Лабораторная работа 3 по С#

Разработка классов.

using LAB3C\_;

Console.WriteLine("Вариант-9 шоколад");

Chocolate chocolate = new Chocolate("Аленка",120,20,20,20);

Console.WriteLine(chocolate.ToString());

Console.WriteLine($"калорийность {chocolate.Name} {chocolate.Energy():0.00}ккал");

Chocolate chocolate2 = new Chocolate();

Console.WriteLine(chocolate2.ToString());

Console.WriteLine($"калорийность {chocolate2.Name} {chocolate2.Energy():0.00}ккал");

Chocolate chocolate3 = new Chocolate("DOVE", 140, new double[] { 13,25,44,20});

Console.WriteLine(chocolate3.ToString());

Console.WriteLine($"калорийность {chocolate3.Name} {chocolate3.Energy():0.00}ккал");

//if (chocolate3.excess())

//{

// Console.WriteLine("Калорийность превышена! Состав будет изменен");

// Console.WriteLine($"калорийность {chocolate3.Name} {chocolate3.Energy():0.00}ккал");

// Console.WriteLine(chocolate3.ToString());

//}

//else

//{

// Console.WriteLine("Калорийность в норме");

//}

Chocolate chocolate4 = new Chocolate("Горький", 140, 11, 33, 10, 25 );

Console.WriteLine(chocolate4.ToString());

Console.WriteLine($"калорийность {chocolate4.Name} {chocolate4.Energy():0.00}ккал");

if (chocolate4.excess())

{

Console.WriteLine("Калорийность превышена! Состав будет изменен");

Console.WriteLine($"калорийность {chocolate4.Name} {chocolate4.Energy():0.00}ккал");

Console.WriteLine(chocolate4.ToString());

}

else

{

Console.WriteLine("Калорийность в норме");

}

if (chocolate3.Better(chocolate4))

{

Console.WriteLine($"{chocolate3.Name} лучше чем {chocolate4.Name}");

}

else

if (chocolate3.Better(chocolate4))

{

Console.WriteLine($"{chocolate3.Name} НЕ лучше чем {chocolate4.Name}");

}

Console.Write($"Лучший шоколад: {Chocolate.TheBest(chocolate4, chocolate2, chocolate3).Name}");

Console.WriteLine();

chocolate2 = chocolate;

Console.WriteLine($"{chocolate}");

chocolate2.Weight = 300;

Console.WriteLine($"{chocolate2}");

Console.WriteLine($"{chocolate}");

Console.ReadKey();

namespace LAB3C\_

{

public class Chocolate

{

const double Max\_calories = 400;

private string name;

private double weight;

double[] recipe = new double[4];

double[] fit = new double[] { 3,72,0,2};

double[] carbo = new double[] { 10, 33, 67, 14};

double[] protein = new double[] { 15, 10, 0, 22 };

double[] calories = { 9.3, 4.1, 7.2};

public string Name { get => name;}

public double Weight {

get => weight;

set

{

if(value >= 0 & value<=1000)

weight = value;

}

}

public Chocolate():this("Noname",100)

{

}

public Chocolate(string name, double weight):this(name,weight,0)

{

}

public Chocolate(string name, double weight,params double[] recipe)

{

this.name = name;

Weight = weight;

for(int i = 0; i < recipe.Length && i<4; i++)

{

this.recipe[i] = recipe[i];

}

}

public double Energy()

{

double k = 0;

for(int i=0; i<recipe.Length; i++)

{

k += recipe[i] / 100 \* fit[i] \* calories[0]; //жиры

k += recipe[i] / 100 \* protein[i] \* calories[1]; //белки

k += recipe[i] / 100 \* carbo[i] \* calories[2];// углеводы

}

return k;

}

public bool excess()

{

if(Energy() > Max\_calories)

{

recipe[0] += recipe[2];

recipe[2] = 0;

return true;

}

return false;

}

public bool Better(Chocolate other)

{

return Energy() > other.Energy();

}

public static Chocolate TheBest(Chocolate chocolate1, Chocolate chocolate2, Chocolate chocolate3)

{

double energy1 = chocolate1.Energy();

double energy2 = chocolate2.Energy();

double energy3 = chocolate3.Energy();

if(energy1 > energy2 && energy1>energy3) { return chocolate1; }

if(energy2 > energy3) { return chocolate2; }

return chocolate3;

}

public override string? ToString()

{

string str= $"{Name} весит {Weight} грамм";

if(recipe != null)

{

str += $":{recipe[0]}% какао-порошок, {recipe[1]}% какао-масло, {recipe[2]}% сахар, {recipe[3]}% молоко";

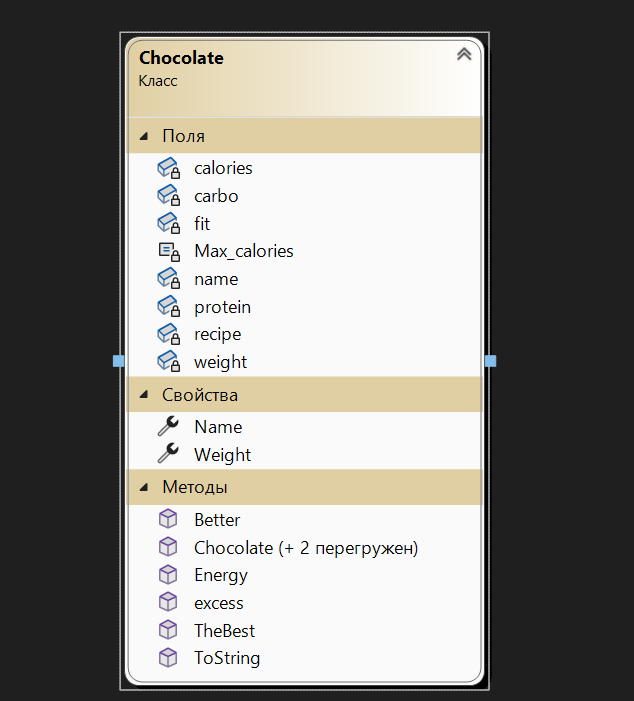
}

return str;

}

}

}



Вариант 1 Студент

using LAB3C\_;

Student student1 = new Student();

Console.WriteLine(student1);

Student student2 = new Student(5, 4, "Сидоров", "Игорь", "Владимирович");

Console.WriteLine(student2);

Student student3 = new Student("Новик", "Cлава", "Евгеньевич");

Console.WriteLine(student3);

Student student5 = new Student(5, 4, "Позняк", "Владимир", "Владимирович", 5 , 4, 4, 5, 4);

Console.WriteLine($"Средний бал студента {student5.Fam} {student5.name} {student5.otch} {student5.AverageBal():0.00}");

Student student6 = new Student(7, 7, "Позняк", "Владимир", "Владимирович", 5, 4, 4, 5, 4);

student6.Otchislenie();

Console.WriteLine(student6);

Student student7 = new Student(5, 4, "Сикорский", "Артем", "Александрович", 5, 4, 3, 5, 4);

student7.Otchislenie();

Console.WriteLine(student7);

if (student6.Sravnenie(student7)) { Console.WriteLine($"{student6.Fam} {student6.name} {student6.otch} учится лучше чем {student7.Fam} {student7.name} {student7.otch}"); }

else { Console.WriteLine($"{student7.Fam} {student7.name} {student7.otch} учится лучше чем {student6.Fam} {student6.name} {student6.otch}"); }

Student student8 = new Student(10, 6, "Димченко", "Артур", "Александрович", 5, 4, 5, 5, 4);

Console.WriteLine(Student.TheBest(student6, student7, student8));

student6 = student8;

Console.WriteLine();

Console.WriteLine(student8);

student6.Ngroup = 19;

Console.WriteLine(student8);

Console.WriteLine(student6);

namespace LAB3C\_

{

public class Student

{

public const int k = 5;

private int ngroup; // номер группы

private int nstudent; // номер студента

private string fam;

public string name;

public string otch;

public int[] ocenki=new int[k];

public int Ngroup

{

get => ngroup;

set

{

if (value >= 0 & value <= 10)

{

ngroup = value;

}

}

}

public int Nstudent

{

get => nstudent;

set

{

if (value >= 0 & value <= 20)

{ nstudent = value; }

}

}

public string Fam { get => fam; }

public Student() : this("Noname", "Noname", "Noname")

{

}

public Student(string fam, string name, string otch):this(0,0,fam,name,otch)

{

}

public Student(int ngroup, int nstudent, string fam, string name, string otch, params int[] ocenki)

{

this.fam = fam;

this.name = name;

this.otch = otch;

Ngroup = ngroup;

Nstudent = nstudent;

for(int i = 0; i < ocenki.Length && i<k; i++)

{

this.ocenki[i] = ocenki[i];

}

}

public double AverageBal() //метод для определения среднего балла

{

double sum = 0;

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

{

sum+= this.ocenki[i];

}

return sum / k;

}

public void Otchislenie() //метод, определяющий отчисление студента (если хотя бы 1 оценка <4 – номер группы обнуляется)

{

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

{

if (this.ocenki[i] < 4) { Ngroup = 0;break; }

}

}

public bool Sravnenie(Student other) //метод, определяющий более успешного студента из двух (возвращает true, если средний балл текущего студента выше)

{

return AverageBal()>other.AverageBal();

}

public static Student TheBest(Student student1, Student student2, Student student3) //статический метод, определяющий более успешного студента из трех (возвращает студента, чей средний балл выше)

{

if(student1.AverageBal() > student2.AverageBal() & student1.AverageBal()> student3.AverageBal()) { return student1; }

if(student2.AverageBal()>student3.AverageBal()) { return student2; }

return student3;

}

public override string? ToString()

{

string str = "Студент под номером ";

if (fam != null & name != null & otch != null)

