Datastrukturer och Algoritmer

Programmeringsuppgift 2 - Ordkedjor

Linus Forsberg  
Olle Olsson  
Jimmy Åkesson \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Källor

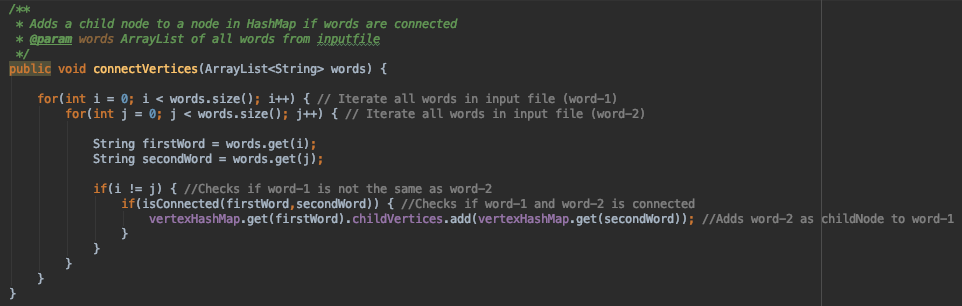
Lösningen utgår från boken och även bokens hemsidas exempel på bredden-först-sökning (BFS algoritmen). Vi tyckte det var smidigast och passande att använda oss av kurslitteraturen för denna uppgift och eftersom exemplen var tydliga och väldokumenterade gav det oss tillräckligt med kunskap för att kunna lösa uppgiften.

Tidskomplexitet

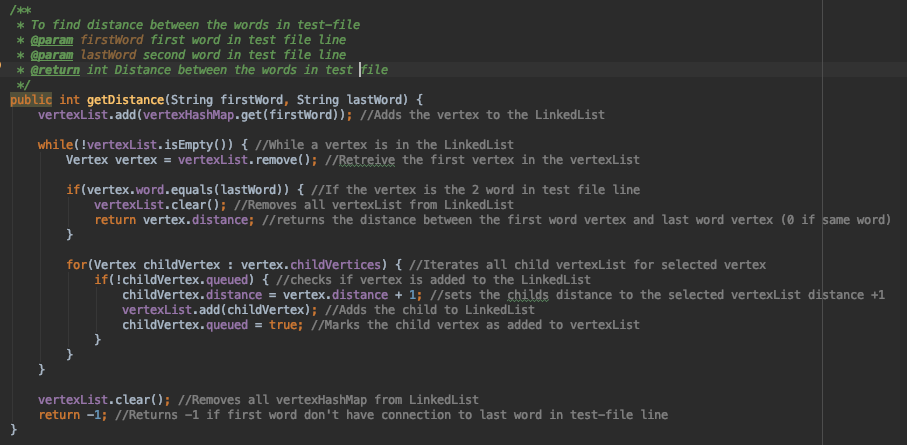
Skapa en riktad graf: O(1) tid, Lägga till ord i graf: O(N) tid. Att lägga till element i Linked List tar O(1) tid och att lägga till element i HashMap tar O(1) tid. Vi är inte intresserade av konstanter i något fall vilket resulterar i en tidskomplexitet på O(N).

Den mest dominerande uppgiften är att iterera genom alla ord och sedan hitta kopplingar till dessa då vi behöver använda oss av två for-loopar för att kunna jämföra varje ord med resterande ord i listan (se metoden nedan). I den inre for-loopen finns även två if-satser som kontrollerar om orden är samma samt om orden har någon koppling baserat på kraven. Eftersom det är två for-loopar (vilka vi är intresserade utav) och två if-satser (som vi inte är intresserade utav) kan vi förhålla oss till for-looparna vilket ger oss en tidskomplexitet på O(N2).

Se kodsnutt nedan från BFS- klassen:



Tidskomplexiteten för att göra bredden först- sökningen tar i värsta fall O(N2). Metoden getDistance i BFS- klassen har en while-loop som itererar genom kön så länge där finns noder att läsas genom vilket i värsta fall tar O(V) eftersom vi inte är intresserade av if-satsen. Däremot finns där en for-loop i denna while-loop som itererar genom alla barn-noder för vald nod, vilket ger oss O(E) eftersom detta utförs för varje nod (V). Detta kan i bästa fall vara O(1) och värsta fall O(V2). Se kodsnutt i nedan från BFS-klassen:



Klassen BFS lagrar alla objekt av klassen Vertex i en HashMap. Detta innebär att komplexiteten för minnesutrymmet är O(V). Varje Vertex sparar sedan alla barnnoder i en hash map, vilket ger en komplexitet för minnesutrymmet på O(E).

Det totala minnesutrymmet för minneshanteringen blir (O(V+E)). V står för antalet ord (vertices eller noder) och E står för antalet bågar (edges) mellan orden.