Tasks

1. делегаты/события +
2. дженерики +
3. сериализация +
4. стэкфрейм
5. интерфейсы +
6. yield+
7. Напишите программу, в которой определено событие и выполнена подписка на него. Продемонстрируйте смысл событий.

namespace Ex1

{

class Publisher

{

public event EventHandler<string> Notify;

public void DoSomething()

=> Notify.Invoke(this, "Some happened in Publisher");

}

}

namespace Ex1

{

class Subscriber

{

public void Subscribe(ref Publisher publisher) =>

publisher.Notify += Send;

public void Unsubscribe(ref Publisher publisher) =>

publisher.Notify += Send;

public void Send(object sender, string message) =>

Console.WriteLine($"Отправить сообщение от подписчика: {message}");

}

}

namespace Ex1

{

class Program

{

//1) Напишите программу,

//в которой определено событие и выполнена подписка на него.Продемонстрируйте смысл событий.

static void Main(string[] argv)

{

Publisher publisher = new Publisher();

Subscriber subscriber = new Subscriber();

Subscriber subscriber1 = new Subscriber();

subscriber.Subscribe(ref publisher);

subscriber1.Subscribe(ref publisher);

publisher.DoSomething();

subscriber1.Unsubscribe(ref publisher);

publisher.DoSomething();

}

}

}

1. Напишите метод-расширения для десериализации generic – типа +-
2. Напишите метод-расширения для сериализации generic-типа
3. Напишите программу, получающую доступ ко всем кадрам стека и выводящие их содержимое на консоль +-
4. Напишите программу, содержащий собственный generic-тип и пример его использования +

class GenericClass<T>

{

public T Value { get; set; }

public GenericClass(T Value)

{

this.Value = Value;

}

}

class Program

{

static void Main(string[] args)

{

GenericClass<int> first = new GenericClass<int>(1);

int b = 1;

int c = first.Value + b;

GenericClass<string> second = new GenericClass<string>("Няя");

string str = "Не";

//if (second is string)

//{

Console.WriteLine($"{second.Value + str}");

//}

}

}

1. Напишите программу, содержащую пример применения итератора yield +

namespace Ex6

{

class Program

{

static void Main(string[] argv)

{

Console.WriteLine("Hello, World!");

foreach(int num in GetNums())

{

Console.WriteLine(num);

}

}

public static IEnumerable<int> GetNums()

{

for(int i = 0; i < 10; i++)

{

yield return i;

}

}

}

}

1. Напишите программу, считывающую с клавиатуры цифры и сохраняющие их в файл. Продемонстрируйте чтение из этого файла.

using System;

using System.IO;

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Введи набор символов");

string input = "was ist\ndas"; // Ваш ввод

string path = "data.txt";

// Запись

using (StreamWriter sw = new StreamWriter(path))

{

sw.Write(input);

}

// Чтение из файла

using (StreamReader sr = new StreamReader(path))

{

string line = sr.ReadToEnd();

Console.WriteLine("Прочитано из файла:");

Console.WriteLine(line);

}

Console.ReadLine();

}

}

1. Напишите программу, реализующую шаблон проектирования «cтратегия” (конкретный класс может не знать реализацию методов других классов, которые использует через интерфейс) +

namespace Ex8

{

interface ICar

{

public void Move();

}

}

namespace Ex8

{

class DieselCar : ICar

{

public void Move()

{

Console.WriteLine("Движется дизельная машина");

}

}

}

namespace Ex8

{

class ElectroCar : ICar

{

public void Move()

{

Console.WriteLine("Движется Электромобиль");

}

}

}

namespace Ex8

{

class Car

{

public string Name { get; set; }

public Car(string Name) => this.Name = Name;

public ICar TypeCar { get; set; }

public void Move()

{

TypeCar.Move();

}

}

}

namespace Ex8

{

class Program

{

static void Main(string[] args)

{

Car car = new Car("Volkswagen");

car.TypeCar = new DieselCar();

car.TypeCar.Move();

}

}

}

1. Напишите программу, реализующую шаблон проектирования «наблюдатель” +

namespace Ex9

{

interface IObservable

{

public void Subscribe(IObserver observer);

public void UnSubscribe(IObserver observer);

public void Notify(double Price);

}

}

namespace Ex9

{

interface IObserver

{

public void Update(double Price);

}

}

namespace Ex9

{

//Subscriber

class Observable : IObservable

{

//список издательств

//подписчик подписывается на разные издательства

private List<IObserver> observers = new List<IObserver>();

public void Subscribe(IObserver obs)

{

Console.WriteLine($"подпика на издательство ");

observers.Add(obs);

}

public void UnSubscribe(IObserver obs)

{

observers.Remove(obs);

}

public void Notify(double Price)

{

foreach (var obs in observers)

