## INSTITUTO TECNOLÓGICO DE COSTA RICA I SEMESTRE 2021

**INGENIERÍA EN COMPUTACIÓN**

## IC-8060 RECUPERACIÓN DE INFORMACIÓN TEXTUAL TAREAS PROGRAMADA 2

La tarea consiste en programar una aplicación que permita indexar una colección de páginas web geográficas así como hacer consultas a dicha colección. Se usará la biblioteca [Lucene de Apache](https://lucene.apache.org/) tanto para crear el índice de la colección como para realizar las búsquedas. Se quiere aprovechar además la capacidad de Lucene para indexar partes de los documentos en “campos” independientes que pueden ser usados explícitamente en las búsquedas.

La colección consiste de un conjunto de archivos HTML que contienen la descripción de los diferentes países y dependencias del mundo. Es tomado de Wikipedia en español. Cada archivo contiene la información de un país o dependencia. Se hará un procesamiento limitado de la estructura. Interesa indexar el texto incluido en los elementos **<p>** de las páginas. Además se hará un indexado especial del texto incluido en los elementos **<a>**. Por ejemplo:  
 **<a href=...>** *texto de interés* **</a>**.  
El resto de las etiquetas HTML debe ser ignorado.

Se considerarán como términos de búsqueda las palabras formadas por secuencias consecutivas de las letras del alfabeto español. Los números y las letras de otros alfabetos no serán tomados en cuenta en la indización.

**Indexado**

La colección debe ser procesada extrayendo la siguiente información:

|  |  |
| --- | --- |
| **Elemento** | **Procesamiento** |
| **<body>** | * Todo el texto contenido en elementos **<p>** dentro de este elemento será indexado en un campo llamado ***texto***. * ***Texto*** debe ser el campo default en búsquedas. * Separar el texto de este elemento en palabras (secuencias consecutivas de los caracteres [A-Za-zÁÉÍÓÚÜáéíóúüÑñ] * Eliminar stopwords * Aplicar stemming a las palabras. Se supondrá que el algoritmo de stemming convertirá las letras a minúsculas y eliminará las tildes; si no lo hace, entonces debe programarse eso explícitamente. |
| **<a>** | * El contenido de este elemento será indexado en un campo llamado ***ref***. * Para este campo, no se aplicará stemming a las palabras; solo se convertirán a minúsculas y se eliminarán las tildes. Se debe preservar la eñe en su forma original. |
| **<title>** | * El contenido de este elemento será indexado en un campo llamado ***titulo***. Se le aplicarán las mismas operaciones que para el elemento <a>. |

Adicionalmente se incluirán en el índice los siguientes campos

|  |  |
| --- | --- |
| **resumen** | * Consiste de los primeros 200 caracteres incluidos en el campo **texto**. * El texto no sufre transformación. Se mantiene como viene en el documento (con puntuación, tildes y mayúsculas), pero sin etiquetas de HTML. |
| **URL** | * Ruta local de la página en la computadora. * Se va a usar para mostrar la página completa si es escogida en una búsqueda. |

Se usará la lista de stopwords que se adjunta al final de este documento. Los términos serán simplificados por medio de la extracción de raíces (stemming). Se utilizará la técnica de eliminación de afijos por medio del algoritmo de Porter para español. Se recomienda usar la implementación Snowball del stemmer en español. Está disponible en muchos lenguajes de programación.

La aplicación desarrollada debe poder escoger los directorios que indexa. Se deben indexar inicialmente los continentes América y Asia. Luego se agregan África, Europa y Oceanía. **No se debe reindexar todo de nuevo**, sino que se deben aprovechar las facilidades que ofrece Lucene para indexar en forma incremental.

Luego de cada paso de indexado, deben poder realizarse búsquedas. Las búsquedas deben tomar en cuenta los nuevos archivos incluidos.

**Búsqueda**

Lucene provee un lenguaje de consulta muy rico, el cual puede ser usado entre otras cosas para ejecutar consultas booleanas, vectoriales, frases, por proximidad, o con errores; se pueede escoger además los campos específicos del índice usado. En la página [Apache Lucene – Query Parser Sintax](https://lucene.apache.org/core/2_9_4/queryparsersyntax.html) se muestran ejemplos de los diferentes tipos de consulta que están disponibles directamente.

La aplicación debe proveer una interface que permita escoger un índice, leer consultas e invocar las facilidades de Lucene para realizar dichas consultas. Debe ser posible aprovechar los diferentes campos especificados.

Como resultado de las consultas, la aplicación debe mostrar la similitudes, el campo **título** y el campo **resumen** de los mejores páginas del escalafón. Inicialmente se muestran los mejores 20 páginas del escalafón, pero el usuario puede pedir grupos adicionales con las siguientes 20 páginas del escalafon.

Adicionalmente, el usuario debe poder visualizar cualquier página que aparezca en el resultado que se le presenta. El sistema debe localizar la página escogida en la colección y mostrarla en un navegador. **¡Las páginas no deben almacenarse en el índice!** Se debe usar el campo **URL** del índice para localizar las páginas escogidas.

Al final de este documento se incluyen algunas consultas de prueba.

**Actualizar índices**

La aplicación debe poder tomar un conjunto de páginas y actualizar un índice ya existente. Esto es, sin crear ese índice desde el inicio, simplemente debe incorporar los nuevos documentos.

