировать возврат значения через параметр out и через параметр ref. Показать различие этих механизмов;

15) Продемонстрировать разумное использование оператора this;

16) Продемонстрировать перегрузку операторов '+', '++'.

17) Продемонстрировать обработку строк (string);

!) Использование Git обязательно.

***Репозиторий:*** https://github.com/Zireael-of-Hope/Lab7

Код программы:

**C++**

**Lac7.cpp:**

#include <iostream>

#include <conio.h>

#include <string>

#include "Class.h"

using namespace std;

int main()

{

ClassLab7 class1, class2;

int ref(5), ptr(6);

cout << class1.return\_var() << endl;

//Возврат по ссылке

//Использование this в методе

cout << class1.return\_ref\_var(ref) << endl;

//Возврат через указатель

//Использование this в методе

cout << class1.return\_pointer\_var(&ptr) << endl;

//Возврат private поля через дружественную функцию класса

cout << return\_private\_data(class1) << endl;

//Перегрузка префиксного инкримента

class2 = ++class1;

cout << return\_private\_data(class2) << " " << return\_private\_data(class1) << endl;

//Перегрузка постфиксного инкримента

class2 = class1++;

cout << return\_private\_data(class2) << " " << return\_private\_data(class1) << endl;

ClassLab7 string;

string.string\_method();

}

**Class.h:**

#pragma once

#include <iostream>

#include <string>

using namespace std;

class ClassLab7 {

private:

int variable;

public:

int& return\_var() {

return variable;

}

ClassLab7();

int return\_ref\_var(int& variable);

int return\_pointer\_var(int\* variable);

friend int return\_private\_data(ClassLab7& a);

ClassLab7 operator +(const ClassLab7& other);

ClassLab7& operator ++();

ClassLab7& operator ++(int value);

void string\_method();

};

**Class.cpp:**

#include "Class.h"

ClassLab7::ClassLab7(){

variable = 2;

}

int ClassLab7::return\_ref\_var(int& variable) {

variable = variable - this->variable;

return variable;

}

int ClassLab7::return\_pointer\_var(int\* variable) {

\*variable = \*variable - this->variable;

return \*variable;

}

ClassLab7 ClassLab7::operator+(const ClassLab7& other) {

ClassLab7 temp;

temp.variable = this->variable + other.variable;

return temp;

}

ClassLab7& ClassLab7::operator++() {

this->variable++;

return \*this;

}

ClassLab7& ClassLab7::operator++(int value) {

ClassLab7 temp;

temp.variable = this->variable++;

return temp;

}

void ClassLab7::string\_method() {

string line, line\_1;

cout << "Enter the line: ";

getline(cin, line);

cout << "Size() = " << line.size() << endl;

cout << "Resize(): ";

line.resize(10, '!');

cout << line << " line was resized to 10 symbols (if there were less then 10 syblos, '!' were added)" << endl;

cout << "Enter another line: ";

getline(cin, line\_1);

line.insert(3, line\_1);

cout << "Insert() string 2 has been inserted in string 1 after 3 symbol: " << line << endl;

line.clear();

cout << "Clear() string 1 has been cleared: " << line;

}

int return\_private\_data(ClassLab7& a) {

return a.variable;

}

**C#**

**Program.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Runtime.InteropServices;

namespace Lab7\_Sharp

{

class Program

{

static void Main(string[] args)

{

//Работа с массивом

int i, j, k, temp\_int, array\_size\_x = 3, array\_size\_y = 3;

int[,] array = new int[array\_size\_x, array\_size\_y];

Console.WriteLine("Enter double array one elemnt by one: ");

for (i = 0; i < array\_size\_x; i++)

for (j = 0; j < array\_size\_y; j++)

array[i, j] = Convert.ToInt32(Console.ReadLine());

for (i = 0; i < array\_size\_x; i++)

{

for (j = 0; j < array\_size\_y; j++)

Console.Write(array[i, j] + " ");

Console.WriteLine();

}

Console.WriteLine();

for (k = 0; k < array\_size\_x; k++)

for (i = 1; i < array\_size\_y; i++)

if (array[k, i] > array[k, i - 1])

for (j = i; j > 0; j--)

if (array[k, j] > array[k, j - 1])

{

temp\_int = array[k, j];

array[k, j] = array[k, j - 1];

array[k, j - 1] = temp\_int;

}

Console.WriteLine("Array has been sorted in descending order: ");

for (i = 0; i < array\_size\_x; i++)

{

for (j = 0; j < array\_size\_y; j++)

Console.Write(array[i, j] + " ");

Console.WriteLine();

}

Class class\_1 = new(2);

Class class\_2 = new(4);

int example\_variable = 0;

