**Instituto Tecnológico de Costa Rica**
**Escuela de Ingeniería en Computación**
**IC-4700 Lenguajes de programación**

# Asignación #3, Índice KWIC

Historial de revisiones:
- 2020.07.26: Versión base.

***Lea con cuidado este documento.*** Si encuentra errores en el planteamiento[1], por favor comuníqueselos inmediatamente al profesor.

## Objetivo

Al concluir esta asignación, Ud. habrá aprendido más sobre el paradigma de programación funcional vía la aplicación práctica de sus conocimientos de Computación con el lenguaje Haskell. También aprenderá sobre la forma en que se plantea la entrada y salida de datos, sobre archivos, en un lenguaje funcional 'puro'. Además, conocerá respecto de una de las primeras técnicas de indexación computacionales, *KWIC* (Key-word-in-context), desarrollada a fines de la década de 1950 como apoyo al entonces naciente campo de la *Recuperación de información* (*Information retrieval*) para el manejo de información textual. Su relevancia puede ser constatada cada vez que Ud. hace una búsqueda en la Web. Por ejemplo:

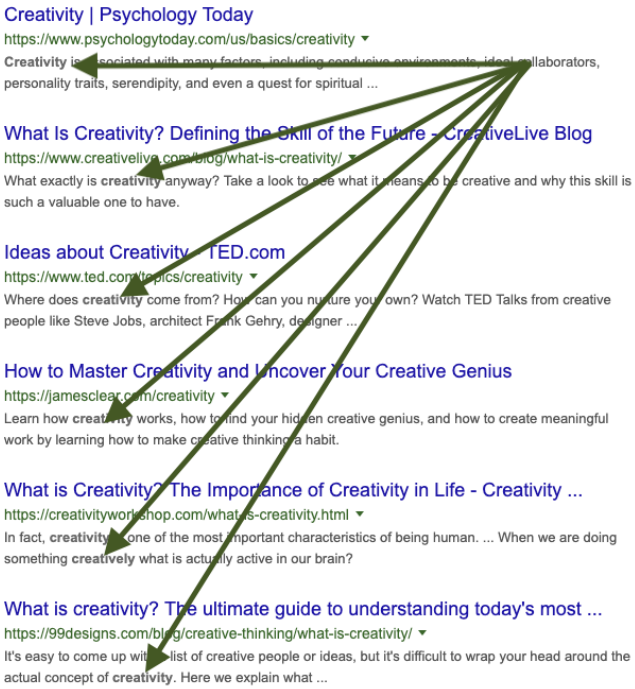


Imagen tomada de [Magnet, 2019].

Observe que la palabra 'creativity' aparece rodeada de su contexto, lo cual hace mas útil la búsqueda para quien la formuló.

La indexación estilo KWIC ha sido propuesta múltiples veces en las competencias (olimpiadas) de programación que se celebran mundialmente; es un ejercicio saludable resolver problemas relacionados a los que aparecen en esas competencias.

[1] El profesor es un ser humano, falible como cualquiera.

**Bases**
Referencias sobre KWIC y código prototípico desarrollado por el profesor.

**El problema clásico**
Dados los títulos de varios libros o artículos, producir un *índice* en que aparezcan 'rotaciones' de todos los títulos, ordenadas alfabéticamente, según las palabras clave (o *significativas*) que aparezcan en los títulos.

David Parnas, uno de los padres de la Ingeniería del Software, lo planteó así en uno de sus artículos clásicos sobre modularización [Parnas, 1972]:

**Example System 1: A KWIC Index Production System**

The following description of a KWIC index will suffice for this paper. The KWIC index system accepts an ordered set of lines, each line is an ordered set of words, and each word is an ordered set of characters. Any line may be "circularly shifted" by repeatedly removing the first word and appending it at the end of the line. The KWIC index system outputs a listing of all circular shifts of all lines in alphabetical order.

This is a small system. Except under extreme circumstances (huge data base, no supporting software), such a system could be produced by a good programmer within a week or two. Consequently, none of the difficulties motivating modular programming are important for this system. Because it is impractical to treat a large system thoroughly, we must go through the exercise of treating this problem as if it were a large project. We give one modularization which typifies current approaches, and another which has been used successfully in undergraduate class projects.

El profesor recomienda vivamente la lectura de este influyente artículo.

