Министерство образования и науки Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования «Алтайский государственный технический университет им. И.И. Ползунова»

Факультет информационных технологий

Кафедра прикладной математики

Отчёт защищён с оценкой\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Преподаватель Троицкий В.С.

«\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2020 г.

Отчёт

по лабораторной работе № 7

«Ассоциация»

по дисциплине «Программирование - 3»

Студент группы ПИ 92 В.М. Шульпов

Преподаватель доцент, к.т. н. Троицкий В.С.

Барнаул 2020

ЗАДАЧА

**C++**

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

2) Продемонстрировать разумное использование this;

3) Создать дружественную функцию и продемонстрировать ее использование;

4) Выполнить перегрузку операторов '+', '++' (два варианта, префиксный и постфиксный). Продемонстрировать в main;

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 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;

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

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

+17) Продемонстрировать обработку строк (string);

!) Использование Git обязательно.

ТЕОРИЯ

**Когда использовать возврат по адресу**:

  при возврате динамически выделенной памяти;

  при возврате аргументов функции, которые были переданы по адресу.

**Когда не использовать возврат по адресу**:

  при возврате переменных, которые были объявлены внутри функции (используйте возврат по значению);

  при возврате большой структуры или класса, который был передан по ссылке (используйте возврат по ссылке).

**Когда использовать возврат по ссылке**:

  при возврате ссылки-параметра;

  при возврате элемента массива, который был передан в функцию;

  при возврате большой структуры или класса, который не уничтожается в конце функции (например, тот, который был передан в функцию).

**Когда не использовать возврат по ссылке**:

  при возврате переменных, которые были объявлены внутри функции (используйте возврат по значению);

  при возврате стандартного массива или значения указателя (используйте возврат по адресу).

2)

|  |  |  |
| --- | --- | --- |
| **Вопрос** | **Структура** | **Класс** |
| И какого же типа экземпляр объекта? | Структура значимый (value) тип | Класс ссылочный (reference) тип |
| А где “живут” экземпляры этих объектов? | Экземпляры структуры называют *значениями* и “живут” они в стеке (stack). | Экземпляры классов называют *объектами* и “живут” они в куче (heap). |
| Можно ли создать конструктор по умолчанию? | Нет | Да |
| Если создается свой конструктор будет ли компилятор генерировать конструктор по умолчанию? | Да | Нет |
| Если в своём конструкторе не будут инициализированы некоторые поля, будут ли они автоматически инициализированы компилятором? | Нет | Да |
| Разрешается ли инициализировать переменные в месте их объявления? | Нет | Да |

КОД ПРОГРАММЫ

1. **Класс С++**
2. **Структура C#**

using System;

namespace lab7

{

/////////////////////////////////////////////////

class Program

{

static void Main(string[] args)

{

//Console.WriteLine("Kek lol program is working");

Car bmw = new Car();

bmw.init("BMW X6", 3500000, "BLACK", 0, 0); //инициализируем поля объекта

bmw.displayDataCar();

bmw.readCarData();

bmw.displayDataCar();

Console.ReadKey();

}

}

///////////////////////////////////////////////

struct Car

{

private String name;

private int price;

private String color;

private int speed;

private int benzine;

/\*

Ref параметры предназначены для данных, которые могут быть изменены,

out параметры предназначены для данных, которые являются дополнительным выходом для функции

(например, int.TryParse), которые уже используют возвращаемое значение для чего-то.

\*/

public void getStr(out String str)

{

str = "Машина " + name + ", цвет " + color + ", цена " + price;

}

public void changeStr(ref String str)

{

str = str + "!!!";

}

public String secondColor { private set; get; }

public void init(String name, int price, String color, int speed, int benzine)

{

this.name = name;

this.price = price;

this.color = color;

this.secondColor = color;

this.speed = speed;

this.benzine = benzine;

Console.WriteLine("Car initialized!");

}

public void readCarData()

{

string str = "";

int number = 0;

Console.WriteLine("ENTER CAR DATA:");

//////////////////////////////////////////////////

Console.Write("\tname:\t");

this.name = Console.ReadLine();

///////////////////////////////////////////////

bool flag = false; //флаг правильности ввода чисел

while (!flag)

{

Console.Write("\tprice:\t");

str = Console.ReadLine();

flag = int.TryParse(str, out number);

if (!flag)

{

Console.Write("re-enter:");

}

}

this.price = number;

//////////////////////////////////////////////

Console.Write("\tcolor:\t");

this.color = Console.ReadLine();

///////////////////////////////////////////////////////

flag = false;

while (!flag)

{

Console.Write("\tspeed:\t");

str = Console.ReadLine();

flag = int.TryParse(str, out number);

if (!flag)

{

Console.Write("re-enter:");

}

}

this.speed = number;

///////////////////////////////////////////////////////

flag = false;

while (!flag)

{

Console.Write("\tbenzine:\t");

str = Console.ReadLine();

flag = int.TryParse(str, out number);

if (!flag)

{

Console.Write("re-enter:");