{

str+=$"{nstudent} группы {ngroup} {fam} {name} {otch}";

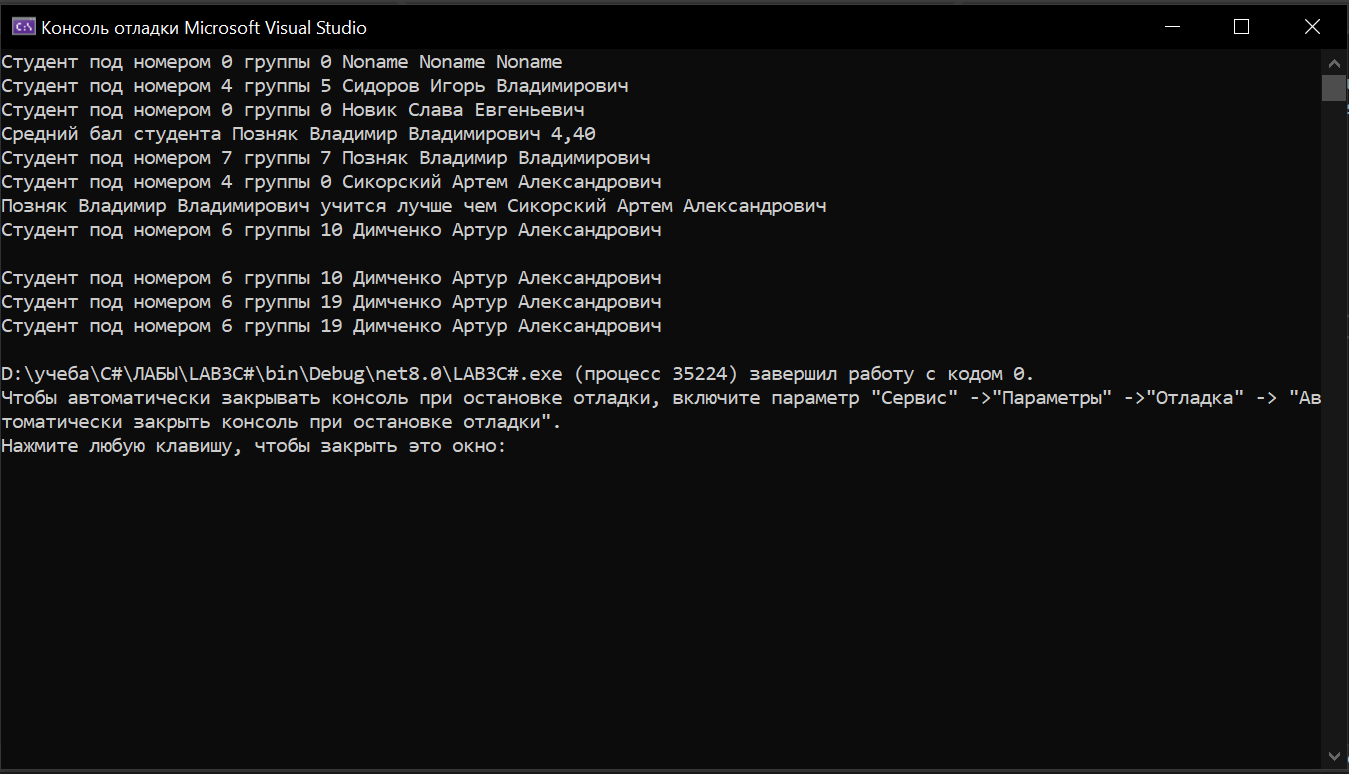
}

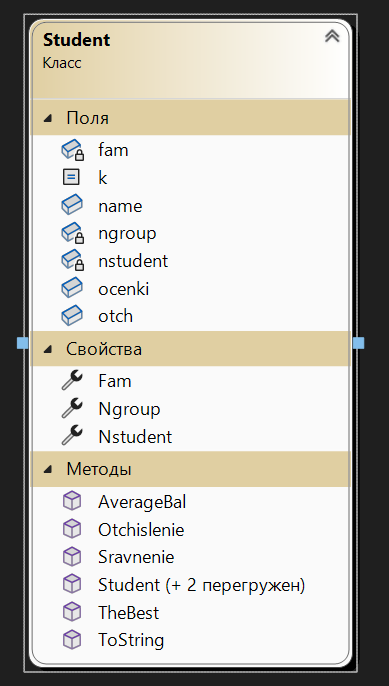
return str;

}

}

}





Вариант 2 Автомобиль

using LAB3C\_;

Auto auto = new Auto();

Console.WriteLine(auto);

Auto auto1 = new Auto("Ауди","А6",50,45, 150);

Console.WriteLine(auto1);

Auto auto2 = new Auto("Мерседес", "Е-класс", 60, 35, 190, 9.5, 2.2, 2.2, 2.1, 2.3);

Console.WriteLine(auto2);

Console.WriteLine($"Максимальный пробег на оставшемся топливе {auto2.Marka} {auto2.model} {auto2.MaxProbeg():0.00}");

Auto auto3 = new Auto("БМВ", "5-серия", 55, 45, 150, 11.5, 1.8, 2.2, 2.1, 2.3);

Console.WriteLine(auto3);

if (auto3.ProboiKolesa()) Console.WriteLine($"Все в порядке.Автомобиль {auto3.Marka} {auto3.model} пройдет дистанцию");

else Console.WriteLine($"Пробой колеса.Автомобиль {auto3.Marka} {auto3.model} остановится");

if (auto3.Sravnenie(auto2)) Console.WriteLine($"{auto3.Marka} {auto3.model} быстрее {auto2.Marka} {auto2.model}");

else Console.WriteLine($"{auto2.Marka} {auto2.model} быстрее {auto3.Marka} {auto3.model}");

Console.WriteLine(Auto.TheBest(auto1, auto2, auto3));

Console.WriteLine();

auto2 = auto3;

Console.WriteLine(auto3);

auto2.rtopl = 9.0;

Console.WriteLine(auto2);

Console.WriteLine(auto3);

Console.ReadKey();

namespace LAB3C\_

{

public class Auto

{

public const int k = 4;

private string marka;

public string model;

private int vbenz; //объем бензобака

private int kbenz; //кол-во бензина

private int tekS; //текущая скорость

public double rtopl; //расход топлива

public double[] pkoles = new double[k]; //давление в каждом колесе

public string Marka { get => marka;}

public int Vbenz

{

get => vbenz;

set

{

if(value>=0 & value<=100)

vbenz = value;

}

}

public int Kbenz

{

get => kbenz;

set

{

if(value >= 0 & value <= vbenz)

kbenz = value;

}

}

public int TekS

{

get => tekS;

set

{

if(value >= 0 & value <= 200)

tekS = value;

}

}

public Auto():this("Noname","Noname",0,0,0)

{

}

public Auto(string marka, string model, int vbenz, int kbenz, int tekS):this(marka,model,vbenz,kbenz,tekS,0)

{

}

public Auto(string marka, string model, int vbenz, int kbenz, int tekS, double rtopl, params double[] pkoles)

{

this.marka = marka;

this.model = model;

this.rtopl = rtopl;

Vbenz = vbenz;

Kbenz = kbenz;

TekS = tekS;

for(int i = 0; i < pkoles.Length & i<k; i++)

{

this.pkoles[i] = pkoles[i];

}

}

public double MaxProbeg() //метод для определения максимального пробега на оставшемся топливе

{

return kbenz / rtopl\*100;

}

public bool ProboiKolesa() //метод, определяющий пробой колеса (если хотя бы в одном колесе давление меньше нормы – автомобиль останавливается)

{

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

{

if (pkoles[i] < 2.0) return false;break;

}

return true;

}

public bool Sravnenie(Auto other) //метод, определяющий более быстрый автомобиль из двух (возвращает true, если скорость текущего авто выше)

{

return tekS > other.tekS;

}

public static Auto TheBest(Auto auto1, Auto auto2, Auto auto3) //статический метод, определяющий более быстрый автомобиль из трех (возвращает авто, чья скорость выше)

{

if (auto1.tekS > auto2.tekS & auto1.tekS > auto3.tekS) return auto1;

if (auto2.tekS > auto3.tekS) return auto2;

return auto3;

}

public override string ToString()

{

string str = "Автомобиль";

if(marka != null & model != null)

{

str += $" {marka} {model} c V-м бензобака {vbenz} и кол-вом бензина {kbenz} с тек S-ю {TekS} и расходом топлива {rtopl} ";

