# CARACTERES EN RACKET

Los caracteres son un tipo de dato. Los caracteres se utilizan para: mostrarlos en pantalla y así comunicarnos con las personas o visualizar una página Web y entenderla, hacer comparaciones para realizar búsquedas en Internet (ej.Google), en bases de datos, en archivos, formularios, en aplicaciones de escritorio, etc. En general, un caracter es un símbolo o acción asociada a una tecla de su computador, siendo diferente por ejemplo, a de A, o de ó, entre otros.

## Sintaxis

Los caracteres en **Racket** se escriben usando la notación #\<caracter>.

**Caracteres especiales**
**#\space**
**#\newline**

Son especiales pues se distinguen por representar un (1) solo caracter, pero, en su escritura tienen varios caracteres.

***Ejemplo:***
(display #\space) (display #\newline), imprimen un espacio y un salto de línea respectivamente. Hacen la misma función que (display " ") y que (newline).

**Operaciones con caracteres**
Scheme provee el procedimiento char?, el cual determina si un dato es un caracter. Tiene la siguiente sintaxis: (char? x)

***Ejemplos:***

```
(char? #\a)          ->#t
(char? 'a)           ->#f    Devuelve falso porque 'a es un símbolo, no un caracter.
(char? "arepa")      ->#f    Devuelve falso porque "arepa" es una cadena, no un caracter.
(char? #\space)      ->#t
```

Existen también procedimientos para encontrar si un carácter va antes o después de otro carácter en el código ASCII. Estos procedimientos son:

## Sintaxis

**(char=? ch1 ch2)** **;** Es ch1 el mismo caracter que ch2?
**(char<? ch1 ch2)** **;** ch1 está antes que ch2 en el alfabeto?
**(char>? ch1 ch2)** **;** ch1 está después que ch2 en el alfabeto?
**(char<=? ch1 ch2)** **;** ch1 está antes que ch2 en el alfabeto o son los mismos?
**char>=? ch1 ch2)** **;** ch1 está después que ch2 en el alfabeto o son los mismos?

Existe el modificador **–ci**, o caso insensible, es decir que no importará si hay mayúsculas o minúsculas, el lenguaje las tomará como iguales.

## Sintaxis

**(char-ci=? ch1 ch2)** **;** Lo mismo que char=? Pero insensible
**(char-ci<? ch1 ch2)** **;** Lo mismo que char<? Pero insensible
**(char-ci>? ch1 ch2)** **;** Lo mismo que char>? Pero insensible
**(char-ci<=? ch1 ch2)** **;** Lo mismo que char<=? Pero insensible

**(char-ci>=? ch1 ch2) ;** Lo mismo que char>=? Pero insensible

***Ejemplos:***

(char=? #\s #\S) -> #f Compara basándose en el codigo ASCII, donde la s minúscula está antes que la s mayúscula.

(char-ci=? #\s #\S) -> #t Devuelve verdadero porque –ci hace que no se distinga la mayúscula de la minúscula por lo tanto Racket las evalúa como iguales.

(char<? #\a #\b) -> #t Devuelve verdadero pues la "a" va antes que la "b" en el código ASCII.

(char<? #\a #\A) -> #f Devuelve falso pues las mayúsculas van antes que las minúsculas en el código ASCII.

***Ejercicios***

- Hacer una función que reciba un parámetro y devuelva verdadero si el parámetro es un caracter.
- Hacer una función ***EsVocal*** que reciba un parámetro y devuelva verdadero si el parámetro es una vocal y falso de lo contrario. No importa si la vocal es minúscula o mayúscula.
- Hacer una función que reciba un caracter e imprima "Es una vocal" si el caracter es una vocal o devuelva un mensaje de error en caso contrario. Usar la función anterior.
- Hacer una función que compare 2 caracteres y devuelva si son iguales, o si el primero va antes que el segundo en el código ASCII o si el segundo va antes que el primero.

Racket también provee procedimientos para encontrar que tipo de caracter se está evaluando. Puede ser un caracter alfabético, numérico, espacio en blanco, mayúscula o minúscula, respectivamente como se muestra a continuación:

## Sintaxis

**(char-alphabetic? ch)**
**(char-numeric? ch)**
**(char-whitespace? ch)**
**(char-upper-case? ch)**
**(char-lower-case? ch)**

***Ejemplos:***

(char-alphabetic? #\a) -> #t
(char-numeric? #\a) -> #f
(char-numeric? #\2) -> #t
(char-whitespace? #\space) -> #t
(char-upper-case? #\A) -> #t
(char-lower-case? #\A) -> #f

***Ejercicio:***

- Hacer una función que reciba un parámetro y retorne "es alfabético" si el parámetro es un carácter alfabético, o "es numérico" si es numérico, o "es un espacio en blanco" si es un espacio en blanco y lo mismo si es mayúscula y minúscula.

También hay procedimientos para convertir caracteres a enteros en el conjunto de datos del código ASCII y viceversa. Estos procedimientos son:

## Sintaxis

**(char->integer ch)**
**(integer->char n)**

***Ejemplos:***

(char->integer #\3) ->51

(char->integer #\a)  -> 97
(integer->char 97)  -> #\a
(char->integer #\@) -> 64
(integer->char 56)  -> #\8
(integer->char (char->integer #\a)); #\a

***Ejercicio:***

- Hacer una función que reciba un parámetro, si el parámetro es un caracter devolver el número que corresponda en la tabla del código ASCII y si es un número devolver el caracter que corresponda en la tabla. **Nota:** la función ***(number? n)***, retorna verdadero si ***n*** es un número y falso de lo contrario.
- Hacer una función que reciba un parámetro, si el parámetro es un carácter presente un mensaje que indique si está en mayúscula o si está en minúscula, para ello se debe hacer uso de la evaluación del caraceter a través de su código ASCII.

Racket también provee dos procedimientos para convertir caracteres de minúsculas a mayúsculas y viceversa. Ellos son:

## Sintaxis

**(char-upcase ch)**
**(char-downcase ch)**

***Ejemplos:***

(char-upcase #\a)  -> #\A
(char-downcase #\B)  -> #\b
(char-upcase #\b)  -> #\B
(char-upcase (char-downcase #\B))  -> #\B

***Ejercicio:***

- Hacer una función que reciba un parámetro. Si el parámetro es un caracter alfabético, determinar si está en minúscula y pasarlo a mayúscula y retornar este valor. Hacer lo mismo en caso contrario.
- Hacer una función recursiva que solicite al usuario S de SI o N de NO para saber si desea terminar la ejecución de la función, de forma que se quede solicitando la respuesta mientras esta no sea S, además, si la respuesta es diferente de S o N que indique ERROR. Cuando el usuario por fin diga S, la función entregará una estadística de:
    - Cuantas veces tecleo N
    - Cuantas veces tecleo Error
    - Cuantas veces el Error fue un número
    - Cuantas veces tecleo n
- Hacer una función que reciba un símbolo, determine si es caracter y luego lo muestre ENCRIPTADO.