Bewerber Aufgabe Ökotec

Diese Aufgabe dient um die Fähigkeiten des Bewerbers im Bereich von (linearen/collection) Datenstrukturen, Erstellen von komplexen Algorithmen und allgemeinem analysieren und umsetzen von komplexen Prozessen zu bewerten.

Der Umfang der Aufgabe soll in einem überschaubaren Rahmen bleiben und in einem Zeitfenster von einer Stunde realisiert werden. In dieser Zeit ist eine Implementierung nur schwer umzusetzen, deswegen steht im Vordergrund mit Hilfe von Pseudo-Code oder einer detaillierten Beschreibung die angestrebte Umsetzung zu verfassen. Gerne bewerten wir auch eine umgesetzte Implementierung, bitte denken Sie aber daran, dass dies im vorgegeben Zeitfenster schwer zu erreichen ist.

Die Aufgabe umfasst zwei Listen aus Intervallen aus denen eine neue Liste mit Intervallen erstellt werden soll. Ein Intervall besteht aus zwei DateTime Objekten jeweils eins für einen Start- und einen Endwert. Ein Interval ist als rechtsoffenen definiert, das bedeutet, dass der Startwert im Interval enthalten ist, aber der Endwert nicht mehr Teil des Intervals ist. Die Implementierung von Interval kann in Interval.cs gefunden werden. Die gegebenen Listen beinhalten die Intervalle in sortierter aufsteigender Reihenfolge und es kann davon ausgegangen werden, dass innerhalb einer Liste sich Intervalle nicht überschneiden. Wegen der rechtsoffenen Definition ist es aber möglich, dass der Endwert eines Intervals gleich dem Startwert des darauffolgenden Intervals entspricht. Beispiele für die Listen können TestData.cs und TestClass.cs entnommen werden.

Der eigentliche Kern der Aufgabe ist es aus zwei Intervalllisten eine neue Intervallliste zu generieren, die alle Intervalle der der beiden Inputlisten enthält und gleichzeitig auch alle Intervallgrenzen (Start und End) in der neuen Liste berücksichtigt. Dies wird im Folgenden anhand von ein paar Beispielen genauer erläutert. Die Beispiele zeigen jeweils die zwei Input Intervallisten und die zu erwartende Ergebnissliste auf einer Zeitachse übereinander. Die Notation ist wie folgt zu interpretieren:

| entspricht Start und/oder End eines Intervals.

- signalisiert, dass für diesen Bereich ein Interval definiert ist.

~ signalisiert, dass für diesen Bereich kein Interval definiert ist.

Beispiel 1:

Input 1: |- - - -|- - - -|- - - -|

Input 2: |-|-|-|-|-|-|-|-|-|-|-|-|

Output: |-|-|-|-|-|-|-|-|-|-|-|-|

Beispiel 2:

Input 1: |- - - -|~ ~ ~ ~|- - - -|

Input 2: |- - - - - -|~ ~ ~ ~|- -|

Output: |- - - -|- -|~ ~|- -|- -|

Beispiel 3:

Input 1: |- - - - - - -|~|-|~ ~|- - -|

Input 2: ~ ~|- - -|-|~|- - -|~ ~|- -|

Output: |- -|- - -|-|-|-|-|-|~|-|- -|

Beachten Sie beim lösen der Aufgabe, dass die zu verarbeitenden Listen sehr groß sein können ( > 1e5). Legen Sie aus diesem Grund den Fokus auf eine ressourcensparende Umsetzung. Sollten Sie eine konkrete Implementierung realisieren wollen, können Sie die vorhandenen Tests nutzen um Ihre Implementierung zu überprüfen.

Erweiterte Aufgabe:

Passen Sie ihren Algorithmus so an, dass leicht zwischen der vorherigen Implementierung und folgender alternativen Implementierung gewechselt werden kann:

Anstatt in der Ergebnisliste jedes Input-Intervall, unabhängig davon aus welcher der zwei Input-Intervall-Listen dieses kam, zu berücksichtigen, ist es nun die Aufgabe nur Intervalle im Ergebnis zu führen in denen beide Input-Intervall-Listen auch ein Intervall haben.

Angepasstes Beispiel 3:

Input 1: |- - - - - - -|~|-|~ ~|- - -|

Input 2: ~ ~|- - -|-|~|- - -|~ ~|- -|

Output: ~ ~|- - -|-|~ ~|-|~ ~ ~|- -|

Viel Spaß und gutes Gelingen!