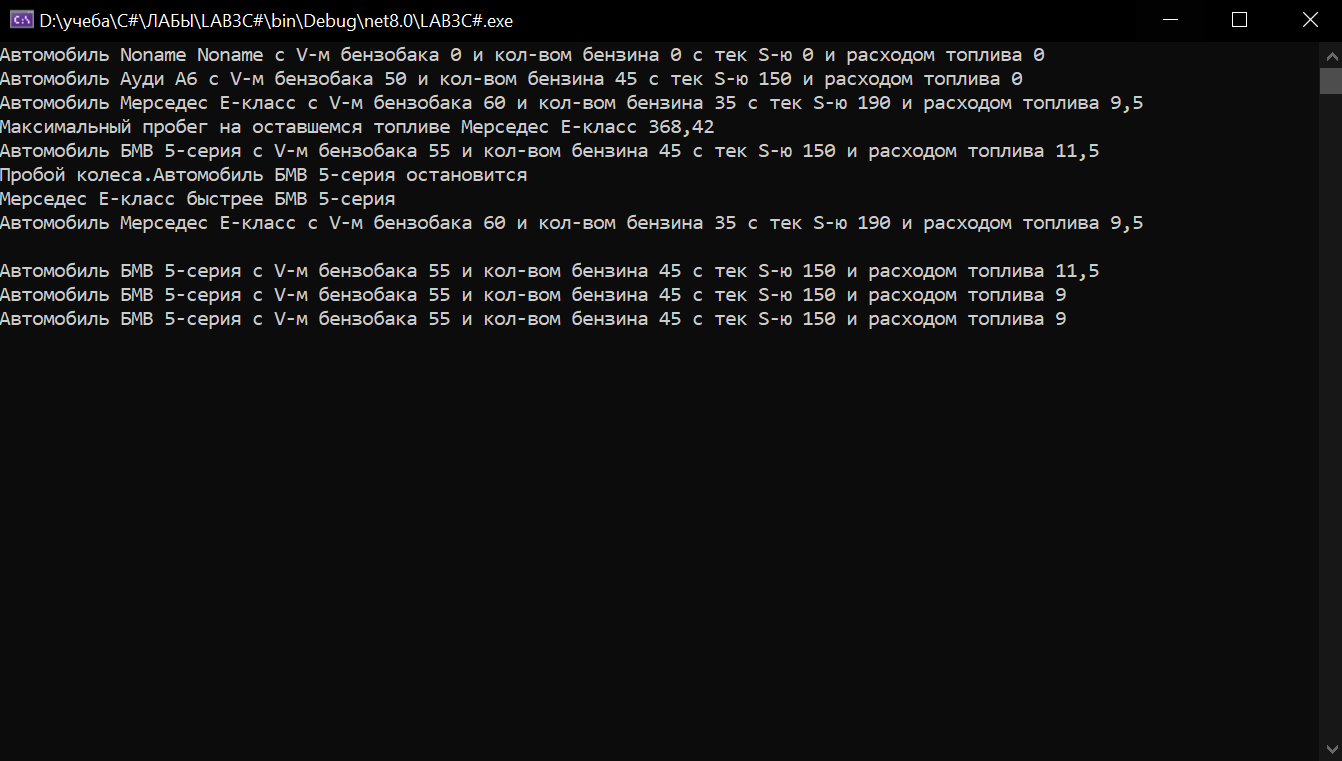
}

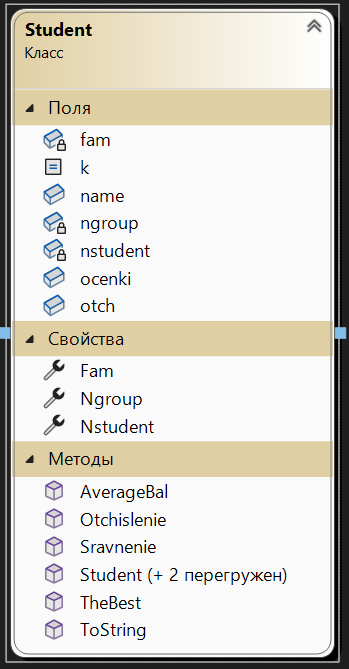
return str;

}

}

}





Вариант 3 Самолет

using LAB3C\_;

Samolet samolet = new Samolet();

Console.WriteLine(samolet);

Samolet samolet1 = new Samolet(123,568,7.4,30,890,11.5);

Console.WriteLine(samolet1);

Samolet samolet2 = new Samolet(125, 789, 9.5, 40, 930, 12, 78, 58, 95, 89);

Console.WriteLine($"Потребление горючего всеми двигателями самолета {samolet2.Nborta} по рейсу {samolet2.Nreisa} {samolet2.PotreblenieGor():0.00}");

Console.WriteLine(samolet2);

Samolet samolet3 = new Samolet(137, 789, 9.5, 40, 950, 12, 78, 30, 95, 89);

samolet3.SboiDvig();

Console.WriteLine(samolet3);

if (samolet3.Sravnenie(samolet2)) Console.WriteLine($"Самолет с номером борта {samolet3.Nborta} быстрее самолета с номером борта {samolet2.Nborta}");

else Console.WriteLine($"Самолет с номером борта {samolet2.Nborta} быстрее самолета с номером борта {samolet3.Nborta}");

Samolet samolet4 = new Samolet(146, 589, 9.5, 40, 870, 12, 78, 70, 95, 89);

Console.WriteLine($"Самый быстрый самолет: самолет с номером борта {Samolet.TheBest(samolet2, samolet3, samolet4).Nborta}");

Console.WriteLine();

samolet3 = samolet2;

Console.WriteLine(samolet2);

samolet3.Nreisa = 156;

Console.WriteLine(samolet3);

Console.WriteLine(samolet2);

Console.ReadKey();

namespace LAB3C\_

{

public class Samolet

{

public const int k = 4;

public int Nborta; //номер борта

public int Nreisa; // номер рейса

public double[] wdvig=new double[k]; //мощность двигателей

private double potrGor; //потр-е горю-го при 100% мощ-и 1 дви-м за 1 час

private double vgor; //общий объем горю-го

private double scor; //скорость

private double vis;//высота

public Samolet():this(0,0,0,0,0,0,0)

{

}

public Samolet(int nborta, int nreisa, double potrGor, double vgor, double scor, double vis):this(nborta, nreisa, potrGor,vgor,scor,vis,0)

{

}

public Samolet(int nborta, int nreisa, double potrGor, double vgor, double scor, double vis, params double[] wdvig)

{

Nborta = nborta;

Nreisa = nreisa;

PotrGor = potrGor;

Vgor = vgor;

Scor = scor;

Vis = vis;

for(int i = 0;i<wdvig.Length & i < k; i++)

{

this.wdvig[i] = wdvig[i];

}

}

public double PotrGor

{

get => potrGor;

set

{

if(value>2& value<12)

potrGor = value;

}

}

public double Vgor

{

get => vgor;

set

{

if(value>=20& value<=50)

vgor = value;

}

}

public double Scor

{

get => scor;

set

{

if(value>=800& value<=1000)

scor = value;

}

}

public double Vis

{

get => vis;

set

{

if(value>=0 & value<=12)

vis = value;

}

}

public double PotreblenieGor() //метод, определяющий потребление горючего в данном режиме полета всеми двигателями

{

double sum = 0;

for(int i = 0; i<wdvig.Length; i++)

{

sum+= wdvig[i]/100\*potrGor;

}

return sum;

}

public void SboiDvig() //метод, определяющий сбой двигателя (если мощность ниже 50%, скорость и высота падают пропорционально мощности)

{

for(int i = 0; i < wdvig.Length; i++)

{

if (wdvig[i] < 50)

{

scor-= (scor / k) \* (100-wdvig[i])/100;

vis -= (vis / k) \* (100 - wdvig[i]) / 100;

}

}

}

public bool Sravnenie(Samolet other) //метод, определяющий более быстрый самолет из двух (возвращает true, если скорость текущего выше)

{

return scor > other.scor;

}

public static Samolet TheBest(Samolet sam1, Samolet sam2, Samolet sam3)

{

if(sam1.scor> sam2.scor && sam1.scor> sam3.scor )return sam1;

if(sam2.scor> sam3.scor )return sam2;

return sam3;

}

public override string? ToString()

{

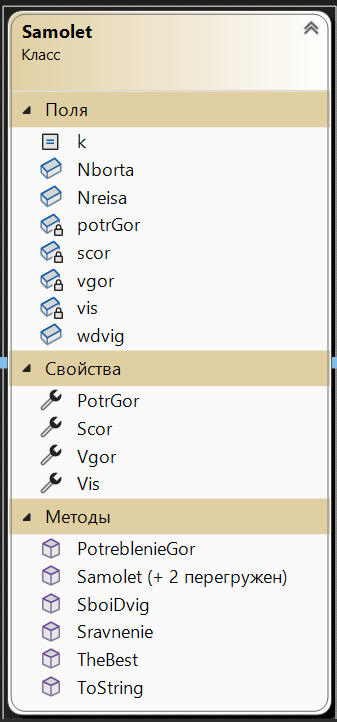
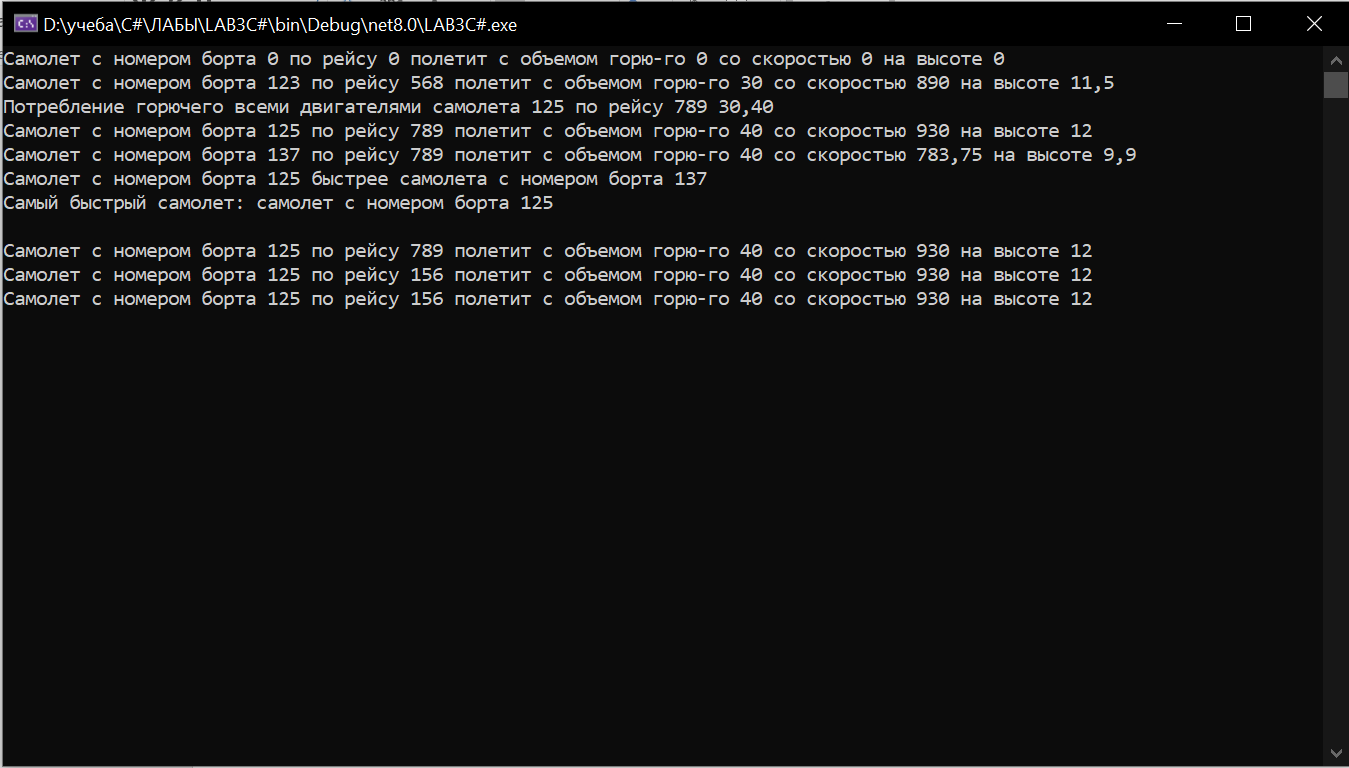
string str = $"Самолет c номером борта {Nborta} по рейсу {Nreisa} полетит с объемом горю-го {vgor} со скоростью {scor} на высоте {vis}";

