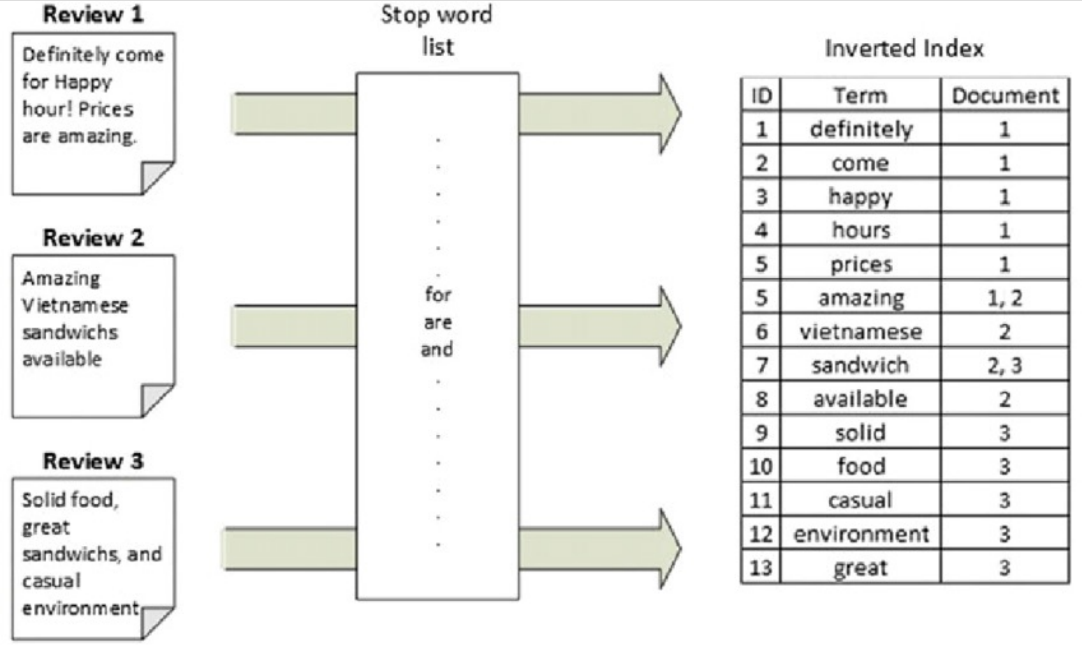
Parte 1 - Liste

Un caso di studio: Inverted Index



ID	Term	Document
1	definitely	1
2	come	1
3	happy	1
4	hours	1
5	prices	1
5	amazing	1, 2
6	vietnamese	2
7	sandwich	2, 3
8	available	2
9	solid	3
10	food	3
11	casual	3
12	environment	3
13	great	3

Inverted Index

Un inverted index è una struttura dati in cui si collegano I contenuti alle loro posizioni in un documento o in un insieme di documenti.

ESEMPIO

1

Giorgio oggi gioca con la palla

2

Oggi la palla costa 5 euro

3

Oggi ho pranzato con Giorgio spendendo 2 euro

Parola	Totale	Documento
Giorgio	2	1,3
Oggi	3	1,2,3
Gioca	1	1
Palla	2	1,2
Euro	2	2,3
Pranzato	1	3
…		…

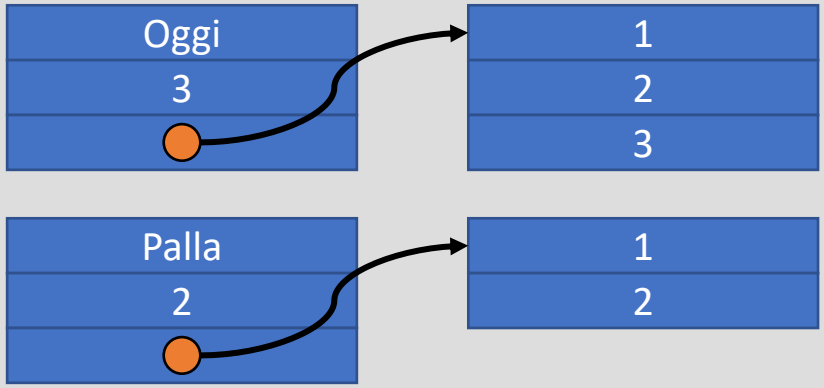
Note per l'implementazione

La chiave è quindi la **PAROLA**

Viene poi mantenuto anche il numero totale di occorrenze

Il valore associato rappresenta invece la posizione all'interno di un documento, o il nome/numero del documento

