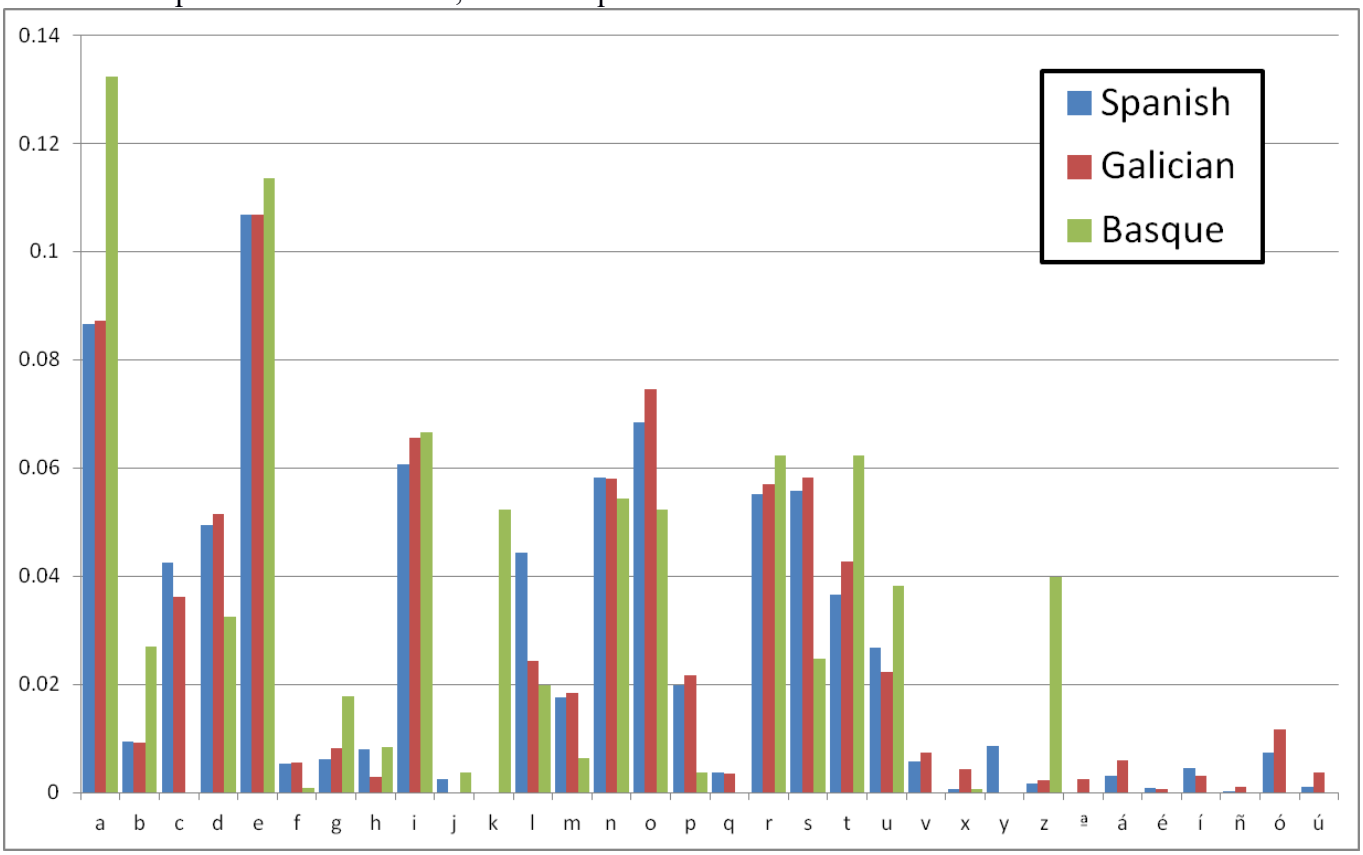
TAP 2021-22. Laboratorio.

La torre de Babel (I/O y Colecciones)

La finalidad de la presente práctica será poder relacionar diferentes lenguas entre sí, estableciendo una medida de semejanza en función de su forma escrita.

Supongamos que tenemos un mismo texto traducido a dos lenguas diferentes. Es de esperar que si estas dos lenguas son *parecidas*[1], los respectivos textos también lo sean. Obviamente, esto no ocurrirá siempre, ya que podrían existir dos lenguas semejantes que utilicen una escritura totalmente distinta. Pero, como veremos en esta práctica, este sencillo criterio será suficiente para la gran mayoría de las lenguas de nuestro planeta. Por tanto, definiremos la semejanza entre dos lenguas como la semejanza entre un mismo texto traducido a ambas lenguas.

De nuevo, definir la semejanza entre dos textos es una tarea compleja. Una de las maneras más sencilla (y aun así relativamente eficaz) es comparar la frecuencia de aparición de los caracteres en cada texto. El histograma de la figura, muestra la frecuencia de aparición de algunos caracteres (no todos) que podríamos encontrar en textos en español, gallego y vasco. Como puede observarse en la figura, el español y el gallego comparten cierta semejanza en cuanto a frecuencia de aparición de caracteres, mientras que el vasco se diferencia de ellos.



1 Lo de las "distancias" entre lenguas lleva al "conflicto" de considerar a algunas idiomas o dialectos. Hablando del Andaluz he aquí un interesante video de José María Pérez Orozco, catedrático de Lengua Española y Literatura.

Es posible definir una sencilla función que calcule la semejanza o grado de solapamiento de dos histogramas *h1* y *h2*:

$$\text{Similarity}(h1, h2) = \sum_{c \in h1 \cup h2} \min\big(h1(c), h2(c)\big)$$

Es decir, la similitud es la suma del mínimo de frecuencias para todos los caracteres de ambos histogramas. Dos lenguas que no compartan ningún carácter (tengan una escritura totalmente diferente) tendrán una semejanza 0, mientras que la semejanza entre una lengua y ella misma será 1 (la suma de todas las frecuencias de un histograma es igual a 1).

Un ejemplo práctico

Si tomamos la Declaración Universal de los Derechos Humanos, que está traducida a casi todas las lenguas del planeta (http://www.un.org/es/documents/udhr/law.shtml), y analizamos cuál es la semejanza entre las cuatro lenguas oficiales de España (español-spn, gallego-gln, catalán-cln y vasco-bsq), obtendríamos la tabla adjunta.

	gln	cln	bsq
spn	0.946	0.909	0.757
gln		0.899	0.765
cln			0.761

Tarea 1

Descargar el archivo http://gtts.ehu.es/German/Docencia/assets/udhr.zip que contiene la Declaración Universal de los Derechos Humanos traducida a 281 lenguas en formato de texto (codificación UTF-8) y descomprimirlo[2].

Crear una clase `Language` que represente a una lengua y pueda instanciarse mediante un fichero de texto. El constructor podrá ser:

```
public Language(String id, String descr, String fileName){…}
```

Donde `id` será un identificador de dicha lengua, `descr` una breve descripción y `fileName` será el nombre de un fichero de texto en formato UTF-8 que contendrá texto en dicha lengua. Ej.:

```
Language spn=new Language("spn","español (internacional)","udhr\txt\spn.txt");
```

La clase `Language` deberá almacenar internamente el histograma que se obtenga a partir del fichero de texto. Para ello, puede utilizarse un mapa que relacione cada carácter visto con su frecuencia de aparición: `Map<Character,Double>`. La clase también deberá implementar los siguientes métodos:

```
public String getId() {…}
public String getDescriptor(){…}
public String toString(){…}
public static double similarity(Language a, Language b){…}
```

No hace falta comentar los tres primeros métodos, ya que resulta obvio qué pueden devolver. El método **`similarity`** deberá tomar dos lenguas (nótese el modificador **`static`**) y devolver su semejanza.

Desde una clase para pruebas se deberá obtener la semejanza entre las cuatro lenguas oficiales de España en base a los textos de la Declaración Universal de los Derechos Humanos. Es decir, deberán obtenerse los valores de la tabla anterior.

[2] Una segunda versión de esta misma practica puede plantearse accediendo a los datos directamente dentro de "zip". En caso de hacerse, se podrá comparar el tiempo de ejecución entre ambas versiones.