// out и ref

//Можем изменить зачение, а можем не менять

Class.PlusVariableOrNot\_ref(class\_1, ref example\_variable);

Console.WriteLine("\nNew value: " + example\_variable);

//Обязаны изменить значение

Class.AssignPrivateVariableToInt(class\_1, out example\_variable);

Console.WriteLine("\nNew value: " + example\_variable);

//Перегрузка операторов

class\_1++;

Console.WriteLine("\nClass private variable new value: " + class\_1.variable);

Class class\_3 = class\_1 + class\_2;

Console.WriteLine("\nClass private variable new value: " + class\_3.variable);

//Строки

string line\_1, line\_2;

Console.Write("Enter a line: ");

line\_1 = Console.ReadLine();

Console.WriteLine("\nYour line: " + line\_1);

Console.Write("\nEnter another line. It will be compared with previous one: ");

line\_2 = Console.ReadLine();

Console.WriteLine("\nYour second line: " + line\_2);

Console.Write("\nResult of comparing: ");

if (line\_1 == line\_2)

Console.WriteLine("equal");

else Console.WriteLine("not equal");

String[] line\_array = line\_1.Split(" ");

Console.WriteLine("\nString array was created from first line:");

for (i = 0; i < line\_array.Length; i++)

Console.WriteLine(line\_array[i]);

//Struct

Struct hello = new(2);

hello++;

Console.WriteLine(hello.Variable);

}

}

}

**Class.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab7\_Sharp

{

internal class Class

{

public int variable { get; private set; }

public Class(int variable) {

this.variable = variable;

}

//Используя ref мы передаем ссылку на переменную в метод, т.е. в методе мы работаем с самой переменной, а не с ее копией

static public void PlusVariableOrNot\_ref(Class classs, ref int variable\_par) {

Console.WriteLine("Do you want to add private variable to your int? (Yes - 1, No - 0)");

int choice;

bool pass;

do {

pass = false;

choice = Convert.ToInt32(Console.ReadLine());

if (choice == 0 || choice == 1)

pass = true;

} while (pass == false);

switch (choice) {

case 0:

break;

case 1:

variable\_par = +classs.variable;

break;

}

}

//Аналогично ref

//Но,используя out мы обязаны присвоить значение входному параметру с модификатором out

static public void AssignPrivateVariableToInt(Class classs, out int variable\_par)

{

variable\_par = classs.variable;

}

//Перегрузка + для класса

static public Class operator +(Class operator\_1, Class operator\_2) {

return new Class(operator\_1.variable + operator\_2.variable);

}

//Перегрузка ++ для класса

static public Class operator ++(Class operator\_1)

{

operator\_1.variable++;

return operator\_1;

}

}

}

**Struct.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab7\_Sharp

{

//Структура в отлчие от класса значимый тип, а не ссылочный

internal struct Struct

{

public int Variable { get; private set; }

private int Variable\_2;

//Конструктор Struct в отличие от Class обязан определять все поля до возврата

public Struct(int variable) {

Variable = variable;

Variable\_2 = ++variable;

}

public static Struct operator ++(Struct operator1) {

operator1.Variable++;

return operator1;

}

}

**Java:**

**Main.java:**

package com.company;

import java.util.Scanner;

import java.util.Arrays;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int i, j, k, temp\_int, array\_size\_x = 3, array\_size\_y = 3;

int[][] array = new int[array\_size\_x][array\_size\_y];

ThisExample class\_1 = new ThisExample(2);

String line\_1 = new String();

String line\_2 = new String();

for (i = 0; i < array\_size\_x; i++)

for (j = 0; j < array\_size\_y; j++)

array[i][j] = scanner.nextInt();

for (i = 0; i < array\_size\_x; i++) {

for (j = 0; j < array\_size\_y; j++)

System.out.print(array[i][j] + " ");

System.out.println();

}

System.out.println();

for (k = 0; k < array\_size\_x; k++)

for (i = 1; i < array\_size\_y; i++)

if (array[k][i] > array[k][i - 1])

for (j = i; j > 0; j--)

if (array[k][j] > array[k][j - 1]) {

temp\_int = array[k][j];

array[k][j] = array[k][j - 1];

array[k][j - 1] = temp\_int;

}

System.out.println("Array has been sorted in descending order: ");

for (i = 0; i < array\_size\_x; i++) {

for (j = 0; j < array\_size\_y; j++)

System.out.print(array[i][j] + " ");

System.out.println();

}

System.out.println("Which is greater: [0][0] or [0][1]: " + Helper.which\_int\_is\_greater(array[0][0], array[0][1]));

System.out.println("\nDifference between element [2][1] and private variable in ThisExample class: " + class\_1.this\_example\_method(array[2][1]));

scanner.nextLine();