return str;

}

}

}



Вариант 4 Танкер

using LAB3C\_;

Tanker tanker=new Tanker();

Console.WriteLine(tanker);

Tanker tanker1 = new Tanker("Эсмеральда", 13000, 1250);

Console.WriteLine(tanker1);

Tanker tanker2 = new Tanker("Квазимодо", 15000, 1550, 10, 85, 35, 65);

Console.WriteLine(tanker2);

Console.WriteLine($"Время, необходимое для заполнения танкера {tanker2.Name} {tanker2.Zapolnenie():0.00} часов");

if (tanker2.Oprokidivanie()) Console.WriteLine($"Разность зап-ния танков с разных сторон танкера {tanker2.Name} больше пол-ны их объема \nтанкер {tanker2.Name} может опрок-ться – необ-мо сбросить лишний груз");

else Console.WriteLine($"Все в порядке с танкором {tanker2.Name} - опрокидывание не случится");

Tanker tanker3 = new Tanker("Клод Фролло", 14000, 1450, 70, 55, 45, 55);

if(tanker3.Sravnenie(tanker2)) Console.WriteLine($"Танкер {tanker3.Name} имеет наибольшую загрузку чем {tanker2.Name}");

else Console.WriteLine($"Танкер {tanker2.Name} имеет наибольшую загрузку чем {tanker3.Name}");

Tanker tanker4 = new Tanker("Пьер Гренгуар", 14000, 1350, 50, 44, 67, 89);

Console.WriteLine($"Самый загруженный танкер среди {tanker2.Name}, {tanker3.Name}, {tanker4.Name}: {Tanker.TheBest(tanker2,tanker3,tanker4).Name}");

Console.WriteLine();

tanker4 = tanker2;

Console.WriteLine(tanker4);

tanker4.Vtan = 12345;

Console.WriteLine(tanker4);

Console.WriteLine(tanker2);

Console.ReadKey();

namespace LAB3C\_

{

public class Tanker

{

public const int k = 4;

private string name;

private int vtan; // объем танков (все танки одного объема)

public double[] napol=new double[k];// наполненности каждого танка

private double scor;//скорости заполнения танков (одна скорость для всех)

public Tanker():this("Noname", 0, 0)

{

}

public Tanker(string name, int vtan, double scor):this(name,vtan,scor,0)

{

}

public Tanker(string name, int vtan, double scor, params double[] napol)

{

this.name = name;

Vtan = vtan;

Scor = scor;

for(int i = 0; i <napol.Length & i<k; i++)

{

this.napol[i] = napol[i];

}

}

public string Name { get => name;}

public int Vtan

{

get => vtan;

set

{

if(value>=12000 & value<=15000)

vtan = value;

}

}

public double Scor

{

get => scor;

set

{

if(value>=0 & value<=2000)

scor = value;

}

}

public double Zapolnenie() //метод для определения времени, необходимого для полного заполнения танкера

{

double sum = 0;

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

{

sum += ((100-napol[i])/100\*vtan)/scor;

}

return sum;

}

public bool Oprokidivanie() //метод, определяющий угрозу опрокидывания (если разность заполнения

//танков с разных сторон танкера больше половины их объема – танкер

//может опрокинуться – необходимо сбросить лишний груз)

{

double stor1=0;

double stor2=0;

double napolvsego = 0;

for(int i = 0;i < k; i += 2)

{

stor1 += (napol[i] / 100 \* vtan);

}

for (int i = 1; i < k; i += 2)

{

stor2 += (napol[i] / 100 \* vtan);

}

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

{

napolvsego += (napol[i] / 100 \* vtan);

}

if (Math.Abs(stor1-stor2)>napolvsego/2) return true;

return false;

}

public bool Sravnenie(Tanker other) //метод, определяющий танкер с наибольшей загрузкой из двух (возвращает true, если текущий загружен больше)

{

double zapol1 = 0;

double zapol2 = 0;

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

{

zapol1 += napol[i] / 100 \* vtan;

}

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

{

zapol2 += other.napol[i] / 100 \* vtan;

}

return zapol1 > zapol2;

}

public static Tanker TheBest(Tanker tank1, Tanker tank2, Tanker tank3)// статический метод, определяющий танкер с наибольшей загрузкой из

//трех(возвращает объект, чья загрузка больше)

{

double zapoln1 = 0;

double zapoln2 = 0;

double zapoln3 = 0;

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

{

zapoln1 += tank1.napol[i] / 100 \* tank1.vtan;

}

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

{

zapoln2 += tank2.napol[i] / 100 \* tank2.vtan;

}

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

{

zapoln3 += tank3.napol[i] / 100 \* tank3.vtan;

}

if (zapoln1 > zapoln2 && zapoln1 > zapoln3) return tank1;

if(zapoln2> zapoln3 ) return tank2;

return tank3;

}

public override string? ToString()

{

string str = "Танкер ";

if(name != null)

{

str += $"{name} с объемом танков {vtan} и скоростью заполнения танков {scor}";

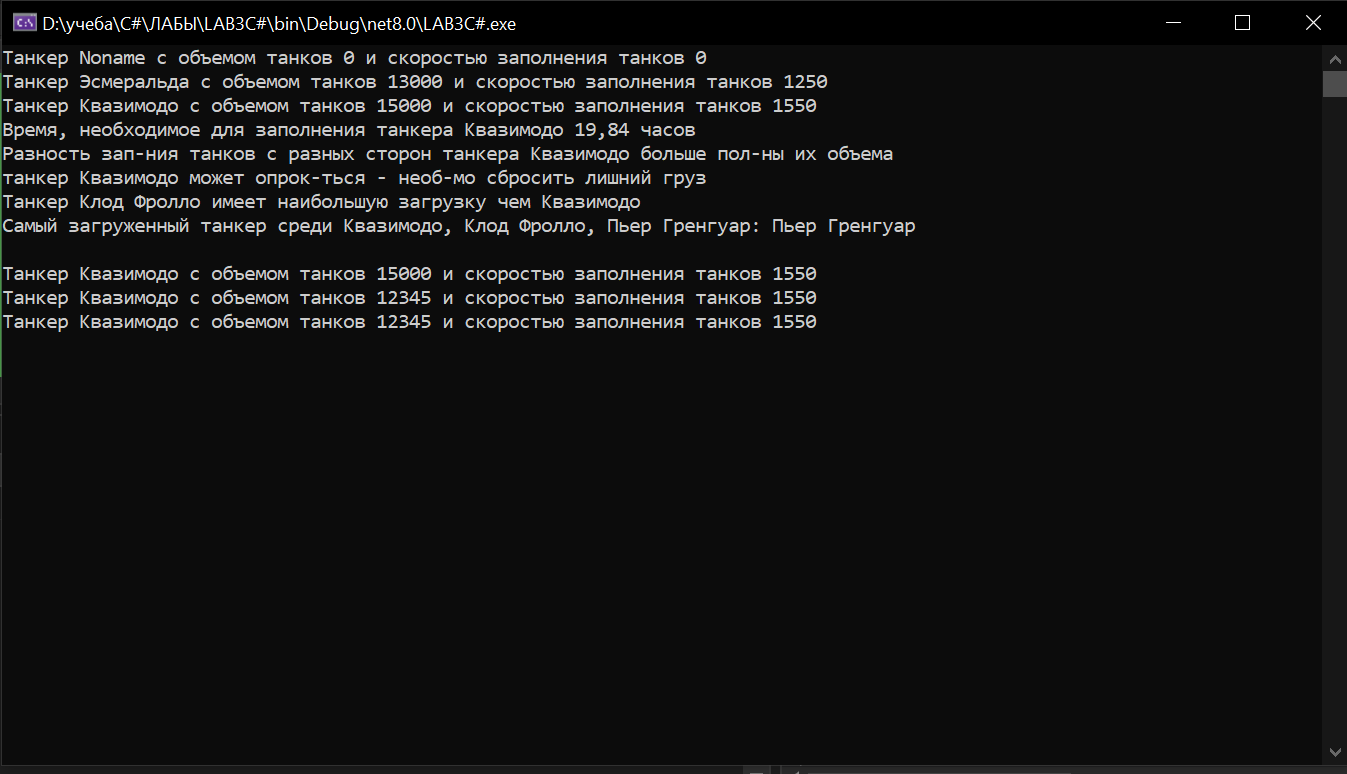
}

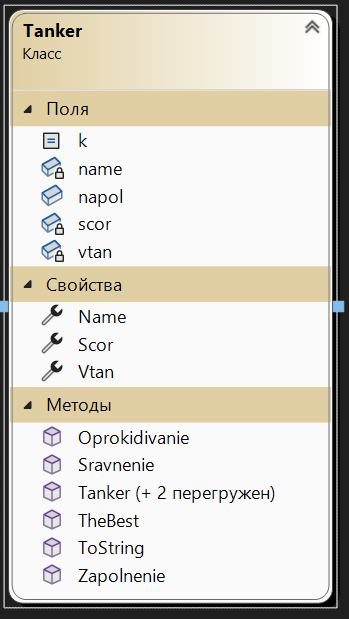
return str;

