**List.Aggregate**

یک متد کاربرد در کالکشن ها است که برای تجمیع یا کاهش مقادیر یک کالکشن کاربرد دارد.

یعنی یک collection داریم که می خواهیم همه مقادیرش را با یک عمل خاص به یک نتیجه نهایی برسانیم.

Item هر ایتم در کالکشن

Acc جمع عمل انجام شده

همچنین قابلیت مقدار دهی اولیه برای acc نیز وجود دارد.

var result = collection.Aggregate((acc, item) =>{ } );

مثال : جمع کردن اعداد.

int[] numbers = { 1, 2, 3, 4, 5 };

int sum = numbers.Aggregate((acc, n) => acc + n);

Console.WriteLine(sum); // خروجی: 15

مثال: مشخص کردن طولانی ترین رشته

string[] words = { "apple", "banana", "cherry", "watermelon" };

string longest = words.Aggregate((acc, word) =>

acc.Length > word.Length ? acc : word);

Console.WriteLine(longest); // خروجی: watermelon

مثال: استفاده از مقدار اولیه

Seed

int[] numbers = { 1, 2, 3, 4 };

int product = numbers.Aggregate(1, (acc, n) => acc \* n);

Console.WriteLine(product); // خروجی: 24

🔑 خلاصه:  
 Aggregate مثل یک حلقه است که همه‌ی عناصر مجموعه را طبق یک فرمول مشخص روی هم انباشته می‌کند تا به یک مقدار نهایی برسید.

مثال sum

int sum1 = numbers.Sum();

int sum2 = numbers.Aggregate((acc, n) => acc + n);

مثال max

int max1 = numbers.Max();

int max2 = numbers.Aggregate((acc, n) => acc > n ? acc : n);

مثال Average

double avg1 = numbers.Average();

double avg2 = numbers.Aggregate(

(acc, n) => acc + n

) / (double)numbers.Count();

به طور کلی مناسب عملیات های پچیده تر است.

string[] names = { "Ali", "Sara", "Reza" };

string result = names.Aggregate((acc, name) => acc + ", " + name);

Console.WriteLine(result);

// خروجی: Ali, Sara, Reza

string[] words = { "cat", "elephant", "dog" };

int totalLetters = words.Aggregate(0, (acc, word) => acc + word.Length);

Console.WriteLine(totalLetters);

// خروجی: 14 (3 + 8 + 3)

int n = 5;

int factorial = Enumerable.Range(1, n)

.Aggregate(1, (acc, num) => acc \* num);

Console.WriteLine(factorial);

// خروجی: 120