PRESENTACIÓN 5

CADENAS DE CARACTERES y VARIABLES TIEMPO

```
urls <- c("https://duckduckgo.com?q=Johann+Carl+Friedrich+Gauss",
      "https://search.yahoo.com/search?p=Jean+Baptiste+Joseph+Fourier",
      "http://www.bing.com/search?q=Isaac+Newton",
      "http://www.google.com/search?q=Brahmagupta")
```

Longitud de las cadenas

```
nchar(urls)
nchar(c("one",NA,1234))
nchar(as.factor(c("a","b","a","a","b","c")))
nchar(as.factor(c("a","b","a","a","b","c")),allowNA=TRUE)
```

```
#  Si tenemos una lista de tipos de archivos, y uno es una cadena vacía,
#  usamos nzchar para crear índices lógicos y así ignorar la cadena vacía

tiposArch <- c("txt","","html","txt")
nzchar(tiposArch)
tiposArch[nzchar(tiposArch)]
```

Posición de una subcadena

```
#  Usando la definición anterior del vector urls, queremos hallar las
#  posiciones de los dos puntos (:) en las URL, ya que son el delimitador
#  entre el protocolo y y el nombre del anfitrión.

dospuntos <- regexpr(":",urls)
dospuntos
dospuntos[2]
dospuntos[3]
```

Extracción o cambio de una subcadena

```
#  Extracción

protocolos <- substr(urls,1,dospuntos-1)
protocolos

#  Remplazo de cada uno de los protocolos por el protocolo mailto

mailto <- urls
substr(mailto,1,dospuntos-1) <- c("mailto","mailto","mailto")
mailto
```

```
#  Las cadenas 'https' y 'http' tiene menos caracteres que la cadena 'mailto',
#  de modo que el comando trunca la nueva cadena al sustituir la cadena original
```

Transformación de mayúsculas en minúsculas y viceversa

```
tolower(urls[1])
toupper(urls[2])
```

```
#  queremos reemplazar las ocurrencias de = con un carácter # para delimitar el
#  final de la URL y el inicio de la lista de argumentos. Supongamos que también
#  queremos reemplazar los caracteres + por espacios
```

```
chartr("=+","# ",urls)
```

```
#  el primer argumento es una cadena cuyos elementos serán cambiados, el segundo
#  argumento es una cadena que contiene los caracteres de remplazo, y el tercer
#  argumento es la cadena para cambiar
```

Rompiendo cadenas

```
#  queremos dividir cada URL en su protocolo y el resto de la información
```

```
splitURL <- strsplit(urls,":")
splitURL
splitURL[[1]]
splitURL[[1]][2]
```

Creando cadenas formateadas

```
#  El caracter %s se refieren a una cadena en la lista de argumentos, y %d se
#  refiere a un número entero a continuación de la lista de argumentos
```

```
n <- 1
calculation <- 123.0
theTitle <- sprintf("URL: %s, Count=%d",urls[n],calculation)
theTitle
```

EXPRESIONES REGULARES

```
#  examinamos los resultados de la búsqueda del delimitador = en la primera entrada
#  de nuestro vector urls definido anteriormente
```

```
loc <- gregexpr("=",urls[[1]])
loc
loc[[1]][1]
```

```
#  la función devuelve una lista. Dado que el uso de un solo corchete devuelve otra
#  lista, utilizamos corchetes dobles para asegurar que nos devuelve el elemento
#  dentro de la lista como un vector
```

```
sub("\\?.*$","",urls)

#  Observamos que se deben usar dos barras hacia atrás. Se utiliza la primera barra
#  hacia atrás para indicar que el siguiente carácter es un símbolo debe ser interpretado,
#  y si se usa una segunda barra hacia atrá, entonces significa que el par de barras se
#  han de interpretar como una sola barra hacia atrás.


## CONVERSIÓN DE CADENAS A DATOS TIEMPO

#  los argumentos del comando son la cadena a convertir seguida de la cadena de formato

laHora <- c("08:30:00 1867-07-01","18:15:00 1864-10-27")
conversion <- strptime(laHora,"%H:%M:%S %Y-%m-%d")
conversion
typeof(conversion[1])
conversion[1]-conversion[2]

#  el comando tiene un argumento para especificar zona horaria

Sys.timezone()
OlsonNames()
conversion <- strptime(laHora,"%H:%M:%S %Y-%m-%d",tz="Pacific/Auckland")
conversion
conversion[1]-conversion[2]

#  los resultados de strptime son datos tipo POSIXlt, los convertimos a POSIXct

otraHora <- as.POSIXct(conversion)
otraHora
typeof(otraHora[1])
cat(otraHora[1],"\n")

#  cuando se presentan las fechas se convierten a un formato legible para el humano
#  Esto es conveniente pero esconde el tipo de datos implícito. Es fácl perder la
#  pista de cómo el R trata realmente los valores.

#  el comando strptime no da un indicador visible cuando ocurre un error, devuelve
#  el valor NA en lugar de cada error

unaHora <- c("2014-05-05 08:00:00","2014/05/05 08:00:00")
interna <- strptime(unaHora,"%Y/%m/%d %H:%M:%S")
interna
is.na(interna)

#  otra manera de guardar fechas e información horaria en un archivo es especificando
#  la fecha y la hora en una columna. Si la información está guardada en un bastidor de
#  datos, se añade una nueva columna
```

```
archInfo <- data.frame(hora=c("2014-01-01 00:00:00",
                    "2013-12-31 23:59:50","2013-12-31 23:55:12"),
                    bienestar=c(1.0,0.9,0.8))
archInfo
archInfo$horaInterna <- strptime(archInfo$hora,"%Y-%m-%d %H:%M:%S")
archInfo
summary(archInfo)
```

CONVERSIÓN DE DATOS TIEMPO A CADENAS

```
lasHoras <- c("08:35:00 1867-07-01","18:20:00 1864-10-27")
conversion <- strptime(lasHoras,"%H:%M:%S %Y-%m-%d",tz="Canada/Eastern")
conversion
typeof(conversion)
deVuelta <- strftime(conversion,"%j - %B")
deVuelta
typeof(deVuelta[1])
```

OPERACIONES CON DATOS TIPO TIEMPO

```
anterior <- strptime("2014-01-01 00:00:00","%Y-%m-%d %H:%M:%S")
posterior <- strptime("2014-01-02 00:00:00","%Y-%m-%d %H:%M:%S")
posterior - anterior
retraso <- posterior - anterior
as.double(retraso)
anterior+retraso

#   en este ejemplo, las unidades son días, un pequeño cambio da lugar a un
#   resultado de tipo distinto

posterior <- strptime("2014-01-01 12:00:00","%Y-%m-%d %H:%M:%S")
posterior - anterior
retraso <- posterior - anterior
retraso

#   el R lleva la cuenta de qué unidades se emplean

attributes(retraso)
attr(retraso,"units")

#   especificar las unidades

as.numeric(retraso,units="weeks")
as.numeric(retraso,units="secs")

#   uso básico de difftime
```

```
retraso <- difftime(posterior,anterior,units="sec")
retraso
retraso <- difftime(posterior,anterior,units="day")
retraso

#   Las unidades disponibles son: 'auto', 'secs', 'mins', 'hours', 'days', o 'weeks'
#   El tipo de datos 'difftime' ofrece un forma conveniente de hacer algunas
#   operaciones aritméticas con las fechas

posterior <- strptime("2014-01-01 12:00:00","%Y-%m-%d %H:%M:%S")
unaHora = as.difftime(1,units="hours")
posterior + unaHora
```