Para mostrar eso, se debe indexar primero los continentes América y Asia. Luego se debe actualizar el índice usando los otros continentes (África, Europa y Oceanía).

Más concretamente, se deben realizar las siguientes pruebas:

1. Crear **INDX1** con las páginas de América y Asia.
2. Realizar búsquedas usando ese INDX1.
3. Crear **INDX2** con las páginas de África, Europa y Oceanía.
4. Realizar las mismas búsquedas del punto 2 usando INDX2.
5. Actualizar **INDX1** agregando los páginas de África, Europa y Oceanía.
6. Repetir el punto 2 ahora con INDX1 actualizado.  
   Se debe obtener la combinación de los resultados de los puntos 2 y 4.

**Consideraciones finales**

La tarea puede ser realizada en grupos de dos personas. Pueden usar cualquier lenguaje que se pueda combinar fácilmente con Lucene que está escrito en Java. Hay un “wrapper” de Lucene para Python llamado PyLucene. Para el manejo de HTML pueden usar las herramientas disponibles en el lenguaje escogido; por ejemplo [Beautiful Soup](https://www.crummy.com/software/BeautifulSoup/bs4/doc/) en Python.

La fecha límite de entrega es el lunes 17 de mayo.

**Lista de stopwords**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| a  acá  ahí  ajena  ajenas  ajeno  ajenos  al  algo  algún  alguna  algunas  alguno  algunos  allá  alli  allí  ambos  ampleamos  ante  antes  aquel  aquella  aquellas  aquello  aquellos  aqui  aquí  arriba  asi  atras  aun  aunque  bajo  bastante  bien  cabe  cada  casi  cierta  ciertas  cierto  ciertos  como  cómo  con  conmigo  conseguimos  conseguir  consigo | consigue  consiguen  consigues  contigo  contra  cual  cuales  cualquier  cualquiera  cualquieras  cuan  cuán  cuando  cuanta  cuánta  cuantas  cuántas  cuanto  cuánto  cuantos  cuántos  de  dejar  del  demás  demas  demasiada  demasiadas  demasiado  demasiados  dentro  desde  donde  dos  el  él  ella  ellas  ello  ellos  empleais  emplean  emplear  empleas  empleo  en  encima  entonces  entre  era | eramos  eran  eras  eres  es  esa  esas  ese  eso  esos  esta  estaba  estado  estais  estamos  estan  estar  estas  este  esto  estos  estoy  etc  fin  fue  fueron  fui  fuimos  gueno  ha  hace  haceis  hacemos  hacen  hacer  haces  hacia  hago  hasta  incluso  intenta  intentais  intentamos  intentan  intentar  intentas  intento  ir  jamás  junto | juntos  la  largo  las  lo  los  mas  más  me  menos  mi  mía  mia  mias  mientras  mio  mío  mios  mis  misma  mismas  mismo  mismos  modo  mucha  muchas  muchísima  muchísimas  muchísimo  muchísimos  mucho  muchos  muy  nada  ni  ningun  ninguna  ningunas  ninguno  ningunos  no  nos  nosotras  nosotros  nuestra  nuestras  nuestro  nuestros  nunca  os | otra  otras  otro  otros  para  parecer  pero  poca  pocas  poco  pocos  podeis  podemos  poder  podria  podriais  podriamos  podrian  podrias  por  por qué  porque  primero  puede  pueden  puedo  pues  que  qué  querer  quien  quién  quienes  quienesquiera  quienquiera  quiza  quizas  sabe  sabeis  sabemos  saben  saber  sabes  se  segun  ser  si  sí  siempre  siendo | sin  sín  sino  so  sobre  sois  solamente  solo  somos  soy  sr  sra  sres  sta  su  sus  suya  suyas  suyo  suyos  tal  tales  también  tambien  tampoco  tan  tanta  tantas  tanto  tantos  te  teneis  tenemos  tener  tengo  ti  tiempo  tiene  tienen  toda  todas  todo  todos  tomar  trabaja  trabajais  trabajamos  trabajan  trabajar  trabajas | trabajo  tras  tú  tu  tus  tuya  tuyo  tuyos  ultimo  un  una  unas  uno  unos  usa  usais  usamos  usan  usar  usas  uso  usted  ustedes  va  vais  valor  vamos  van  varias  varios  vaya  verdad  verdadera  vosotras  vosotros  voy  vuestra  vuestras  vuestro  vuestros  y  ya  yo |

**CONSULTAS**

|  |  |
| --- | --- |
| **nilo** | Buscar “nilo” en el campo default, ***texto***. |
| **ref:nilo** | Buscar “nilo” en el campo ***ref***. |
| **río AND ref:nilo** | Búsqueda booleana |
| **"río nilo"** | Búsquedad de frases |
| **fiebre^1 amarilla^4** | Búsqueda vectorial con prioridad para amarilla |
| **fiebre^4 amarilla^1** | Búsqueda vectorial con prioridad para fiebre |
| **cuanacaste~0.8** | Búsqueda con error (guanacaste) |
| **"fuga cerebros"~2** | Búsqueda por proximidad (aceptaría “fuga de cerebros” y “cerebros en fuga”) |
| **ref:"fiebre oro"~2** | Búsqueda por proximidad para el campo ***ref***. |