4 группа

```
enum Statuses
{
    ready,
    fly,
    stop
}
abstract class Transport
{

}

interface IAirable
{
    public void Fly(StreamWriter sw);
    public void Check();
}

interface IAirHostess
{
    public void Check(StreamWriter sw);
}

class Air : Transport, IAirable, IAirHostess
{
    public int Speed { get; set; }
    public int CountOfPassanger { get; set; }

    public Statuses Status;

    public void Fly(StreamWriter sw)
    {
        if (Status == Statuses.fly)
        {
            Console.WriteLine("Я в полете");
            sw.WriteLine("Я в полете");
        }

        else
            throw new Exception();
    }

    public void Check()
    {
        if (Status == Statuses.ready && CountOfPassanger > 0 && Speed > 0)
            Status = Statuses.fly;
        else if (CountOfPassanger == 0 && Speed == 0)
            Status = Statuses.stop;
        else if (CountOfPassanger > 0 && Speed == 0)
            Status = Statuses.ready;
    }

    void IAirHostess.Check(StreamWriter sw)
    {
        if (CountOfPassanger > 20 && CountOfPassanger < 100)
        {
            Console.WriteLine("Ready");
            sw.WriteLine("Ready");
        }
    }
}
```

```
class Program
{
    public static void Main()
    {
        using (StreamWriter sw = new StreamWriter("Text.txt"))
        {
            int[] arr = new int[4];
            int[,] recArr = new int[,] { { 1, 2, 3 }, { 1, 2, 3 } };

            Air air = new Air();
            air.Speed = 2222;
            air.CountOfPassanger = 12;
            air.Check();
            air.Fly(sw);

            IAirHostess airHostess = air;
            airHostess.Check(sw);


            Air air2 = new Air();
            air2.Speed = 2222;
            air2.CountOfPassanger = 22;
            air2.Check();
            air2.Fly(sw);


            Air air3 = new Air();
            air3.Speed = 2222;
            air3.CountOfPassanger = 22;
            air3.Check();
            air3.Fly(sw);

            Air air4 = new Air();
            air4.Speed = 2222;
            air4.CountOfPassanger = 22;
            air4.Check();
            air4.Fly(sw);

            Air air5 = new Air();
            air5.Speed = 22;
            air5.CountOfPassanger = 12;
            air5.Check();
            air5.Fly(sw);


            List<Air> list = new List<Air>();
            list.Add(air);
            list.Add(air2);
            list.Add(air3);
            list.Add(air4);
            list.Add(air5);

            var linqList = from p in list
                           where p.Status == Statuses.fly
                           select p;
            Console.WriteLine(linqList.Count());

            var Avarage = list.Average(p => p.Speed);
            Console.WriteLine(Avarage);

        }
    }
}
```

Второй

```
using System.Text.Json;

interface INumber
{
    public int Number { get; set; }
}

class MoneyEx: Exception
{
    public MoneyEx(string message): base(message) { }
}

class NoBillInWallet : Exception
{
    public NoBillInWallet(string message) : base(message) { }
}

class Bill : INumber
{
    private int _number;
    public int Number
    {
        get
        {
            return _number;
        }
        set
        {
            if (value == 50 || value == 100 || value == 5 || value == 10 || value == 50 || value == 20)
                _number = value;
        }
    }
}

class Wallet<T> where T : class
{
    public List<Bill> _billList = new List<Bill> { };
    private int _sum = 0;
    public string date;

    private static readonly Wallet<T>? instance = new Wallet<T>();

    private Wallet()
    {
        date = System.DateTime.Now.TimeOfDay.ToString();
    }

    public static Wallet<T>? GetInstance()
    {
        return instance;
    }


    public void Add(Bill bill)
    {
        _billList.Add(bill);
        _sum += bill.Number;
        if (_sum > 200)
            throw new MoneyEx("To mush groshey");
    }

    public void Remove()
    {
        Bill bill = _billList[0];
        foreach (var b in _billList)
        {
            if (bill.Number >= b.Number)
                bill = b;
        }
        _billList.Remove(bill);
        _sum -= bill.Number;
    }

}

class Programm
{
    public static void Main()
    {
        Bill bill1 = new Bill();
        bill1.Number = 10;

        Bill bill2 = new Bill();
        bill2.Number = 10;

        Bill bill3 = new Bill();
        bill3.Number = 10;

        Bill bill4 = new Bill();
        bill4.Number = 50;

        Bill bill5 = new Bill();
        bill5.Number = 20;

        Bill bill6 = new Bill();
        bill6.Number = 10;

        Wallet<Bill> wallet = Wallet<Bill>.GetInstance();
        wallet.Add(bill1);
        wallet.Add(bill2);
        wallet.Add(bill3);
        wallet.Add(bill4);
        wallet.Add(bill5);
        wallet.Add(bill6);

        (new Thread(() =>
        {
            Wallet<Bill> tt = Wallet<Bill>.GetInstance();
            Console.WriteLine(tt.date);
        })).Start();

        Console.WriteLine(wallet.date);

        /*        foreach (var x in wallet._billList)
                {
                    Console.WriteLine(x.Number);
                }

                Console.WriteLine("------------------------");
                wallet.Remove();

                foreach (var x in wallet._billList)
                {
                    Console.WriteLine(x.Number);
                }
                Console.WriteLine("------------------------");
                wallet.Remove();

                foreach (var x in wallet._billList)
                {
                    Console.WriteLine(x.Number);
                }
                Console.WriteLine("------------------------");
                wallet.Remove();

                foreach (var x in wallet._billList)
                {
                    Console.WriteLine(x.Number);
                }
                Console.WriteLine("------------------------");
                wallet.Remove();

                foreach (var x in wallet._billList)
                {
                    Console.WriteLine(x.Number);
                }*/

        var linqWallet = from p in wallet._billList
                         group p by p.Number into s
                         select s;

        foreach (var x in linqWallet)
        {
            Console.WriteLine($"{x.Key} - {x.Count()}");
        }

        string json = JsonSerializer.Serialize(wallet._billList);
        Console.WriteLine(json);

    }
}
```