**Nuestro problema**
La búsqueda y la clasificación son aplicaciones informáticas prototípicas. Este proyecto versa sobre diseñar, construir y probar un programa en Haskell que organiza los *títulos* de obras escritas - como libros o artículos (o hasta oraciones) - para facilitar una "búsqueda humana" eficiente basada en diferentes palabras claves. Dadas una lista de títulos y una lista de palabras 'no significativas'[2], deseamos escribir un programa que genere un índice *KWIC* (palabra clave en contexto) de los títulos.

Las palabras clave son cualesquiera palabras que son 'significativas'[3] y no palabras 'no significativas'[4]. En un índice KWIC, un título aparece *una* vez por *cada* palabra clave que aparece en el título original; el título es 'rotado' de manera que cada una de las palabras clave inicia una 'rotación significativa' del título y el segmento que la sigue (en el título original) se concatena al final de esa rotación. Las rotaciones que comienzan con una palabra no significativa no forman parte del índice KWIC. El índice KWIC está ordenado alfabéticamente por palabra clave.

**Entradas**
Su programa ofrecerá una sencilla interacción con el usuario para que éste indique:

- El nombre de un archivo de texto donde aparecen los *títulos* de las obras por indexar.
- El nombre del archivo de texto donde están almacenadas las palabras *no significativas*.

[2] Palabras que podemos ignorar, por ser muy comunes en un idioma y poco relevantes para realizar procesos de búsqueda y clasificación de texto.
[3] Generalmente son sustantivos y verbos.
[4] Como artículos, adjetivos, adverbios, conjunciones, preposiciones y pronombres.

- El nombre del archivo de texto donde se dejará la *salida* producida por su programa.

En el archivo de títulos, cada título aparecerá en una línea aparte. Su programa leerá el archivo de títulos de manera que cada título conforme una hilera (*string*) aparte y todos queden almacenados en una lista simple de Haskell.

En el archivo de palabras no significativas, estas pueden aparecer separadas por uno o más espacios en blanco, tabuladores o fines de línea. Su programa leerá el archivo de palabras no significativas de manera que cada palabra conforme una hilera (*string*) aparte y todas queden almacenados en una lista simple de Haskell.

Las palabras clave son todas las palabras presentes en los títulos pero que *no* pertenecen al conjunto de las palabras no significativas.

Tanto las palabras clave como las palabras no significativas deben estar conformadas por *letras*. Los caracteres correspondientes a signos de puntuación no deben formar parte de las palabras, aunque sí formen parte de los títulos por indexar.

Las palabras clave pueden aparecer más de una vez en un mismo título.

**Proceso**
Su programa deberá generar *todas* las rotaciones posibles para cada título, descartar las que inicien con una palabra no significativa, mantener las rotaciones que comiencen con una palabra significativa, y ordenar alfabéticamente las rotaciones significativas (de todos los títulos).

Por ejemplo, suponga que este es el contenido de un archivo de entrada[5]:

```
Descent of Man
The Ascent of Man
The Old Man and The Sea
A Portrait of The Artist As a Young Man
```

La salida que produce el prototipo facilitado por el profesor[6] es esta:

```
Artist As a Young Man >< A Portrait of The
Ascent of Man >< The
Descent of Man ><
Man >< A Portrait of The Artist As a Young
Man >< Descent of
Man >< The Ascent of
Man and The Sea >< The Old
Old Man and The Sea >< The
Portrait of The Artist As a Young Man >< A
Sea >< The Old Man and The
Young Man >< A Portrait of The Artist As a
```

Las rotaciones *significativas* inician cada una con una *palabra significativa* y están ordenadas ascendentemente (lexicográficamente, según un diccionario alfabético).

Arriba ilustramos el tipo de formato básico. El prototipo facilitado por el profesor ilustra el proceso, pero está *incompleto* respecto de las especificaciones dadas en el presente enunciado.

**Salidas**
La salida deberá quedar guardada en un archivo de texto simple, cuyo nombre es especificado como tercera entrada al interactuar con su programa. Si ya existiera un archivo con ese nombre, se deberá indicar eso al usuario y darle la opción de cambiar el nombre del archivo.

---

[5] Ejemplo tomado de https://users.cs.duke.edu/~ola/courses/cps100/spr97/kwic/kwic.html
[6] Ver código en `kwic v1.hs`, que ahora incluye varias conjunciones en inglés, y los títulos usados en el presente ejemplo.

Hay dos posibles formas para las salidas: *básica* y *alineada*.

*Salida básica*
La salida básica puede ser como la mostrada por el prototipo facilitado por el profesor. Ustedes tienen la opción de mejorarla.

