**Функция myFunction возвращает кортеж**

void zd5(int a, int b, char c)

{

string str = "hello world";

int[] array = new int[5];

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

array[i] = i;

(int, int, char) myFunction(int[] arrayOfInt, string string1)

{

(int, int, char) tuple3 = (arrayOfInt.Min(), arrayOfInt.Max(), string1.First());

return tuple3;

}

Console.WriteLine(myFunction(array, str));

}

**Функиция переопределения**

public static Set<T> operator -(Set<T> list, T item) //удаление элемента

{

list.RemoveEl(item);

return list;

}

Set<int> Arr1 = new Set<int>();

Arr1 -= 5;

**Делегаты и события**

class Director

{

public delegate void Raise(int add);

public event Raise Raising;

public delegate void Penalty();

public event Penalty Penalizing;

public void Add(int add) => Raising(add);

public void Minus() => Penalizing?.Invoke();

}

Director Dr\_1 = new Director();

Dr\_1.Raising += tokar1.AddZp;

Dr\_1.Raising += tokar2.AddZp;

Dr\_1.Raising += tokar3.AddZp;

class Program

{

    delegate void Message(); // 1. Объявляем делегат

    static void Main(string[] args)

    {

        Message mes; // 2. Создаем переменную делегата

        if (DateTime.Now.Hour < 12)

        {

            mes = GoodMorning; // 3. Присваиваем этой переменной адрес метода

        }

        else

        {

            mes = GoodEvening;

        }

        mes(); // 4. Вызываем метод

        Console.ReadKey();

    }

    private static void GoodMorning()

    {

        Console.WriteLine("Good Morning");

    }

    private static void GoodEvening()

    {

        Console.WriteLine("Good Evening");

    }

}

string testString = "Делегат, Func возвращает результат, действия и может принимать, параметры!";

Func<string, string> func;

func = ChangeString.DeletePunctSigns; //удаление знаков препинания

func += ChangeString.ToUpperCase; //перевод строки в верхний регистр

func += ChangeString.DeleteUnnecessarySpace; //уделение лишних пробелов

**LINQ**

List<Customer> cust1 = (from i in customers

orderby i //по возрастанию

select i).ToList();

List<Customer> cust4\_1 = (from i in customers

where i.Name == "jon" //условие

select i).ToList();

List<Customer> cust4\_3 = (from i in customers

orderby i.Name descending //по убыванию

select i).ToList();

var result = from i in summer\_winterMonthes

join m in monthes on i equals m //join

select new {i};

**Cереализация**

BinaryFormatter formatter = new BinaryFormatter();   
XmlSerializer xmlSerializer = new XmlSerializer(typeof(Word));   
DataContractJsonSerializer jsonSerializer = new DataContractJsonSerializer(typeof(Word));  
  
Console.WriteLine("Binary formater");   
using (FileStream fs = new FileStream("Word.dat", FileMode.OpenOrCreate))   
{   
formatter.Serialize(fs, text);   
}  
Console.WriteLine("XML Serializer");   
using (FileStream fs = new FileStream("xml.xml", FileMode.OpenOrCreate))   
{   
xmlSerializer.Serialize(fs, text);   
}  
Console.WriteLine("Json Serializer");   
using (FileStream fs = new FileStream("json.json", FileMode.OpenOrCreate))   
{   
jsonSerializer.WriteObject(fs, text);   
}

**ПОТОКИ**

object locker = new object();

if (File.Exists("OddEvenConsistently.txt"))

{

File.Delete("OddEvenConsistently.txt");

}

Thread OddThread = new Thread(new ThreadStart(PrintOdd));

Thread EvenThread = new Thread(new ThreadStart(PrintEven));

OddThread.Start();

EvenThread.Start();

void PrintOdd()

{

lock (locker)

{

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

{

if (i % 2 != 0)

{

Console.WriteLine(i + " odd");

WriteResultToFile(i);

Thread.Sleep(250);

}

}

}

**ТАSК**

Task<int> task1 = new Task<int>(Mt1);

Task<int> task2 = new Task<int>(Mt2);

Task<int> task3 = new Task<int>(Mt3);

await task1.ContinueWith((t1) => Console.WriteLine($"Задача 1 завершена. Результат: {t1.Result}"));

await task2.ContinueWith((t2) => Console.WriteLine($"Задача 2 завершена. Результат: {t2.Result}"));

await task3.ContinueWith((t3) => Console.WriteLine($"Задача 3 завершена. Результат: {t3.Result}"));

task1.Start();

task2.Start();

task3.Start();

var awaiter = task1.GetAwaiter();

awaiter.OnCompleted(() => Console.WriteLine($"Task1 completed. Result: {awaiter.GetResult()}"));

Реализация интерфейса IEnumerable

public IEnumerator GetEnumerator()

{

return Model.GetEnumerator();

}

**Реализация IEnumerator**

class WeekEnumerator : IEnumerator

    {

        string[] days;

        int position = -1;

 public object Current

        {

            get

            {

                if (position == -1 || position >= days.Length)

                    throw new InvalidOperationException();

                return days[position];

            }

        }

        public bool MoveNext()

        {

            if(position < days.Length - 1)

            {

                position++;

                return true;

            }

            else

                return false;

        }

        public void Reset()

        {

            position = -1;

        }

    }

**Реализация IComparer**

interface IComparer

{

    int Compare(object o1, object o2);

}

class PeopleComparer : IComparer<Person>

{

    public int Compare(Person p1, Person p2)

    {

        if (p1.Name.Length > p2.Name.Length)

            return 1;

        else if (p1.Name.Length < p2.Name.Length)

            return -1;

        else

            return 0;

    }

}

**Реализация IComparable**

class Person : IComparable

{

    public string Name { get; set; }

    public int Age { get; set; }

    public int CompareTo(object o)

    {

        Person p = o as Person;

        if (p != null)

            return this.Name.CompareTo(p.Name);

        else

            throw new Exception("Невозможно сравнить два объекта");

    }

}

**ИНДЕКСАТОРЫ**

возвращаемый\_тип this [Тип параметр1, ...]

{

    get { ... }

    set { ... }

}

**СЕРЕАЛИЗАЦИЯ**

if (format == "SOAP")

{

SoapFormatter formatter = new SoapFormatter();

formatter.Serialize(file, obj);

}

else if (format == "BINARY")

{

BinaryFormatter formatter = new BinaryFormatter();

formatter.Serialize(file, obj);

}

else if (format == "XML")

{

XmlSerializer xmlF = new XmlSerializer(typeof(T));

xmlF.Serialize(file, obj);

ДЕСЕРЕЛИЗАЦИЯ

}

if (format == "SOAP")

{

SoapFormatter formatter = new SoapFormatter();

obj = formatter.Deserialize(file);

}

else if (format == "BINARY")

{

BinaryFormatter formatter = new BinaryFormatter();

obj = formatter.Deserialize(file);

}

else if (format == "XML")

{

XmlSerializer xmlF = new XmlSerializer(typeof(T));

obj = xmlF.Deserialize(file);

}