}

}

this.benzine = number;

///////////////////////////////////////////////////////

}

public void displayDataCar()

{

Console.WriteLine("\n\tCAR DATA");

Console.WriteLine("\t\tName:\t" + this.name.ToLower());

Console.WriteLine("\t\tPrice:\t" + this.price);

Console.WriteLine("\t\tColor:\t" + this.color.ToLower());

Console.WriteLine("\t\tBenzine:\t" + this.benzine);

Console.WriteLine("\t\tSpeed:\t" + this.speed);

}

public void addBenzine(int liters)

{

Console.WriteLine(liters + "lit. benzine added!");

this.benzine += liters;

}

public void addSpeed(int speed)

{

this.speed += speed;

Console.WriteLine("Car speeded up!");

}

public void reduceSpeed(int speed)

{

if (this.speed > 0)

{

this.speed -= speed;

Console.WriteLine("Car speeded down!");

}

else

{

Console.WriteLine("Car is parking. Car didn't speed down!");

}

}

}

/////////////////////////////////////////////////

}

1. **Класс C#**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab7

{

/////////////////////////////////////////////////

class Program

{

static void Main(string[] args)

{

//Console.WriteLine("Kek lol program is working");

Car bmw = new Car();

Car audi = new Car();

Engine engine1 = new Engine();

Engine engine2 = new Engine();

engine1.init(0, 4395, 625, 8);

engine2.init(0, 4395, 625, 8);

Car[] car = new Car[2];**//массив объектов**

bmw.init("BMW X6", 3500000, "BLACK", 0, 0, engine1); //инициализируем поля объекта

audi.init("Audi", 3300000, "BLACK", 0, 0, engine2); //инициализируем поля объекта

audi.readCarData();

car[0] = bmw;

car[1] = audi;

car[0].displayDataCar();

car[1].displayDataCar();

car[1] = car[1] + 50; **//перегрузка +**

car[1].displayDataCar();

car[1]++; **//перегрузка ++**

car[1].displayDataCar();

Console.WriteLine("\tВторой цвет: " + car[0].secondColor);

String str = "";

car[1].getStr(out str) **//out**

Console.WriteLine("out str="+str);

car[1].changeStr(ref str); **//ref**

Console.WriteLine("ref str=" + str);

Console.ReadKey();

}

}

/////////////////////////////////////////////////

class Engine

{

private int engineRPM; //количество оборотов в минуту

private int capacity; //объем см куб

private int enginePower; //мощность Л.С.

private int quantityOfCylinders; //количество цилиндров

public void init(int engineRPM, int capacity, int enginePower, int quantityOfCylinders)

{

this.engineRPM = engineRPM;

this.capacity = capacity;

this.enginePower = enginePower;

this.quantityOfCylinders = quantityOfCylinders;

}

public void setEngineRPM(int engineRPM)

{

this.engineRPM = engineRPM;

}

public void setCapacity(int capacity)

{

this.capacity = capacity;

}

public void setEnginePower(int enginePower)

{

this.enginePower = enginePower;

}

public void setQuantityOfCylinders(int quantityOfCylinders)

{

this.quantityOfCylinders = quantityOfCylinders;

}

public int getEngineRPM()

{

return this.engineRPM;

}

public int getCapacity()

{

return this.capacity;

}

public int getEnginePower()

{

return this.enginePower;

}

public int getQuantityOfCylinders()

{

return this.quantityOfCylinders;

}

}

/////////////////////////////////////////////////

class Car

{

private String name; **//String тип**

private int price;

private String color; **//String тип**

private int speed;

private int benzine;

private Engine engine;

/\*

Ref параметры предназначены для данных, которые могут быть изменены,

out параметры предназначены для данных, которые являются дополнительным выходом для функции

(например, int.TryParse), которые уже используют возвращаемое значение для чего-то.

\*/

public void getStr(out String str) **//out**

{

str = "Машина " + name + ", цвет " + color + ", цена " + price;

}

public void changeStr(ref String str)

{

str = str + "!!!";**//ref**

}

public String secondColor { private set; get; }**//свойства полей**

public void init(String name, int price, String color, int speed, int benzine, Engine engine)

{

this.name = name;

this.price = price;

this.color = color;

this.secondColor = color;

this.speed = speed;

this.benzine = benzine;

this.engine = engine;

Console.WriteLine("Car initialized!");

}

public void readCarData()

{

string str = "";

int number = 0;

Console.WriteLine("ENTER CAR DATA:");

//////////////////////////////////////////////////

Console.Write("\tname:\t");

this.name = Console.ReadLine();

///////////////////////////////////////////////

bool flag = false; //флаг правильности ввода чисел

while (!flag)

{

Console.Write("\tprice:\t");

str = Console.ReadLine();

flag = int.TryParse(str, out number);

if (!flag)

{

Console.Write("re-enter:");

}

}

this.price = number;

//////////////////////////////////////////////

Console.Write("\tcolor:\t");

this.color = Console.ReadLine();

