Relazione Esercizio 2

Primo uso: metodo *editDistance()*

Come richiesto dall'esercizio, è stato implementato, un metodo ricorsivo *editDistance()*, che prende in input due stringhe qualsiasi, e restituisce l'edit distance tra di esse. Il suddetto metodo è stato implementato seguendo le specifiche del testo, utilizzando due controlli iniziali sulla lunghezza delle stringhe, e assegnando gli opportuni valori ai parametri interi che rappresentano rispettivamente, la "*d_no-op*", la "*d_canc*" e la "*d_ins*".

E' possibile trovare i test del metodo *editDistance()* nella classe *EditDistanceTest*, all'interno della quale sono contenuti dei test di unità per ogni metodo implementato nella classe *EditDistance*. Tutti i test sono risultati essere soddisfatti da tutti i metodi *editDistance()*, *editDistanceDynBottomUp()*, ed *editDistanceDynTopDown()*.

Secondo uso: *editDistanceDyn()*

Per quel che riguarda l'implementazione del metodo *editDistanceDyn*: è stato deciso in fase di progettazione, di implementare anziché un singolo metodo che facesse uso di una tecnica di programmazione dinamica, due metodi che fanno uso della medesima tecnica di programmazione dinamica, uno iterativo e uno ricorsivo. Questi due metodi hanno le seguenti peculiarità:

1. **editDistanceDynBottomUp:** questo metodo **iterativo** si avvale di una **tecnica di programmazione dinamica** che fa utilizzo di una matrice, all'interno della quale memorizza in maniera **bottom-up** i risultati di tutte le edit distance applicate alle due stringhe, e restituisce la minore di esse.

2. **editDistanceDynTopDown:** questo metodo **ricorsivo** si avvale di una **tecnica di programmazione dinamica** che fa utilizzo di una matrice, all'interno della quale memorizza i risultati di tutte le chiamate ricorsive precedenti applicate alle due stringhe in maniera **top-down**, e restituisce la minore di esse.

Come menzionato prima, entrambi i metodi hanno superato con successo gli stessi test applicati al primo metodo *editDistance()*.

Come richiesto dall'esercizio, è stata implementata una classe *EditDistanceMain*, all'interno della quale è presente un *main()* che utilizza entrambi i metodi sopra descritti con l'obbiettivo di determinare, per ogni parola *w* in *correctme.txt*, la lista di parole in *dictionary.txt* con edit distance minima da *w*. Queste parole vengono stampate a video durante l'esecuzione del programma, insieme al tempo impiegato e alla versione della *editDistanceDyn* impiegata.

Parlando di tempistiche, i tempi di esecuzione dei due metodi *editDistanceDynBottomUp()* ed *editDistanceDynTopDown()* sono stati analizzati, ottenendo i seguenti risultati:

(**Premessa**: nello sviluppare questo programma è stato utilizzato l'IDE Intellij IDEA, del quale è stato più volte utilizzato il terminale integrato per misurare i tempi e controllare l'output, e in virtù di tale impiego, riportiamo qui sotto anche i tempi misurati con esso)

Terminale impiegato	editDistanceDynBottomUp (Iterativo)	editDistanceDynTopDown (Ricorsivo)
Bash (nativa del SO)	Circa **50** secondi	Circa **55** secondi
Integrato in Intellij IDEA	Circa **30** secondi	Circa **35** secondi

Come si può notare dalla tabella, i metodi **differiscono** in ogni caso di **5 secondi**, a parità di memoria e CPU consumate (abbiamo verificato usando il comando top durante l'esecuzione del programma).