System.out.print("Enter a line: ");

line\_1 = scanner.nextLine();

System.out.println("\nYour line: " + line\_1);

System.out.print("\nEnter another line. It will be compared with previous one: ");

line\_2 = scanner.nextLine();

System.out.println("\nYour second line: " + line\_2);

System.out.print("Result of comparing: ");

if (line\_1.equals(line\_2))

System.out.println("equal");

else System.out.println("not equal");

String[] line\_array = line\_1.split(" ");

System.out.println("String array was created from first line:");

for (i = 0; i < line\_array.length; i++)

System.out.println(line\_array[i]);

}

}

**ThisExpamle.Java:**

package com.company;

public class ThisExample {

private int variable;

public ThisExample(int variable) {

this.variable = variable;

}

public int this\_example\_method(int variable) {

variable = variable - this.variable;

return variable;

}

}

**Helper.Java:**

package com.company;

public class Helper {

public static int which\_int\_is\_greater(int value\_1, int value\_2) {

if (value\_1 > value\_2)

return value\_1;

if (value\_1 < value\_2)

return value\_2;

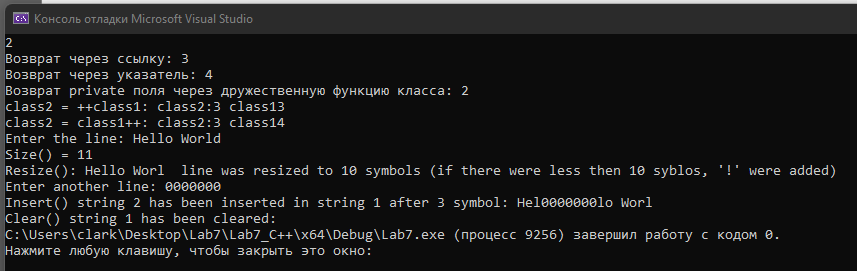
else return 1;

}

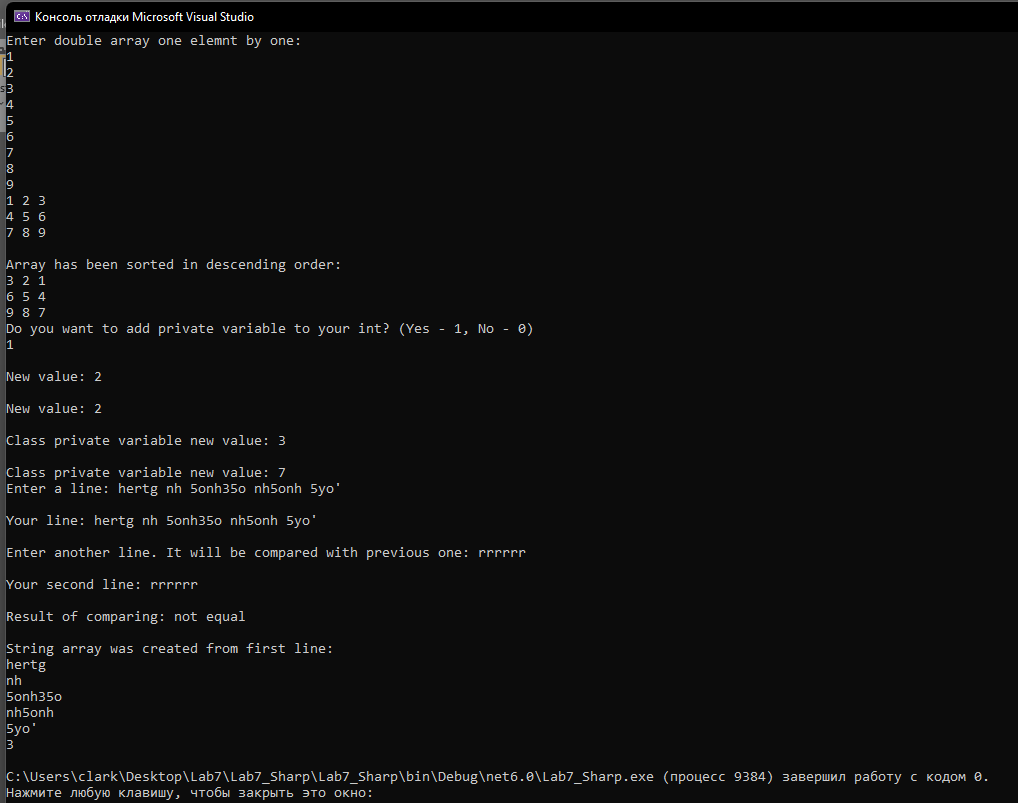
}

Результаты работы программы:

C++



C#



Java

