**2 Классы**

Задание 1. Создать класс А с целочисленными полями а и b и двумя методами согласно варианту. Внутри класса реализовать конструктор для инициализации a и b. Создать объект класса и продемонстрировать работу со всеми элементами класса.

Листинг программы:

namespace task1

{

using System;

class A

{

private double a;

private int b;

public A(double a, int b)

{

this.a = a;

this.b = b;

}

public double CalculateDivide()

{

double result = (b - 2) / (2 - a);

return result;

}

public double CalculateSqrt10()

{

double result = a / 10;

return result;

}

}

internal class Program

{

static void Main()

{

A obj = new A(1, 8);

Console.WriteLine("Значение выражения (b-2)/(2-a): " + obj.CalculateDivide());

Console.WriteLine("Значение a в степени 1/10: " + obj.CalculateSqrt10());

}

}

}

Таблица 2.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 1; 8 | 6;  0,1; |

Анализ результатов:



Рисунок 2.1 – Результат работы программы

Задание 2. Построить иерархию классов в соответствии с вариантом задания: Корабль, пароход, парусник, корвет.

Листинг программы:

namespace task2

{

using System;

class Ship

{

public string Name { get; set; }

public int YearBuilt { get; set; }

public Ship(string name, int yearBuilt)

{

Name = name;

YearBuilt = yearBuilt;

}

public virtual void PrintInfo()

{

Console.WriteLine($"Корабль: {Name}, Год выпуска: {YearBuilt}");

}

}

class Steamship : Ship

{

public int SteamPower { get; set; }

public Steamship(string name, int yearBuilt, int steamPower) : base(name, yearBuilt)

{

SteamPower = steamPower;

}

public override void PrintInfo()

{

base.PrintInfo();

Console.WriteLine($"Мощность : {SteamPower}");

}

}

class SailingShip : Ship

{

public int MastCount { get; set; }

public SailingShip(string name, int yearBuilt, int mastCount) : base(name, yearBuilt)

{

MastCount = mastCount;

}

public override void PrintInfo()

{

base.PrintInfo();

Console.WriteLine($"Колличество мачт : {MastCount}");

}

}

class Corvette : SailingShip

{

public int Cannons { get; set; }

public Corvette(string name, int yearBuilt, int mastCount, int cannons) : base(name, yearBuilt, mastCount)

{

Cannons = cannons;

}

public override void PrintInfo()

{

base.PrintInfo();

Console.WriteLine($"Колличество пушек: {Cannons}");

}

}

class Program

{

static void Main()

{

Ship ship = new Ship("Венера", 1800);

ship.PrintInfo();

Steamship steamship = new Steamship("Титаник", 1912, 1000);

steamship.PrintInfo();

SailingShip sailingShip = new SailingShip("Клевер", 1620, 3);

sailingShip.PrintInfo();

Corvette corvette = new Corvette("Победа", 1765, 3, 104);

corvette.PrintInfo();

}

}

}

Таблица 2.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
|  | Корабль: Венера, Год выпуска: 1800  Корабль: Титаник, Год выпуска: 1912  Мощность : 1000  Корабль: Клевер, Год выпуска: 1620  Колличество мачт : 3  Корабль: Победа, Год выпуска: 1765  Колличество мачт : 3  Колличество пушек: 104 |

Анализ результатов:

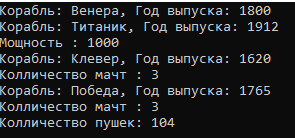


Рисунок 2.2 – Результат работы программы

Задание 3. Описать класс «комната», содержащий сведения о метраже, высоте потолков и количестве окон. Предусмотреть инициализацию с проверкой допустимости значений полей. Описать методы вычисления площади и объема комнаты и свойства для получения состояния объекта.

Написать программу, демонстрирующую все разработанные элементы класса.

Листинг программы:

namespace dopZadanieTask1

{

using System;

using System.Threading.Channels;

class Room

{

private double area;

private double height;

private double windowCount;

public double Area

{

get { return area; }

set

{

if (value > 0)

area = value;

else

throw new ArgumentException("Метраж должен быть больше чем 0.");

}

}

public double Height

{

get { return height; }

set

{

if (value > 0)

height = value;

else

throw new ArgumentException("Высота комнаты должна быть больше чем 0.");

}

}

public double WindowCount

{

get { return windowCount; }

set

{

if (value >= 0)

windowCount = value;

else

throw new ArgumentException("Колличество окон не может быть отрицательным.");

}

}

public Room(double area, double height, double windowCount)

{

Area = area;

Height = height;

WindowCount = windowCount;

}

public double CalculateVolume()

{

return Area \* Height;

}

}

class Program

{

static void Main()

{

Console.WriteLine("Введите метраж комнаты: ");

double a = Convert.ToDouble(Console.ReadLine());

Console.WriteLine("Введите высоту комнаты: ");

double b = Convert.ToDouble(Console.ReadLine());

Console.WriteLine("Введите колличество окон в комнате: ");

double c = Convert.ToDouble(Console.ReadLine());

Room room = new Room(a, b, c);

Console.WriteLine($"Площадь комнаты: {room.Area} квадратных метров");

Console.WriteLine($"Высота комнаты: {room.Height} метров");

Console.WriteLine($"Колличество окон: {room.WindowCount}");

Console.WriteLine($"Обьем комнаты: {room.CalculateVolume()} кубометров");

}

}

}

Таблица 2.3 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 30;  2;  5; | Площадь комнаты: 30 квадратных метров  Высота комнаты: 2 метров  Колличество окон: 5  Обьем комнаты: 60 кубометров |

Анализ результатов:



Рисунок 2.3 – Результат работы программы