*Salida alineada*
Las salidas alineadas deben producirse de manera que *para cada rotación significativa* se resalte en **mayúsculas** la palabra clave que inicia la rotación. La rotación se muestra con todas las palabras en el mismo orden del título original, pero con la palabra clave (inicial) resaltada en *MAYÚSCULAS* y todas las demás palabras en *minúsculas*. Las rotaciones deberán estar alineadas de manera que luzcan alineadas respecto de la palabra clave que caracteriza a cada rotación, también deben aparecer ordenadas alfabéticamente respecto de la palabra clave inicial, como sigue[7]:

```
                  a portrait of the ARTIST as a young man
                                the ASCENT of man
                                    DESCENT of man
                        descent of MAN
                     the ascent of MAN
                           the old MAN and the sea
a portrait of the artist as a young MAN
                                the OLD man and the sea
                                  a PORTRAIT of the artist as a young man
               the old man and the SEA
      a portrait of the artist as a YOUNG man
```

**Idiomas de las palabras no significativas**
Ustedes deberán investigar sobre las palabras *no significativas* de *dos* idiomas: español (castellano) e inglés. En principio, las palabras no significativas (para indexar) corresponden a: artículos, preposiciones, adjetivos, adverbios, pronombres y conjunciones. Tienen libertad de añadir más idiomas. Guarden las palabras no significativas en archivos separados.

**Requerimiento técnico extra**
La estructura de datos básica para almacenar y consultas las palabras no significativas es una lista; la función de búsqueda usual es secuencial y es lineal. Es un requerimiento *extra* almacenar las palabras no significativas en un árbol de búsqueda que esté *optimizado* de manera que su *altura* sea *mínima*, o bien en un arreglo que permita una búsqueda binaria, o bien en una tabla de *hash*.

**Pruebas**
Sus pruebas deben dar evidencia del adecuado funcionamiento de su programa y de los elementos que lo conforman.

**Calificación**

| | |
|---|---|
| Generación correcta del índice KWIC básico | 30 |
| Generación correcta del índice KWIC alineado | 15 |
| Lectura correcta de archivos de entrada | 10 |
| Escritura correcta del archivo de salida | 10 |
| Estilo de codificación | 20 |
| Estructura de datos alterna para palabras no significativas (extra) | 10 |
| Documentación externa, pruebas e investigación alrededor de las palabras no significativas | 15 |

**Documentación**
Deberán preparar un informe técnico que incluya:

- Portada que identifique a los autores del informe, con sus carnets.

[7] Ejemplo tomado de https://users.cs.duke.edu/~ola/courses/cps100/spr97/kwic/kwic.html

- Introducción al informe.
- Describir la investigación realizada sobre las palabras *no significativas*, para cada uno de los idiomas con que trabajó su grupo. Citar las fuentes consultadas.
- Describir su estrategia para diseñar y construir las funciones que componen su programa: manejo de la entrada, procesamiento de la entrada, producción de rotaciones, filtrado de rotaciones significativas, ordenamiento de rotaciones, generación de la salida, manejo de la salida.
- Mostrar los resultados de probar su programa con datos pertinentes, para cada uno de los idiomas indicados. Si trabajaron con más de dos idiomas, incluyan esos datos y las pruebas realizadas.
- Análisis de los resultados obtenidos.
- Discusión y conclusiones respecto del desarrollo del proyecto y de los resultados obtenidos.
- Reflexión sobre la experiencia de trabajar con el lenguaje Haskell y el paradigma de programación funcional *pura* con entrada y salida *monádica*.
- Referencias. Los libros, revistas y sitios Web que utilizaron durante la investigación y el desarrollo de su proyecto. Citar toda fuente consultada (incluidas las referentes a las palabras no significativas para cada idioma trabajado).
- Apéndice: Descripción resumida de las tareas realizadas por cada miembro del grupo de trabajo.
- Apéndice: Código fuente su solución.
- Apéndice: Detalles de las pruebas creadas para ejercitar su programa. Muestras de los resultados obtenidos.
- Apéndice: Instrucciones para ejecutar su programa.
- Para los que desarrollen el tema extra:
  - Describir y justificar la representación de datos elegida y las estrategias algorítmicas para las funciones de consulta e inserción de datos.
  - Demostrar experimentalmente o analíticamente por qué su alternativa (estructura de datos + algoritmos) tiene una eficiencia *mejor* que la básica sobre listas, que es *lineal* (*O(n)*)[8], y caracterizar esa eficiencia (*O(log n)*, *O(1)*, u otra).

