**федеральное государственное автономное образовательное учреждение высшего образования**



**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**(ВЫСШАЯ ШКОЛА ПЕЧАТИ И МЕДИАИНДУСТРИИ)**

**(Факультет информационных технологий)**

***(Институт Принтмедиа и информационных технологий)***

***Кафедра Информатики и информационных технологий***

**направление подготовки**

**09.03.02 «Информационные системы и технологии»**

**ЛАБОРАТОРНАЯ РАБОТА № 7**

**Дисциплина: ООП**

### Тема: Псевдонимы типов и статический импорт. Наследование и Преобразование типов

**Выполнил(а): студент(ка) группы 221-3711**

Ежов Тимофей Алексеевич

(Фамилия И.О.)

**Дата, подпись** 29.11.2023

**Проверил:**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Фамилия И.О., степень, звание) **(Оценка)**

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Замечания: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Москва2023**

# Листинг:

namespace LAB7

{

internal class Animal

{

float \_weight; // вес

float \_powerSupply; // энерговооружённость

float \_draught; // водоизмещение

string \_name; // имя

public Animal(float weight, float power, float draught, string name)

{

\_weight = weight;

\_powerSupply = power;

\_draught = draught;

\_name = name;

}

public string ToString()

{

return $"{\_name}, {\_weight}, {\_powerSupply}, {\_draught}";

}

}

}

namespace LAB7

{

internal class Human : Animal

{

bool \_extremist;

public Human(bool extremist, float weight, float power, float draught, string name): base(weight, power, draught, name)

{

\_extremist = extremist;

}

public void AccuseToExtremism()

{

\_extremist = true;

}

public void UnExtremismus()

{

\_extremist = false;

}

public new string ToString()

{

return $"{\_extremist}, {base.ToString()}";

}

}

}

namespace LAB7

{

internal class Centipede: Animal

{

int \_legsCount;

public Centipede(int legs,float weight, float power, float draught, string name): base(weight, power, draught, name)

{

\_legsCount = legs;

}

public void LegCut(int num)

{

if (num >= \_legsCount) {

Console.WriteLine("Сороконожка вмерла");

\_legsCount = 0;

}

\_legsCount -= num;

}

public new string ToString()

{

return $"{\_legsCount}, {base.ToString()}";

}

}

}

namespace LAB7

{

internal class Registry

{

List<Animal> \_animals = new();

public void Add(Animal candidate)

{

if (candidate != null)

{

\_animals.Add(candidate);

}

}

public string CountByType()

{

int humans = 0;

int centipedes = 0;

foreach (Animal animal in \_animals)

{

if( animal is Human)

{

humans++;

}

else if (animal is Centipede)

{

centipedes++;

};

}

return $"Людей: {humans}, \nСороконожек: {centipedes}, \nВсего: {\_animals.Count}";

}

}

}

using LAB7;

Animal Oleg = new(24f, 32f, 342334f, "Oleg");

Human Victor = new(false, 32f, 453f, 5443f, "Vitya");

Animal Oleg1 = new(241f, 321f, 3423341f, "Oleg1");

Human Victor1 = new(true, 321f, 4531f, 54431f, "Vitya1");

Animal Oleg2 = new(242f, 322f, 3423342f, "Oleg2");

Human Victor2 = new(true, 322f, 4532f, 54432f, "Vitya2");

Centipede Gecktor = new(244, 3321f, 4432f, 422f, "Gecktor");

Centipede Gecktor1 = new(2441, 33211f, 44321f, 4221f, "Gecktor1");

Centipede Gecktor2 = new(2442, 33212f, 44322f, 4222f, "Gecktor2");

Console.WriteLine(Oleg.ToString());

Console.WriteLine(Victor.ToString());

Victor.AccuseToExtremism();

Console.WriteLine(Victor.ToString());

Victor.UnExtremismus();

Console.WriteLine(Victor.ToString());

Console.WriteLine(Gecktor.ToString());

Gecktor.LegCut(25);

Console.WriteLine(Gecktor.ToString());

Registry registry = new();

registry.Add(Oleg);

registry.Add(Oleg1);

registry.Add(Oleg2);

registry.Add(Victor);

registry.Add(Victor1);

registry.Add(Victor2);

registry.Add(Gecktor);

registry.Add(Gecktor1);

registry.Add(Gecktor2);

Console.WriteLine(registry.CountByType());

Результат:

