Pruebas

Se van algunas de las funciones usadas para crear el corpus. Lo obtenido en los apartados 2 y 3 no se va a comprobar pues el resultado es inmediato y no se han usado más funciones auxiliares.

```
setwd("./..")
path = getwd()
generator = paste(path, "/data/generador.R", sep = "")
processor = paste(path, "/data/processor.R", sep = "")
source(generator)
source(processor)
```

Intentamos usar path relativos ("./../data/generador.R") pero no parecía dejar abrir el archivo (usamos esa misma dirección con el file.exists y devolvió TRUE).

get_articles():

Debería devolver los títulos de todos los artículos, quitamos los suprimidos y formateando el nombre (quitar la tilde para unificarlos)

```
setwd("./..")
articulos = get_articles()
```

Buscamos la existencia de tildes o paréntesis. Además comprobaremos la cantidad de artículos que ha escogido para comprobar con otras funciones

```
grep("í",articulos)
```

```
## integer(0)
```

```
grep("[()]",articulos, perl = TRUE)
```

```
## integer(0)
```

```
length(articulos)
```

```
## [1] 697
```

get_content().

Debería devolver el contenido de los artículos, purgándolos del html. Comprobaremos que no tengo ninguna etiqueta del formato <...>. También que no esté vacío.

```
setwd("./..")
contenido = get_content()
grep("<[^>]*>", contenido)
```

```
## integer(0)
```

```
any(contenido=="")
```

```
## [1] FALSE
```

```
length(contenido)
```

```
## [1] 697
```

Vemos que contiene la misma cantidad de elementos que articulos, que ninguno está vacío y que no contiene ninguna etiqueta html.

get_titles():

Debería devolver las cabeceras de cada artículo. Podemos comprobar que la columna de Artículo este llena. Del resto de columnas no podemos hacer mucha comprobación pues no siguen un formato concreto (algunos pueden tener, por ejemplo, capítulo y no tener sección o viceversa, lo caul no nos permite hacer una comprobación de que sean correctos)

```
setwd("./..")
titles = get_titles()
any(is.na(titles$Artículo))
```

```
## [1] FALSE
```

```
nrow(titles)
```

```
## [1] 697
```

```
setwd("./..")
utils = separator()
corpus_names = utils$names
corpus_starts = utils$starts
corpus_docvars = utils$docvars
```

Primero para corpus names, comprobaremos que no esté vacío y que aparezca el artículo en cada uno:

```
any(corpus_names=="")
```

```
## [1] FALSE
```

```
length(grep("Artículo",corpus_names))
```

```
## [1] 697
```

```
length(corpus_names)
```

```
## [1] 697
```

Lo mismo haremos con corpus_starts.

```
any(corpus_starts=="")
```

```
## [1] FALSE
```

```
length(grep("Artículo",corpus_starts))
```

```
## [1] 697
```

```
length(corpus_starts)
```

```
## [1] 697
```

Al igual que con get_titles no podemos comprobar casi nada exceptuando la existencia de la columna Artículo. Comprobaremos adicionalmente que no haya quedado ningún inicio de las cabeceras (Libro II:, Capítulo IV: ...)

```
any(corpus_docvars$Artículo=="")
```

```
## [1] FALSE
```

```
nrow(corpus_docvars)
```

```
## [1] 697
```

```
grep("(Título|TÍTULO) [IVX]+", corpus_docvars$Título, perl = TRUE)
```

```
## integer(0)
```

```
grep("(Capítulo) [IVX]+", corpus_docvars$Capítulo, perl = TRUE)
```

```
## integer(0)
```

```
grep("(Sección) [IVX]+", corpus_docvars$Sección, perl = TRUE)
```

```
## integer(0)
```

```
grep("(Libro|LIBRO) [IVX]+", corpus_docvars$Libro, perl = TRUE)
```

```
## integer(0)
```

get_html_arts().

Esta función devuelve los links de los 697 artículos a scrapear, para comprobar que son correctos solo usaremos readLines para comprobar que no devuelve error. No hace falta comprobar si alguno devuelve vacío, estos es pues después de escribirlos, la función get_content() los lee, y como ya hemos comprobado que esa función ya devolvía no vacío, se deduce que esta tampoco. Solo nos interesa el elemento 33 de readLines pues es el cque contiene el artículo.

```
setwd("./..")
links = get_html_arts()
length(links)
```

```
## [1] 697
```

```
contenido = vector("list", 697)
```

```
for (i in 1:length(articulos)){
  contenido[[i]]= readLines(links[[i]])[33]
}
```

Todo se ha ejecutado correctamente, veámos un ejemplo.

```
substr(contenido[[1]],1,300)
```

```
## [1] "/* ]]> */ </script><link rel=\"https://api.w.org/\" href=\"https://www.conceptosjuridicos.com/w
```