**TC.1.3.** Qual a finalidade dos elementos fork e join na representação de um fluxo de programa?

Dê um exemplo prático.

Fork e Join é um conceito de paralelismo do Design Pattern, onde consiste basicamente em Fork (dividir a task em partes) e Join(juntar as tasks).

Vantagem na programação com elementos fork e join é conseguir executar as tarefas separadas de forma paralela.

Um exemplo clássico é a soma de uma Array (vetor), onde o núcleo(thread) divide a array em partes e atribui para os threads (segmentos) fazerem uma soma separadamente (fork) e depois combina as somas individuais para obter a soma total da array (fazendo um Join novamente).

EXEMPLO NO CÓDIGO:

private static object lockObject = new object();

static void Main()  
{  
 int[] numbers = Enumerable.Range(1, 1000000).ToArray();  
 long totalSum = ParallelSum(numbers);  
 Console.WriteLine($"Total Sum: {totalSum}");  
}

Static long ParallelSum(int[] numbers)  
{  
 long totalSum = 0;  
 Parallel.ForEach(numbers, new ParallelOptions { MaxDegreeOfParallelism = 4 }, () => 0L, (num, loopState, localSum) =>  
 {  
 localSum += num;  
 return localSum;  
 }, localSum =>  
 {  
 lock (lockObject)  
 {  
 int threadId = Task.CurrentId ?? -1;  
 Console.WriteLine($"Thread {threadId}: Local Sum = {localSum}");  
 totalSum += localSum;  
 }  
 });  
 return totalSum;  
}

Fonte : <https://medium.com/@nirajranasinghe/design-patterns-for-concurrent-programming-fork-join-pattern-e8c619b4506b>