Tarea 2

Crear una clase **`Babel`** que represente a un conjunto de lenguas. En combinación con la clase **`Languaje`**, el constructor de **`Babel`** permitirá cargar un conjunto de lenguas a partir de un directorio con el contenido descargado anteriormente. Esto es, un directorio con un fichero de texto denominado **`languages.txt`** conteniendo información sobre el conjunto de lenguas, una por fila. El primer campo (separado por un espacio) será el identificador de la lengua, y el resto de la línea será una descripción breve.

```
bsq Basque (Euskara)
chn Chinese (Mandarin)
ger Deutsch (German)
gls Gàidhlig Albanach (Scottish Gaelic)
...
```

En ese mismo directorio existirá un subdirectorio llamado **`txt`**, que deberá contener tantos ficheros de texto como lenguas contenga el fichero índice, y cuyos nombres coincidirán con los identificadores de cada lengua. Es decir, deberán existir los ficheros (los nombres son relativos a la carpeta que contiene el fichero **`languages.txt`**): **`txt\bsq.txt, txt\chn.txt, txt\ger.txt, txt\gls.txt, ...`**

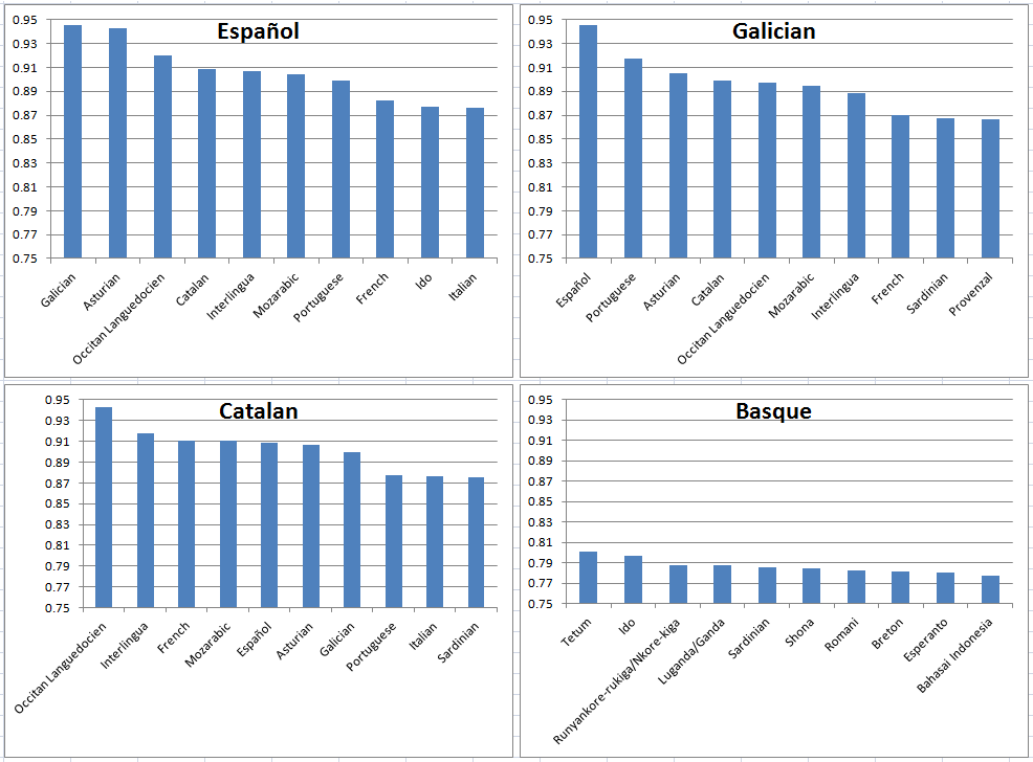
El constructor de la clase Babel deberá parecerse a: **`public Babel(String dirName){…}`**

Además, esta clase deberá implementar el método:

`public Language[] findMostSimilars(String id, int n){…}`

que deberá devolver -dado el identificador **`id`** de una lengua- las **`n`** lenguas con mayor semejanza a dicha lengua (en orden descendente).

En este caso la clase de prueba deberá devolver las 10 lenguas más semejantes a las 4 lenguas oficiales de España. La imagen muestra de manera gráfica el resultado del método.



Nota: el método estático

`java.util.Arrays.sort(T[] a, Comparator<? super T> c)`

ordena el array T de manera ascendente, haciendo uso del comparador c suministrado.