Методы последовательностей

`IEnumerable<T>`– последовательность элементов типа T.

`List<T>, array of T, LinkedList<T>, HashSet<T>, SortedSet<T>` являются последовательностями.

Используемые типы функций

`Func<T,Res>`	`Action<T>`	`Predicate<T>`
`Func2<T1,T2,Res>`	`Action2<T1,T2>`	`Predicate2<T1,T2>`

Генерация последовательностей

… → **последовательность**

```
Range(a,b: integer): sequence of integer
Range(a,b,step: integer): sequence of integer
Range(c1,c2: char): sequence of char
Range(a,b: real; n: integer): sequence of real
SeqRandom(n,a,b: integer): array of integer;
SeqRandomReal(n: integer; a,b: real): array of real;
Seq(params a: array of T): array of T;
SeqGen(count: integer; first: T; next: T -> T): array of T;
SeqGen(count: integer; first,second: T; next: (T,T) -> T): array of T;
SeqWhile(first: T; next: T -> T; pred: T -> boolean): sequence of T;
SeqWhile(first,second: T; next: (T,T) -> T; pred: T -> boolean): sequence of T;
SeqFill(count: integer; x: T): sequence of T;
SeqGen(count: integer; f: integer -> T): sequence of T;
```

Группа 1. Вывод последовательностей

```
Print(delim: string := ' '): sequence of T;
Println(delim: string := ' '): sequence of T;
```

Группа 2. Фильтрация, инвертирование

sequence of T → sequence of T

```
Where(T -> boolean)
Where((T, integer) -> boolean)
TakeWhile(T -> boolean)
TakeWhile((T, integer) -> boolean)
SkipWhile(T -> boolean)
SkipWhile((T, integer) -> boolean)
Take(count)
Skip(count)
Distinct()
Reverse()
```

Группа 3. Проецирование

sequence of T → sequence of TRes

```
Select(T -> TRes)
Select((T, integer) -> TRes)
SelectMany(T -> sequence of TRes)
SelectMany((T, integer) -> sequence of TRes)
```

Группа 4. Упорядочивание

sequence of T → IOrderedEnumerable<T>

```
Sorted()
OrderBy(T -> TKey)
OrderByDescending(T -> TKey)
ThenBy(T -> TKey)
ThenByDescending(T -> TKey)
```

Группа 5. Вычисление скаляра

sequence of T → скалярный тип

```
Count([T -> boolean]): integer
```

```
Average(): double
Average(T -> числовой_тип): double
Sum(): числовой_тип
Sum(T -> числовой_тип): числовой_тип
Max(): T
Max(T -> TRes): TRes
Min(): T
Min(T -> TRes): TRes
Aggregate((T, T) -> T): T
Aggregate(TRes seed, (TRes, T) -> TRes): TRes
Aggregate(TAccum seed, (TAccum,T) -> TAccum, TAccum -> TRes): TRes
```

Группа 6. Вычисление логического значения

sequence of T → boolean

```
All(T -> boolean)
Any(T -> boolean)
Contains(T value)
SequenceEqual(sequence of T second)
```

Группа 7. Сцепление и операции над множествами

sequence of T → sequence of T

```
Concat(second: sequence of T)
Union(second: sequence of T)
Intersect(second: sequence of T)
Except(second: sequence of T)
```

Группа 8. Объединение по ключу

sequence of TOuter → sequence of TRes

```
Join(inner: sequence of TInner, TOuter -> TKey, TInner -> TKey, (TOuter,TInner) -> TRes)
GroupJoin(inner: sequence of TInner, TOuter -> TKey,
      TInner -> TKey, (TOuter, sequence of TInner) -> TRes)
Zip(second: sequence of T, (T,T)->T1): sequence of T1
```

Группа 9. Группировка по ключу

sequence of T → последовательность другого типа

```
GroupBy(T -> TKey) → IEnumerable<IGrouping<TKey, T>>
GroupBy(T -> TKey, T -> TElement) → IEnumerable<IGrouping<TKey, TElement>>
GroupBy(T -> TKey, (TKey, sequence of T) -> TRes) → sequence of TRes
GroupBy(T->TKey, T->TElement, (TKey, sequence of TElement)->TRes) → sequence of TRes
```

Группа 10. Импортирование

IEnumerable → sequence of TRes

```
OfType<TRes>()
Cast<TRes>()
```

Группа 11. Экспортирование

sequence of T → коллекция определенного типа

```
ToArray(): array of T
ToList(): List<T>
ToDictionary(T -> TKey): Dictionary<TKey, T>
ToDictionary(T -> TKey, T -> TElement): Dictionary<TKey, TElement>
AsEnumerable<T>(): sequence of T
```

Группа 12. Поэлементные операции

sequence of T → T

```
First([T -> boolean])
Last([T -> boolean])
Single([T -> boolean])
ElementAt(integer index)
```