}

}

}





Вариант 5 Гостиница

using LAB3C\_;

Gostinica gostinica=new Gostinica();

Console.WriteLine(gostinica);

Gostinica gostinica1 = new Gostinica("Монте-Кристо", "ул.Маркса,38", 800);

Console.WriteLine(gostinica1);

Gostinica gostinica2 = new Gostinica("Эдмон Дантес", "ул.Первомайская,45", 800, 3, 4, 2, 2);

Console.WriteLine(gostinica2);

Console.WriteLine($"Выручка гостиницы {gostinica2.Name} за день составила {gostinica2.Viruchka()}");

Gostinica gostinica3 = new Gostinica("Мерседес", "ул.Красноармейская,78", 700, 0, 0, 0, 2);

Console.WriteLine(gostinica3);

gostinica3.Rentabel();

Console.WriteLine(gostinica3);

Gostinica gostinica4 = new Gostinica("Альбер Де Морсер", "ул.Луговая,32", 600, 3, 3, 4, 2);

if (gostinica4.Sravnenie(gostinica3)) { Console.WriteLine($"Гостиница {gostinica4.Name} более рентабельна чем {gostinica3.Name}"); }

else { Console.WriteLine($"Гостиница {gostinica3.Name} более рентабельна чем {gostinica4.Name}"); }

Gostinica gostinica5 = new Gostinica("Де Вильфор", "ул.Нежная,49", 900, 3, 2, 3, 2);

Console.WriteLine($"Самая рентабельная гостиница среди {gostinica5.Name}, {gostinica4.Name}, {gostinica2.Name}: {Gostinica.TheBest(gostinica5, gostinica4, gostinica2).Name}");

Console.WriteLine();

gostinica5 = gostinica4;

Console.WriteLine(gostinica4);

gostinica5.Adress = "ул.Парковая, 67";

Console.WriteLine(gostinica5);

Console.WriteLine(gostinica4);

Console.ReadKey();

namespace LAB3C\_

{

public class Gostinica

{

public const int k = 4;

private string name;

public string Adress;

public int[] kolZh=new int[k]; //к-во жильцов в каждом номере (массив)

private double stoi; //стоимость проживания 1 жильца за сутки

public Gostinica():this("Noname","NoAdress",0)

{

}

public Gostinica(string name, string adress, double stoi):this(name,adress,stoi,0)

{

}

public Gostinica(string name, string adress, double stoi, params int[] kolZh)

{

this.name = name;

Adress = adress;

Stoi = stoi;

for(int i = 0; i < kolZh.Length & i<k; i++)

{

this.kolZh[i] = kolZh[i];

}

}

public string Name { get => name; }

public double Stoi

{

get => stoi;

set

{

if(value>=0 & value <=1000)

stoi = value;

}

}

public double Viruchka() //метод для определения выручки гостиницы за день

{

double sum = 0;

for(int i = 0;i < kolZh.Length; i++)

{

sum+= kolZh[i]\*stoi;

}

return sum;

}

public void Rentabel() //метод, определяющий, падение рентабельности (если более половины

//номеров пустуют, стоимость падает на 20%)

{

int c = 0;

for(int i = 0; i < kolZh.Length; i++)

{

if (kolZh[i] == 0) { c++; }

}

if(c > k / 2)

{

stoi \*= 0.8;

}

}

public bool Sravnenie(Gostinica other) //метод, определяющий более рентабельную гостиницу из двух – по

//проценту заполненных номеров(возвращает true, если текущая более рентабельна)

{

int c1 = 0;

int c2 = 0;

for(int i = 0;i<kolZh.Length;i++)

{

if (kolZh[i] == 4) { c1++; }

}

for (int i = 0; i < kolZh.Length; i++)

{

if (other.kolZh[i] == 4) { c2++; }

}

return c1 > c2;

}

public static Gostinica TheBest(Gostinica gost1, Gostinica gost2, Gostinica gost3) //статический метод, определяющий более рентабельную гостиницу из

//трех(возвращает объект, чья рентабельность выше)

{

double rent1 = 0;

double rent2 = 0;

double rent3 = 0;

for(int i = 0; i <gost1.kolZh.Length; i++)

{

rent1 += gost1.kolZh[i] \* gost1.stoi;

}

for (int i = 0; i < gost2.kolZh.Length; i++)

{

rent2 += gost2.kolZh[i] \* gost2.stoi;

}

for (int i = 0; i < gost3.kolZh.Length; i++)

{

rent3 += gost3.kolZh[i] \* gost3.stoi;

}

if(rent1 > rent2 & rent1>rent3) { return gost1; }

if (rent2 > rent3) return gost2;

return gost3;

}

public override string? ToString()

{

string str = "Гостиница ";

if(name != null & Adress!=null)

{

str += $"{name} по адресу {Adress} со стоимостью проживания 1 жильца за сутки {stoi}";

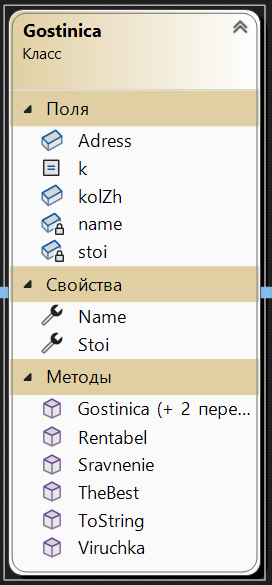
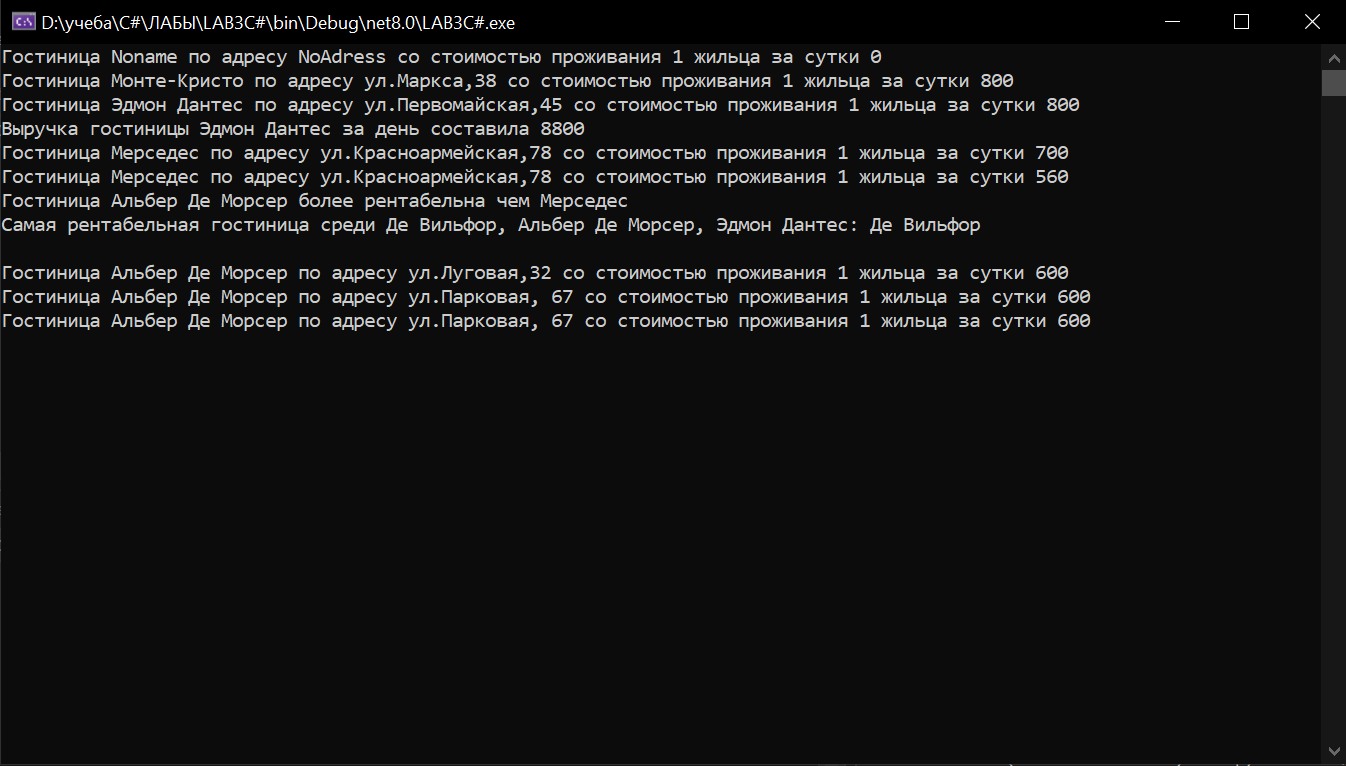
}

return str;

}

}

}



Вариант 6 Грузоперевозки

using LAB3C\_;

Gruzoper gruzoper = new Gruzoper();

Console.WriteLine(gruzoper);

