# TDT4117 Information Retrieval - Autumn
1/11/2022
Assignment 4
Ilias Kalantzis

## Task 1: Text Indexing

At first, we load the document and split it into lines. Afterward, we remove the punctuation and made the letters lower.

a) For an inverted file/list, we split the words into tokes, and then we make a dictionary in which the term has as an index the line that we find it.
b) To make an inverted list with block addressing, we assume that a block has a size of 4 words. So we split our text into blocks of 4 and create a dictionary in which the term has as an index the block that it was assigned
c) Suffix tree, viewing texts as one (single) long string. All text positions are uniquely identified by their position, known as an index point
d) In a posting inverted list, each document in the collection is given a unique number, and then we create a dictionary o in which on one side we have the term and on the other side the unique document number in which you find

## Task 2: Index Analysis Using Lucene

a)
ELK: The ELK stack gives you the ability to aggregate logs from all your systems and applications, analyze these logs and create visualizations for application and infrastructure monitoring, fast troubleshooting, security analytics and many more

Lucene: Lucene is a full-text search library in Java that makes it easy to add search functionality to an application or website. It does so by adding content to a full-text index

b)
Each text is divided into many parts or words to generate the Inverted Index. Although it may be adjusted, the rule is to utilize whitespace as the obvious word separator.

A list of pairs (document id, occurrences) is also provided for each phrase, indicating the document's ID where the term is discovered as well as the quantity of times the term appears in the text.

The Inverted Index can be understood as a simple key/value dictionary where per each term we store a list of appearances of those terms in the documents and their frequency.Thus, an Appearance class represents a single Appearance of a term in a document:

The Database class is a fake in-memory DB used to persist the documents after they have been indexed.

d)
On the first 2 queries, the answer is identical, with both systems returning document 2. On the last query, my index implementation returns all 6 documents in comparison with elk which returns documents 6,2,1. This is probably caused by the fact that my index implementation doesn't take into consideration phrases but only individual words. Unfortunately, by the way my indexing has been build there is no way to get the same result as the ELK

f) From my results, the search returns the same documents for both queries, with none of them having to do anything about the university. But guessing from the format of the queries, probably the first one "Norwegian and University and Science and Technology" should return the documents containing one of the terms in it and the other one "Norwegian University Science Technology" should return the correct answer.