Gli identificatori dei documenti, nel nostro caso numeri interi, sono memorizzati all'interno di una lista detta **posting list**



File di partenza

- **lista.h, lista.cc, tipo.h, tipo.cc**
  - Per la gestione della posting list
- **inverted**
  - Che contiene i dati relativi all'inverted index
    - Prima riga: numero di parole
    - Dalla seconda in poi: PAROLA, TOTALE, [IDENTIFICATIVI]
- **doc**
  - Che contiene i dati relativi ad un documento
    - Struttura: ID, [ELENCO PAROLE]

Costruiamo il tipo di data parola

```
struct parola{
        char p[80]; //parola
        int n_doc;  //numero di documenti che contengono la
                      parola
        lista l;    //lista dei documenti
};
```

Questa struct dobbiamo inserirla all'interno di un file header parola.h

Punti da realizzare – Punto 1

Creare il file `compito.cc` contenente:

Funzione **`parola`**`* load(int& dim)`

Carica dal file inverted l'inverted index. Tutte le parole sono memorizzate in un vettore dinamico di tipo parola. La dimensione del vettore dinamico è letta dalla prima riga del file inverted.

La funzione restituisce il vettore dinamico e la sua dimensione nel parametro passato per riferimento

Procedura `stampa(parola*, int)`

Stampa il contenuto dell'inverted index (primo parametro vettore dinamico, secondo parametro dimensione del vettore).

Un `main` che richiama in sequenza le due funzioni.

Punto 1 (cont)

Creare un makefile per la generazione dell’eseguibile.

Il makefile deve contenere i phony target clean e cleanall.

Punti da realizzare – Punto 2

Funzione di aggiornamento:

- aggiungere la procedura **`update(parola* &II, char* filename, int& n)`**

  Aggiorna l'inverted index caricando il contenuto del documento contenuto nel file fileName.

Ogni file caricato ha la stessa struttura del file doc. Il codice deve gestire il caso di aggiunta di una parola, di aggiunta di un id di documento alla posting list di un parola già presente nell'inverted index. Il parametro n contiene la nuova dimensione della vettore

- Estendere il **main** affinché aggiorni l'inverted index con il documento contenuto nel file `doc`. Il main deve chiedere all'utente il nome del file da caricare, richiamare la procedura update e richiamare la funzione stampa per stampare l'inverted index risultante

Punti da realizzare – Punto 3

Estendere il main con la stampa dei documenti che soddisfano una richiesta “word1 AND word2”:

- Aggiungere a compito.cc la procedura
  void AND(parola* ll, char* W1, char* W2, int n)
  che stampa l’elenco dei documenti che contengono entrambe le parole.
- Estendere il main affinché chieda all’utente le parole e stampi il risultato richiamando la procedura AND.

Punti da realizzare – Punto 4

- Generare la documentazione con Doxygen
- Aggiungere una breve descrizione del programma realizzato per la main page
- Commentare la procedura stampa descrivendo i parametri in ingresso e aggiungendo una breve descrizione sull'obiettivo

Dati di prova

Il Punto 1 deve stampare l'inverted index contenuto nel file inverted.

Il Punto 2 deve stampare:

computer

1 2 4 5

laptop

1 3

tower

1 2 3 5

voltage

5

Dati di prova

Il Punto 3.a data la richiesta **computer AND tower** deve stampare:

1 2 5

Infatti i documenti 1, 2 e 5 contengono tutte e due le parole specificate