Gruzoper gruzoper1 = new Gruzoper("Колеса", 560, 96);

Console.WriteLine(gruzoper1);

Gruzoper gruzoper2 = new Gruzoper("Фургоны", 660, 88, 24, 24, 24, 24);

Console.WriteLine(gruzoper2);

Console.WriteLine($"Максимально возможная выручка при полной загрузке всех фур фирмы {gruzoper2.Name}: {gruzoper2.Viruchka()} рублей");

Gruzoper gruzoper3 = new Gruzoper("Багаж",590, 57, 24, 24, 24, 24);

Console.WriteLine(gruzoper3);

gruzoper3.Rentabel();

Console.WriteLine(gruzoper3);

if(gruzoper2.Sravnenie(gruzoper3)) Console.WriteLine($"Фирма {gruzoper2.Name} более рентабельна чем {gruzoper3.Name}");

else Console.WriteLine($"Фирма {gruzoper3.Name} более рентабельна чем {gruzoper2.Name}");

Gruzoper gruzoper4 = new Gruzoper("Баранки", 690, 90, 24, 24, 24, 24);

Console.WriteLine($"Самая рентабельная фирма среди {gruzoper4.Name}, {gruzoper3.Name}, {gruzoper2.Name}: {Gruzoper.TheBest(gruzoper4,gruzoper3,gruzoper2).Name}");

Console.WriteLine();

gruzoper4 = gruzoper3;

Console.WriteLine(gruzoper4);

gruzoper3.Stoi = 345;

Console.WriteLine(gruzoper3);

Console.WriteLine(gruzoper4);

Console.ReadKey();

namespace LAB3C\_

{

public class Gruzoper

{

public const int k = 4;

private string name;

public int[] gruzopod=new int[k]; //грузоподъемность фур (массив)

private double stoi; //стоимость перевозки 1 тонны

private double massa; //суммарная массa перевезенных грузов

public Gruzoper():this("Noname",0,0)

{

}

public Gruzoper(string name, double stoi, double massa):this(name, stoi, massa, 0)

{

}

public Gruzoper(string name, double stoi, double massa, params int[] gruzopod)

{

this.name = name;

Stoi = stoi;

Massa = massa;

for(int i = 0; i < gruzopod.Length & i<k; i++)

{

this.gruzopod[i] = gruzopod[i];

}

}

public string Name { get => name; }

public double Stoi

{

get => stoi;

set

{

if(value>0 & value <1000)

stoi = value;

}

}

public double Massa

{

get => massa;

set

{

if(value>0)

massa = value;

}

}

public double Viruchka()//метод для определения процентной выручки от максимально возможной

//(суммарную грузоподъемность умножить на стоимость перевозки 1 тонны)

{

double sum = 0;

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

{

sum += gruzopod[i] \* stoi;

}

return sum;

}

public void Rentabel() //метод, определяющий, падение рентабельности (если масса реально

//перевезенных грузов меньше максимальной более чем на 40%, стоимость падает на 15%)

{

int sumpergruz = 0;

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

{

sumpergruz += gruzopod[i];

}

if (sumpergruz\*0.6 > massa) stoi \*= 0.85;

}

public bool Sravnenie(Gruzoper other) //метод, определяющий более рентабельную фирму из двух – по проценту

//рентабельности(возвращает true, если текущая более рентабельна)

{

return massa>other.massa;

}

public static Gruzoper TheBest(Gruzoper gruz1, Gruzoper gruz2, Gruzoper gruz3)//статический метод, определяющий более рентабельную фирму из трех

//(возвращает объект, чья рентабельность выше)

{

if (gruz1.massa > gruz2.massa & gruz1.massa > gruz3.massa) { return gruz1; }

if(gruz2.massa>gruz3.massa) return gruz2;

return gruz3;

}

public override string? ToString()

{

string str = "Грузоперевозки выполняются ";

if(name != null)

{

str += $"{name} со стоимостью перевозки 1 тонны {stoi} и суммарной массой перевезенных грузов {massa}";

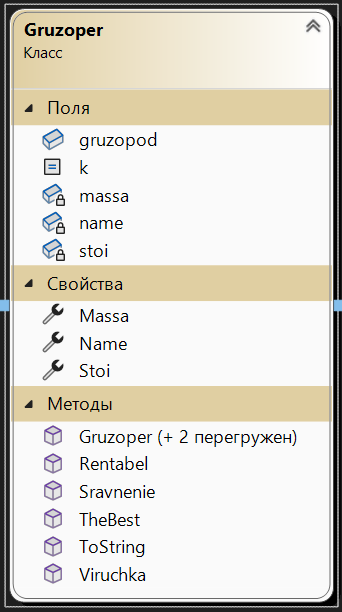
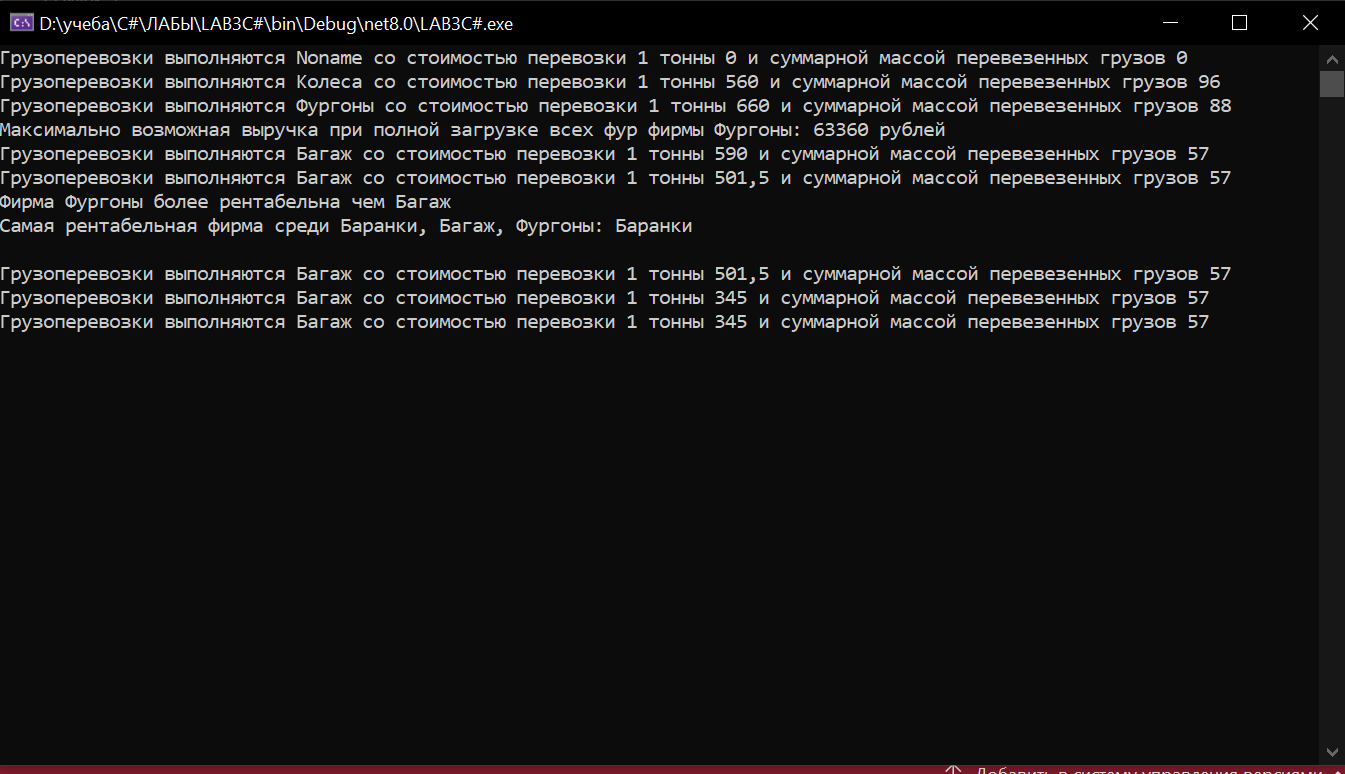
}

return str;

}

}

}



Вариант 7 Грузовой поезд

using LAB3C\_;

Poezd poezd=new Poezd();

Console.WriteLine(poezd);

Poezd poezd1 = new Poezd(1, 5600, "Вересовка");

Console.WriteLine(poezd1);

Poezd poezd2 = new Poezd(2, 4001, "Вересовка", 88, 90, 87, 65);

Console.WriteLine(poezd2);

Console.WriteLine($"Общая масса состава {poezd2.Nomer}: {poezd2.ObMassa()}");

Poezd poezd3 = new Poezd(3, 4001, "Энергетик", 14000, 14000, 15000, 16000);

if (poezd3.Previshenie()) Console.WriteLine($"превышение максимальной массы для данной мощности локомотива {poezd3.Nomer} (необходимо отцепить часть вагонов) ");

else Console.WriteLine("Превышения максимальной массы для данной мощности локомотива нет.");

Poezd poezd4 = new Poezd(4, 12001, "Узда", 78, 80, 77, 95);

if(poezd4.Sravnenie(poezd2)) Console.WriteLine($"Грузовой поезд под номером {poezd4.Nomer} более тяжелый чем грузовой поезд под номером {poezd2.Nomer}");

else Console.WriteLine($"Грузовой поезд под номером {poezd2.Nomer} более тяжелый чем грузовой поезд под номером {poezd4.Nomer}");

Poezd poezd5 = new Poezd(5, 14001, "Столбцы", 57, 89, 98, 88);

Console.WriteLine($"Самый тяжелый поезд из поездов {poezd5.Nomer},{poezd4.Nomer},{poezd2.Nomer}: {Poezd.TheBest(poezd5,poezd4,poezd2).Nomer}");

Console.WriteLine();

poezd5 = poezd4;