{

obs.Update(Price);

}

}

}

}

namespace Ex9

{

//Publisher

class Observer : IObserver

{

private double Price {get; set; }

public Observer(double Price)

{

this.Price = Price;

Console.WriteLine($"Цена на книгу изначально = {this.Price}");

}

public void Update(double Price)

{

this.Price = Price;

Console.WriteLine($"Цена на книгу обновилась = {this.Price}");

}

}

}

namespace Ex9

{

class Program

{

static void Main(string[] argv)

{

Observer Publisher = new Observer(1488);

Observable Subscriber0 = new Observable();

Observable Subscriber1 = new Observable();

// Subscribe the observer to the observable

Subscriber0.Subscribe(Publisher);

Subscriber1.Subscribe(Publisher);

}

}

}

1. Напишите программу, реализующую шаблон проектирования «builder”
2. Напишите программу, реализующую декларативную атрибутивную валидацию с использованием рефлексии.
3. Напишите программу с плагинной архитектурой.
4. Напишите программу, использующие флаговые перечисления. +

namespace Ex7

{

[Flags]

enum Days

{

Monday = 1,

Tuesday = 2,

Wednesday = 3,

Thursday = 4,

Friday = 5,

Saturday = 6,

Sunday = 7

}

}

class Program

{

static void Main(string[] args)

{

Days days = Days.Monday | Days.Tuesday;

Console.WriteLine("Рабочие дни: " + days);

if (days == Days.Wednesday)

{

Console.WriteLine("Среда - рабочий день");

}

//добавление

days = days | Days.Saturday;

Console.WriteLine("День недели " + days);

//удаление

days = days & ~Days.Saturday;

Console.WriteLine("День недели " + days);

}

}

Напишите программу, использующую yield return. +

namespace Ex6

{

class Program

{

static void Main(string[] argv)

{

Console.WriteLine("Hello, World!");

foreach(int num in GetNums())

{

Console.WriteLine(num);

}

}

public static IEnumerable<int> GetNums()

{

for(int i = 0; i < 10; i++)

{

yield return i;

}

}

}

}

1. Напишите программу, предоставляющее REST-API и документацию посредством swagger, для получения списка курсов валют, хранящегося в xml-файле в файловой системе. Результат должен отдаваться на клиент как json. Путь к файлу должен быть указан в конфигурации appSettings.json
2. REST-API = ASP.NET (Шаблон WebAPI)
3. Swagger =  Swashbuckle .NET Core  (не менять)
4. Xml, json
5. Нужно поменять только класс с данными, файл контроллера и добавить Path в appsettings.json
6. Напишите программу, предоставляющее REST-API и документацию посредством swagger, для получения списка студентов, хранящегося в json-файле в файловой системе. Путь к файлу должен быть указан в конфигурации appSettings.json.
7. Напишите программу, предоставляющее REST-API и документацию посредством swagger, для получения списка стран, хранящегося в json-файле в файловой системе. Путь к файлу должен быть указан в конфигурации appSettings.json. Названия стран должны быть представлены на разных языках.
8. Напишите программу, предоставляющее REST-API и документацию посредством swagger, для получения списка курсов образовательной организации, хранящегося в json-файле в файловой системе. Путь к файлу должен быть указан в конфигурации appSettings.json.
9. Напишите программу, предоставляющее REST-API и документацию посредством swagger, для получения списка строк, хранящегося в xml-файле в файловой системе. Путь к файлу должен быть указан в конфигурации appSettings.json.
10. Напишите программу, предоставляющее REST-API и документацию посредством swagger, для получения списка доступных для покупки пицц, хранящегося в json-файле в файловой системе. Путь к файлу должен быть указан в конфигурации appSettings.json.
11. Напишите программу, предоставляющее REST-API и документацию посредством swagger, для получения списка доступных для покупки пицц, хранящегося в базе данных
12. Напишите программу, предоставляющее REST-API и документацию посредством swagger, для получения сведений о нагрузке на компьютер в виде json- ответа. Метод должен принимать на вход логин и пароль, после чего проводить сверку учетных данных с указанными ранее в конфигурационном файле. Допустимо хранить только хэш от пароля.
13. Напишите программу, предоставляющее REST-API и документацию посредством swagger, для получения сведений о параметрах подключившегося клиента (User Agent)
14. Напишите программу, предоставляющее REST-API и документацию посредством swagger, для получения списка доступного съемного жилья по коду страны, переданного в качестве get - параметра, Список хранится в json-файле в файловой системе. Путь к файлу должен быть указан в конфигурации appSettings.json.
15. Напишите программу, предоставляющее REST-API и документацию посредством swagger, для получения списка курсов образовательной, хранящегося в json-файле в файловой системе. Путь к файлу должен быть указан в конфигурации appSettings.json.