**Archivos por entregar**

- Debe guardar su trabajo en una carpeta comprimida (formato **`.zip`**) según se indica abajo[9]. Esto debe incluir:
  - Documentación indicada arriba en un solo documento, con una portada donde aparezcan los nombres y números de carnet de los miembros del grupo. El documento debe estar en formato `.pdf`.
  - Código fuente completo de su solución a esta asignación (en una carpeta, si fuera necesario).
  - Pruebas (en carpeta aparte): descripción de las pruebas, código creado para *probar* su solución, corridas de las pruebas y evidencias (archivos de texto y 'pantallazos').
  - Archivos de palabras *no significativas*, para *cada* idioma trabajado.
  - Archivos de títulos, para *cada* idioma trabajado.

**Entrega**

Fecha límite: **lunes 2020.08.17**, antes de las 23:55. No se recibirán trabajos después de la fecha y la hora indicadas.

Los grupos pueden ser de *hasta* **3** personas.

Debe enviar por correo-e el ***enlace***[10] a un archivo comprimido almacenado en la nube con todos los elementos de su solución a estas direcciones: `itrejos@itcr.ac.cr` y `andres.mirandaarias@gmail.com`.

> El asunto (*subject*) debe ser:
> `IC-4700 - Asignación 3 -` carnet + carnet + carnet.
>
> Los carnets deben ir ordenados ascendentemente.

---

[8] Típica de una búsqueda *secuencial* en una lista no ordenada.

[9] **No use** formato **`.rar`**, porque es rechazado por el sistema de correo-e del TEC.

[10] Los sistemas de correo han estado rechazando el envío o la recepción de carpetas comprimidas con componentes ejecutables. Suban su carpeta comprimida (en formato **`.zip`**) a algún 'lugar' en la nube y envíen el hipervínculo al profesor y a su asistente mediante un mensaje de correo con el formato indicado.

Si su mensaje no tiene el asunto en la forma correcta, su trabajo será castigado con −10 puntos; podría darse el caso de que su proyecto no sea revisado del todo (y sea calificado con 0) sin responsabilidad alguna del profesor o del asistente (caso de que su mensaje fuera obviado por no tener el asunto apropiado). Si su mensaje no es legible (por cualquier motivo), o contiene un virus, o es entregado en formato `.rar`, la nota será 0.

La redacción y la ortografía deben ser correctas. El profesor tiene *altas expectativas* respecto de la calidad de los trabajos escritos y de la programación producidos por estudiantes universitarios de la carrera de Ingeniería en Computación del Tecnológico de Costa Rica. Los profesores esperamos que los estudiantes tomen en serio la comunicación profesional.

**Referencias**
Estas son algunas referencias que pueden serles útiles.


- Carnegie Mellon U. *Keyword In Context (KWIC)*. https://www.cs.cmu.edu/~ModProb/KWIC.html
- librarianshipstudies.com. *Keyword in Context (KWIC) Indexing*. https://www.librarianshipstudies.com/2017/02/keyword-in-context-kwic-indexing.html
- Luhn, H.P. *Keyword-in-Context Index for Technical Literature (KWIC Index)*. American Documentation, 1960.
- Magnet, P. Why Programmers need creativity. *Medium*, julio 2019. https://medium.com/swlh/why-programmers-need-creativity-982a38d345c8
- Marlow, S. (ed.). *Haskell 2010 Language Report*. https://www.haskell.org/onlinereport/haskell2010/, https://www.haskell.org/onlinereport/haskell2010/haskellpa1.html, http://haskell.org/definition/haskell2010.pdf.
- Parnas, David L. On the criteria to be used in decomposing systems into modules. *Communications of the ACM*, 15(12), diciembre 1972. pp. 1053-1058. https://dl.acm.org/doi/10.1145/361598.361623, https://web.archive.org/web/20070820133530/http://www.acm.org/classics/may96/
- U. Duke *CPS 100, Spring 1997: KWIC: Key Word in Context, or, Searching Kwicly*. https://users.cs.duke.edu/~ola/courses/cps100/spr97/kwic/kwic.html
- U. Lethbridge. Problem #8 KWIC Index. http://www.cs.uleth.ca/~forsyth/seminar/problems/kwic.html
- Wikipedia. *Key Word in Context*. https://en.wikipedia.org/wiki/Key_Word_in_Context


**Ponderación**
Este proyecto tiene un valor del 30% de la calificación del curso.