Console.WriteLine(poezd5);

poezd5.Mosh = 13000;

Console.WriteLine(poezd5);

Console.WriteLine(poezd4);

Console.ReadKey();

namespace LAB3C\_

{

public class Poezd

{

public const int k = 4;

private int nomer;

private double mosh;

private string punktnaz;

public double[] massa = new double[k];

public Poezd():this(0,0,"Noname")

{

}

public Poezd(int nomer, double mosh, string punktnaz):this(nomer,mosh,punktnaz,0)

{

}

public Poezd(int nomer, double mosh, string punktnaz, params double[] massa)

{

this.punktnaz = punktnaz;

Nomer = nomer;

Mosh = mosh;

for(int i = 0; i < massa.Length & i<k; i++)

{

this.massa[i] = massa[i];

}

}

public int Nomer

{

get => nomer;

set

{

if(value>0)

nomer = value;

}

}

public double Mosh

{

get => mosh;

set

{

if(value>4000 & value<18000)

mosh = value;

}

}

public double ObMassa()//метод, определяющий общую массу состава

{

double sum = 0;

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

{

sum += massa[i];

}

return sum;

}

public bool Previshenie() //метод, определяющий, превышение максимальной массы для данной

//мощности локомотива(необходимо отцепить часть вагонов)

{

return ObMassa() > (mosh / 1.36 \* 20);

}

public bool Sravnenie(Poezd other) //метод, определяющий более тяжелый поезд из двух (возвращает true,

//если текущий более тяжелый)

{

return ObMassa()>other.ObMassa();

}

public static Poezd TheBest(Poezd p1, Poezd p2, Poezd p3)//статический метод, определяющий более тяжелый поезд их трех

//(возвращает более тяжелый объект)

{

if (p1.ObMassa() > p2.ObMassa() & p1.ObMassa() > p3.ObMassa()) return p1;

if(p2.ObMassa()>p3.ObMassa()) return p2;

return p3;

}

public string Punktnaz { get => punktnaz;}

public override string? ToString()

{

string str = $"Грузовой поезд под номером {nomer} с мощностью лок-ва {mosh}, ";

if(punktnaz != null)

{

str += $"идущий в пункт назн-я {punktnaz}";

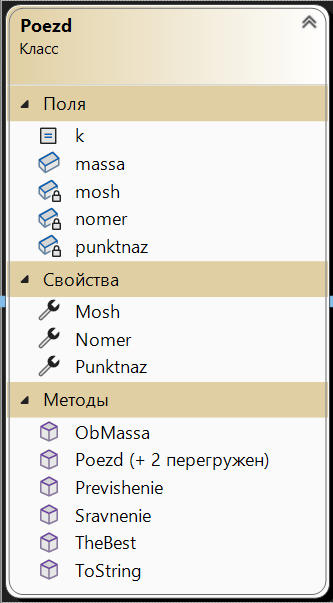
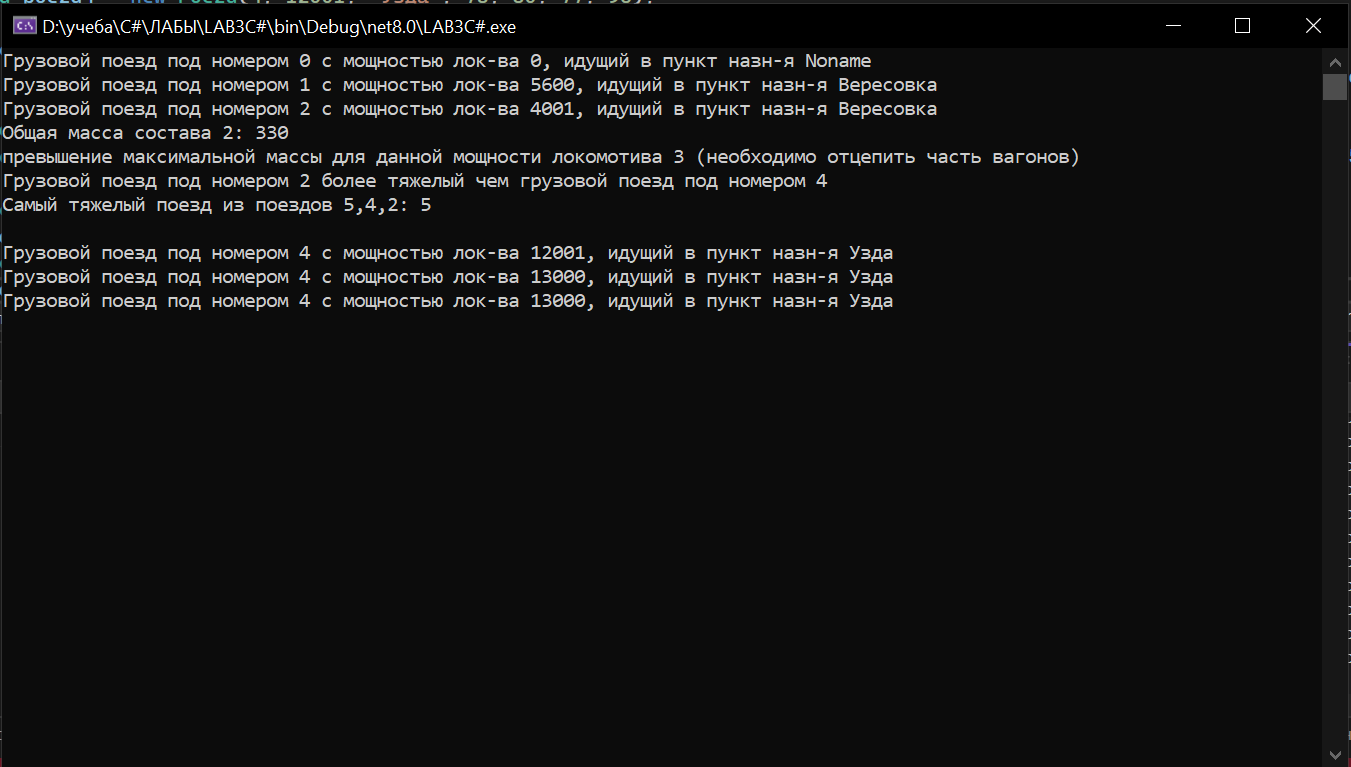
}

return str;

}

}

}



Вариант 8 Магазин

using LAB3C\_;

Magazin magazin = new Magazin();

Console.WriteLine(magazin);

Magazin magazin1 = new Magazin("Визит", 4, 3500, 605);

Console.WriteLine(magazin1);

Console.WriteLine($"Премия каждому продавцу по итогам месяца: {magazin1.Premiya():0.00}");

Magazin magazin2 = new Magazin("Закат", 7, 12000, 905, 4000, 1000);

Console.WriteLine(magazin2);

if(magazin2.PadenieRentab()) Console.WriteLine("падение рентабельности");

else Console.WriteLine("Все в порядке. Работаем дальше");

Magazin magazin3 = new Magazin("Магия", 6, 11000, 705, 5000, 2000);

if(magazin3.Sravnenie(magazin2)) Console.WriteLine($"Магазин {magazin3.Name} рентабельнее чем магазин {magazin2.Name}");

else Console.WriteLine($"Магазин {magazin2.Name} рентабельнее чем магазин {magazin3.Name}");

Magazin magazin4 = new Magazin("Мария", 5, 11000, 755, 4500, 1500);

Console.WriteLine($"Самый рентабельный магазин среди {magazin4.Name}, {magazin3.Name}, {magazin2.Name}: {Magazin.TheBest(magazin4,magazin3,magazin2).Name}");

Console.WriteLine();

magazin4 = magazin3;

Console.WriteLine(magazin4);

magazin4.SumVir = 13567;

Console.WriteLine(magazin4);

Console.WriteLine(magazin3);

Console.ReadKey();

namespace LAB3C\_

{

public class Magazin

{

private string name;

private int kolProd; //количество продавцов

private double sumVir; //суммарная выручка

private double srZp; //средняя зарплата

private double stZak; //стоимость закупки

private double naklRas; //накладные расходы

public Magazin():this("Noname",0,0,0)

{

}

public Magazin(string name, int kolProd, double sumVir, double srZp) : this(name, kolProd, sumVir, srZp,0,0)

{

}

public Magazin(string name, int kolProd, double sumVir, double srZp, double stZak, double naklRas)

{

this.name = name;

KolProd = kolProd;

SumVir = sumVir;

SrZp = srZp;

StZak = stZak;

NaklRas = naklRas;

}

public string Name { get => name;}

public int KolProd

{

get => kolProd;

set

{

if(value >0)

kolProd = value;

}

}

public double SumVir

{

get => sumVir;

set

{

if(value >0)

sumVir = value;

}

}

public double SrZp

{

get => srZp;

set

{

if(value>0)

srZp = value;

}

}

public double StZak

{

get => stZak;

set

{

if( value >0)

stZak = value;

}

}

public double NaklRas

{

get => naklRas;

set

{

if(value >0) naklRas = value;

}

}

public double Premiya() //метод для определения премии (чистая прибыль за месяц/ к-во продавцов)

{

return (sumVir - srZp\*kolProd - stZak - naklRas) / kolProd;

}

public bool PadenieRentab()//метод, определяющий, падение рентабельности (если прибыль меньше 10% от суммарной выручки)

{

return (sumVir - srZp \* kolProd - stZak - naklRas)<(sumVir\*0.1);

}

public bool Sravnenie(Magazin other) //метод, определяющий более рентабельную фирму из двух – по проценту

//рентабельности(возвращает true, если текущая более рентабельна)

{

double rent1 = sumVir - srZp \* kolProd - stZak - naklRas;

double rent2 = other.sumVir - other.srZp \* other.kolProd - other.stZak - other.naklRas;

return (rent1 > rent2);