//////////////////////////////////////////////////

flag = false;

while (!flag)

{

Console.Write("\tengineRPM:\t");

str = Console.ReadLine();

flag = int.TryParse(str, out number);

if (!flag)

{

Console.Write("re-enter:");

}

}

this.engine.setEngineRPM(number);

////////////////////////////////////////////////////

flag = false;

while (!flag)

{

Console.Write("\tcapacity:\t");

str = Console.ReadLine();

flag = int.TryParse(str, out number);

if (!flag)

{

Console.Write("re-enter:");

}

}

this.engine.setCapacity(number);

///////////////////////////////////////////////////////

flag = false;

while (!flag)

{

Console.Write("\tengine power:\t");

str = Console.ReadLine();

flag = int.TryParse(str, out number);

if (!flag)

{

Console.Write("re-enter:");

}

}

this.engine.setEnginePower(number);

///////////////////////////////////////////////////////

flag = false;

while (!flag)

{

Console.Write("\tquantity of cylinders:\t");

str = Console.ReadLine();

flag = int.TryParse(str, out number); **//out**

if (!flag)

{

Console.Write("re-enter:");

}

}

this.engine.setQuantityOfCylinders(number);

///////////////////////////////////////////////////////

flag = false;

while (!flag)

{

Console.Write("\tspeed:\t");

str = Console.ReadLine();

flag = int.TryParse(str, out number);

if (!flag)

{

Console.Write("re-enter:");

}

}

this.speed = number;

///////////////////////////////////////////////////////

flag = false;

while (!flag)

{

Console.Write("\tbenzine:\t");

str = Console.ReadLine();

flag = int.TryParse(str, out number);

if (!flag)

{

Console.Write("re-enter:");

}

}

this.benzine = number;

///////////////////////////////////////////////////////

}

public void displayDataCar()

{

Console.WriteLine("\n\tCAR DATA");

Console.WriteLine("\t\tName:\t" + this.name.ToLower());**//обработка строки**

Console.WriteLine("\t\tPrice:\t" + this.price);

Console.WriteLine("\t\tColor:\t" + this.color.ToLower());**//обработка строки**

Console.WriteLine("\t\tEngineRPM:\t" + this.engine.getEngineRPM());

Console.WriteLine("\t\tCapacity:\t" + this.engine.getCapacity());

Console.WriteLine("\t\tEngine Power:\t" + this.engine.getEnginePower());

Console.WriteLine("\t\tQuanity of cylinders:\t" + this.engine.getQuantityOfCylinders());

Console.WriteLine("\t\tBenzine:\t" + this.benzine);

Console.WriteLine("\t\tSpeed:\t" + this.speed);

}

public void addBenzine(int liters)

{

Console.WriteLine(liters + "lit. benzine added!");

this.benzine += liters;

}

public void startEngine()

{

if (this.benzine > 0)

{

this.engine.setEngineRPM(800);

Console.WriteLine("Engine started!");

}

else

{

Console.WriteLine("No benzine. Engine didn't start!");

}

}

public void stopEngine()

{

if (this.engine.getEngineRPM() > 0)

{

this.engine.setEngineRPM(0);

Console.WriteLine("Engine stopped!");

}

else

{

Console.WriteLine("Engine stopped already!");

}

}

public void addSpeed(int speed)

{

if (this.engine.getEngineRPM() > 0)

{

this.speed += speed;

Console.WriteLine("Car speeded up!");

}

else

{

Console.WriteLine("Engine isn't starting. Car didn't speed up!");

}

}

public void reduceSpeed(int speed)

{

if (this.speed > 0)

{

this.speed -= speed;

Console.WriteLine("Car speeded down!");

}

else

{

Console.WriteLine("Car is parking. Car didn't speed down!");

}

}

public static Car operator +(Car car, int benzine) **//перегрузка +**

{

car.benzine += benzine;

return car;

}

public static Car operator ++(Car car) // ++ один префикс и постфикс!

{

++car.benzine; **//перегрузка ++**

return car;

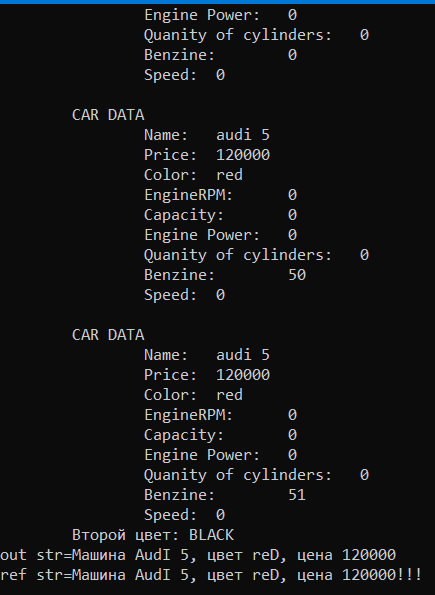
}

}

/////////////////////////////////////////////////

}

1. **Java class**

ТЕСТЫ