**Information Retrieval (Lab Test)**

**01 February 2017**

Given the XML file at <http://tinyurl.com/ir-lab-articles> containing a list of 40,000 scientific articles delimited by the tag <article>…</article>. The goal is to:

1. Create a Lucene index with one document per article, each including three fields:
   * The author name (XML element <author>)
   * The article title (XML element <title>)
   * The year name (XML element <year>)

**Notes:**

* + You can assume that each <article> element has exactly one title, one year and at least one author. In case of multiple authors, their names must be concatenated into a single field.
  + Author names and year must be tokenized with a WhiteSpaceAnalyzer, and title must be tokenized with the ClassicAnalyzer.

1. Build a search engine that lets users to search articles via a command line interface in which she has to specify some keywords for the **author name,** and a **range of years** input as “starting year” and “ending year” (extremes included). The search engine must support the following query: the author names must be matched by the corresponding keywords; and the publishing year must occur in the specified range (*hint:* this can be implemented by generating and issuing a ***sequence of queries***, one per year of the queried range, that are composed by transforming every year of the queried range into a ***string*** to be exactly-matched against the content of the field <year>). The resulting docIDs and their scores have to be concatenated and then sorted by score. The top-10 results have to be printed, along with all fields and ranking score.

Tips:

* We suggest the following procedure:
  1. Build a function that processes the XML files and gets the field values
  2. Test this function to make sure the fields are correctly extracted
  3. Create a script to build the index
  4. Create a script to search the index and run a few test queries
* Relative Xpath queries can be run on single nodes, e.g. article\_node.xpath(“./author/text()”) will get the list of text content of all direct children of article\_node having tag “author”