}

public static Magazin TheBest(Magazin mag1, Magazin mag2, Magazin mag3) //статический метод, определяющий более рентабельную фирму из трех

//(возвращает объект, чья рентабельность выше)

{

double rent1 = mag1.sumVir - mag1.srZp \* mag1.kolProd - mag1.stZak - mag1.naklRas;

double rent2 = mag2.sumVir - mag2.srZp \* mag2.kolProd - mag2.stZak - mag2.naklRas;

double rent3 = mag3.sumVir - mag3.srZp \* mag3.kolProd - mag3.stZak - mag3.naklRas;

if(rent1 > rent2 & rent1>rent3) { return mag1; }

if(rent2 > rent3 ) return mag2;

return mag3;

}

public override string? ToString()

{

string str = "Магазин ";

if(name != null)

{

str += $"{name} с кол-вом продавцов {kolProd} со средней зп {srZp} и суммарной выручке {SumVir}";

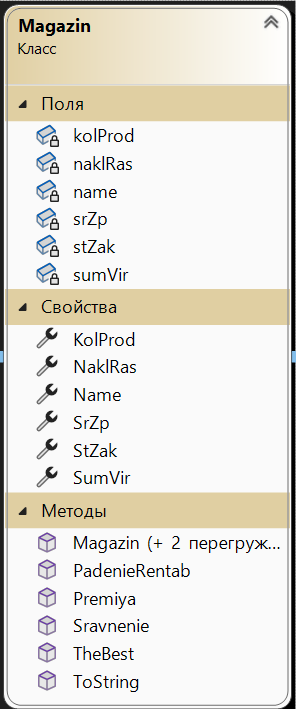
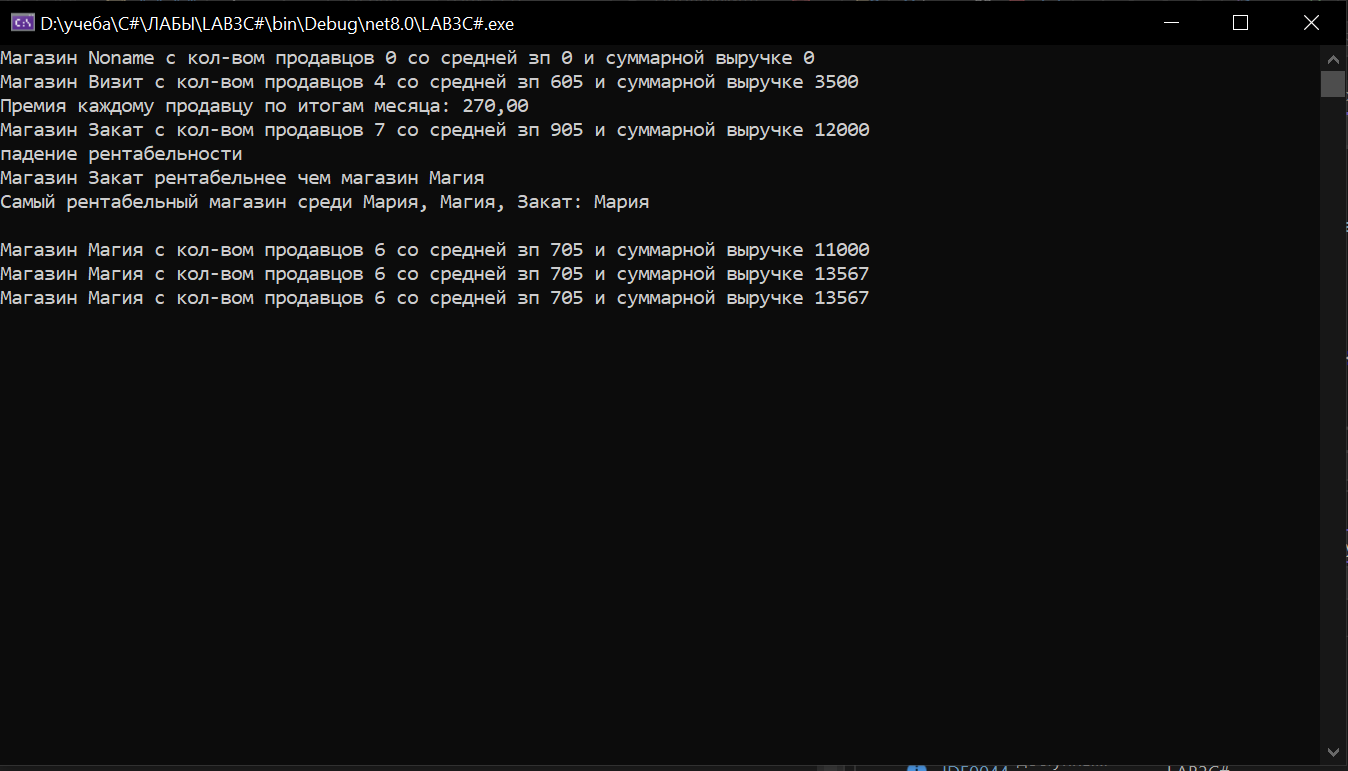
}

return str;

}

}

}



Вариант 10 Зарплата

using LAB3C\_;

Zarplata zarplata = new Zarplata();

Console.WriteLine(zarplata);

Zarplata zarplata1 = new Zarplata("Серов И.В.", "Директор", 4000, 20, 1.5);

Console.WriteLine(zarplata1);

Zarplata zarplata2 = new Zarplata("Серов И.В.", "Директор", 4000, 20, 1.5, 10, 4);

Console.WriteLine($"Зарплата {zarplata2.Name}: {zarplata2.ZarplataMes():0.00}");

Zarplata zarplata3 = new Zarplata("Позняк В.В.", "Программист", 2000, 10, 1.0, 10, 24);

Console.WriteLine(zarplata3);

zarplata3.Hronic();

Console.WriteLine(zarplata3);

Zarplata zarplata4 = new Zarplata("Некрасов И.Е.", "Главный бухгалтер", 3500, 20, 1.75, 20, 3);

if(zarplata4.Sravnenie(zarplata2)) Console.WriteLine($"Сотрудник {zarplata4.Name} имеет более высокую зп чем {zarplata2.Name}");

else Console.WriteLine($"Сотрудник {zarplata2.Name} имеет более высокую зп чем {zarplata4.Name}");

Console.WriteLine($"Самый высокооплачиваемый сотрудник среди {zarplata4.Name}, {zarplata3.Name}, {zarplata2.Name}: {Zarplata.TheBest(zarplata4,zarplata3,zarplata2).Name}");

Console.WriteLine();

zarplata4 = zarplata3;

Console.WriteLine(zarplata3);

zarplata4.Stavka = 1.5;

Console.WriteLine(zarplata4);

Console.WriteLine(zarplata3);

Console.ReadKey();

namespace LAB3C\_

{

public class Zarplata

{

private string name;

public string dolzhnost;

public double oklad;

private double stavka; //ставка (от 0,25 до 1,75)

public int stazh;

private int kolCh; //к-во часов почасовой оплаты ( от 0 до 240)

public int kolDnBol;//к-во дней, оплаченных по больничному листу

public Zarplata():this("Noname", "Nodolzhnost",0,0,0)

{

}

public Zarplata(string name, string dolzhnost, double oklad, int stazh, double stavka):this(name, dolzhnost,oklad,stazh,stavka,0,0)

{

}

public Zarplata(string name, string dolzhnost, double oklad, int stazh, double stavka, int kolCh, int kolDnBol)

{

this.name = name;

this.dolzhnost = dolzhnost;

this.oklad = oklad;

this.stazh = stazh;

this.kolDnBol = kolDnBol;

Stavka = stavka;

KolCh = kolCh;

}

public string Name { get => name;}

public double Stavka

{

get => stavka;

set

{

if(value>=0.25 & value <=1.75)

stavka = value;

}

}

public int KolCh

{

get => kolCh;

set

{

if(value>=0 & value <=240)

kolCh = value;

}

}

// метод, вычисляющий зарплату за месяц:

//• полная ставка \* оклад

//• если стаж >10 лет +10%, если > 15 лет +20%

//• почасовая – 10 за час

//• больничные оплачиваются фиксированно 25 в день

public double ZarplataMes()

{

if(stazh>=10 & stazh <= 15)

{

return (stavka \* oklad) \* 1.1 + kolCh \* 10 + kolDnBol \* 25;

}

else if (stazh > 15)

{

return (stavka \* oklad) \* 1.2 + kolCh \* 10 + kolDnBol \* 25;

}

else

{

return (stavka \* oklad) + kolCh \* 10 + kolDnBol \* 25;

}

}

public void Hronic() //метод, определяющий хронически болеющего сотрудника – если больше 20 дней в месяц – уволить(оклад= 0)

{

if (kolDnBol > 20)

{

oklad = 0;

}

}

public bool Sravnenie(Zarplata other) //метод, определяющий более высокоплачиваемого сотрудника из двух

//(возвращает true, если текущий более высокооплачиваемый)

{

return ZarplataMes() > other.ZarplataMes();

}

//статический метод, определяющий более высокоплачиваемого

//сотрудника из трех(возвращает объект, чья зарплата выше)

public static Zarplata TheBest(Zarplata zp1, Zarplata zp2, Zarplata zp3)

{

if(zp1.ZarplataMes()>zp2.ZarplataMes() & zp1.ZarplataMes()>zp3.ZarplataMes()) return zp1;

if (zp2.ZarplataMes() > zp3.ZarplataMes()) return zp2;

return zp3;

}

public override string? ToString()

{

string str = "";

if(name != null & dolzhnost!=null)

{

str += $"{name}, находящийся на должности {dolzhnost} с окладом {oklad} и ставкой {stavka} со стажем {stazh}";

}

return str;

}

}

}

