**Лабораторная работа № 1**

**Тема:** Разработка и отладка программ демонстрации полиморфизма

**Цель:** Получить практические навыки разработки и отладки программ демонстрации полиморфизма.

**Условие задачи по вашему варианту:**

**Вариант 1**

Создать абстрактный класс Transport (транспортное средство) и на его основе два производных класса Plain (самолёт) и Ship (корабль). Классы должны иметь возможность получать параметры (название, количество мест, скорость, год выпуска, для самолёта – высота, для корабля – порт приписки) средств передвижения, а также вывода на экран сведений о транспортном средстве. Организовать вывод на экран сведений из массива обо всех самолётах и кораблях, вместимость которых больше заданной.

**Диаграмму классов с указанием полей и их типов, методов каждого класса и отношений классов:**



**Текст всех программных модулей с комментариями:**

**Program.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp1

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Введите кол-во мест для сравнения: \n");

int p;

p = Int32.Parse(Console.ReadLine());

Transport transport = new Transport(120, 3, "Volvo", "21.10.2003");

Ship ship = new Ship(40, 200, "Aurora", "10.07.1995", "New York");

Aircraft aircraft = new Aircraft(800, 200, "Aeroflot", "05.12.2017", 11000);

if (transport.Seats > p)

transport.Show();

if (ship.Seats > p)

ship.Show();

if (aircraft.Seats > p)

aircraft.Show();

Console.ReadLine();

}

}

}

**Transport.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp1

{

class Transport

{

// Инициализация перменных

int speed;

int seats;

string name;

string date;

public Transport() //Конструктор класса без аргументов

{

this.speed = 0;

this.seats = 0;

this.name = "-";

this.date = "-";

}

public Transport(int speed, int seats, string name, string date) //Конструктор класса

{

this.speed = speed;

this.seats = seats;

this.name = name;

this.date = date;

}

virtual public void Show() //Метод для вывода на значений на экран

{

Console.WriteLine( "\nМодель: {2}\n" +

"Кол-во мест: {1}\n" +

"Скорость: {0}\n" +

"Дата производства: {3}", speed, seats, name, date);

}

public int Seats { //Возвращение seats для дальнейшей работы с ним

get {

return seats;

}

}

}

}

**Airctaft.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp1

{

class Aircraft : Transport //Производный класс

{

int height;

public Aircraft(int speed, int seats, string name, string date, int height) : base (speed,seats, name, date) { // Конструктор класса

this.height = height;

}

override public void Show() // Переопределенный метод Show()

{

base.Show();

Console.Write("Высота полета: {0}\n", height);

}

}

}

**Ship.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp1

{

class Ship : Transport //Производный класс

{

string port;

public Ship(int speed, int seats, string name, string date, string port) : base(speed, seats, name, date) // Конструктор класса

{

this.port = port;

}

override public void Show() //Переопределенный метод Show()

{

base.Show();

Console.Write("Порт: {0}\n", port);

}

}

}

**Скриншот выполнения кода:**

