Expresiones II - Strings

Lic. Natalia Colussi

- **Cátedra Programación I (LCC)**
- **Cátedra Programación (LM y PM)**

Agosto 2021

¿Qué son las strings?

- Primero, ante todo, definiciones.

Es una secuencia de caracteres, las cuales en general, en los lenguajes de programación aparecen encerradas entre comillas

- Ejemplos:
  - “Hola mundo”
  - “red”
  - “solid”
  - “a”
  - “”

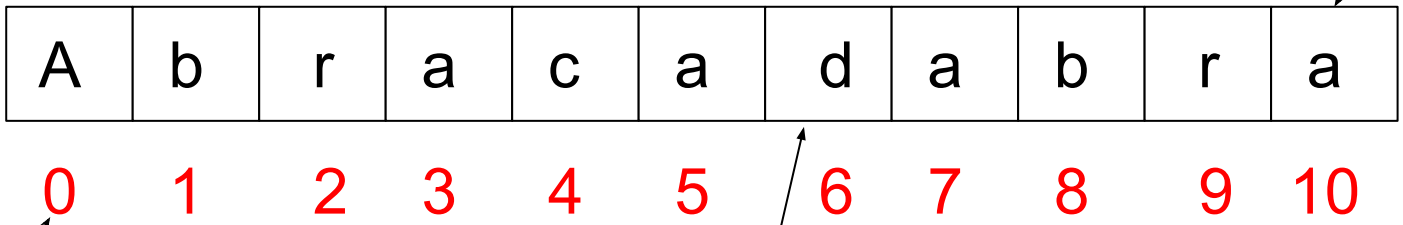
Las strings nos permiten representar la información de una manera clara

Utilizamos las strings de longitud 1 para referirnos a los caracteres

Cadena vacía (comillas sin espacios)

¿ “” = “ ”?

Nos metemos dentro de una string

- “Abracadabra”
- 

La cadena tiene longitud 11, esto representa la cantidad de caracteres de la palabra o string.

Siempre el último carácter de la palabra está ubicado en el índice (longitud de la cadena -1)

Primera posición de la cadena de caracteres. Siempre es 0 (cero)

Cada casilla encierra un carácter

Índices de la string

Aritmética de Strings

- Conjunto de operaciones que se aplican sobre *strings*.
- Algunos ejemplos que nos ofrece **Racket**.

★ `string-append`
★ `string-length`
★ `string-ith`
★ `number->string`
★ `substring`
★ `string-downcase`
★ `string-upcase`

Vamos a visitar cada una de estas operaciones para entender cómo funcionan y cómo se usan.

TAREA: Reveer estas operaciones, proponer ejemplos propios de uso, y si tenemos problemas anotarlos para consultar en el horario de práctica.

string-append

```
Language: Intermediate Student [custom]; memory limit: 128 MB.
Teachpacks: 2htdp/image.rkt and universe.rkt.
> (string-append "Estoy" "estudiando" "en" "clase")
"Estoyestudiandoenclase"
> (string-append "Hola " "estoy " "estudiando " "en "
"clase")


"Hola estoy estudiando en clase"
> (string-append "")
""
>
All expressions are covered                    Show next time?
Intermediate Student custom ▾        12:2      493.81 MB
```

```
(string-append s ...) → string
  s : string
```

Concatenates the characters of several strings.

```
> (string-append "hello" " " "world" " " "good bye")
"hello world good bye"
```

- Concatena dos o más cadenas de caracteres, es decir, toma dos o más cadenas de caracteres y produce una cadena que es la unión consecutiva de las cadenas dadas.
- Si no dejo espacio en blanco al concatenar no va a separa las palabras.
- Tengo dos cadenas, “Abra” de longitud 4 y “Cadabra” de longitud 7, ¿Cuál será a longitud de la concatenación?

string-length

```
(string-length s) → nat
  s : string
```

Determines the length of a string.

```
> (string-length "hello world")
11
```

```
Welcome to DrRacket, version 7.8 [cs].
Language: Intermediate Student [custom]; memory limit: 128 MB.
Teachpacks: 2htdp/image.rkt and universe.rkt.
> (string-length "Abracadabra")
11
> (string-length (string-append "Abra" "Cadabra"))
11
>
```

- Determina la longitud de una cadena de caracteres, es decir, la cantidad de caracteres que tiene la palabra.
- ¿Cuál será la longitud de la cadena vacía?
- ¿Podrá ser el resultado de esta operación un valor negativo?

string-ith

```
Welcome to DrRacket, version 7.8 [cs].
Language: Intermediate Student [custom]; memory limit: 128 MB.
Teachpacks: 2htdp/image.rkt and universe.rkt.
> (string-ith "Abracadabra" 0)
"A"
> (string-ith "Abracadabra" 10)
"a"
> (string-ith "Abracadabra" 3)
"a"
>
```

```
(string-ith s i) → 1string?
  s : string
  i : natural-number
```

Extracts the *i*th 1-letter substring from *s*.

```
> (string-ith "hello world" 1)
"e"
```

- string-ith nos devuelve el carácter que está en la posición i-ésima indicada como segundo argumento de la función.
- ¿Puede ser un índice un valor negativo?
- ¿Qué pasa si indexo más allá de lo posible? Es decir, ¿Qué sucede si accedo a una posición que no existe dentro de una string?

number->string

```
Welcome to DrRacket, version 7.8 [cs].
Language: Intermediate Student [custom]; memory limit: 128 MB.
Teachpacks: 2htdp/image.rkt and universe.rkt.
> (number->string 102)
"102"
> (number->string 1)
"1"
> (number->string -1)
"-1"
> (number->string (sqrt 2))
"1.4142135623730951"
>
```

```
(number->string x) → string
  x : number
```

Converts a number to a string.

```
> (number->string 42)
"42"
```

- Convierte un número en una cadena de caracteres.
- ¿Qué pasa si su argumento no es un número?, por ejemplo, ¿si le pasamos la string “Hola” a esta operación? o ¿números y letras?
- ¿Existe la operación opuesta, es decir, le paso una string y me devuelve un número?

substring

```
Welcome to DrRacket, version 7.8 [cs].
Language: Intermediate Student [custom]; memory limit: 128 MB.
Teachpacks: 2htdp/image.rkt and universe.rkt.
> (substring "Abracadabra" 3 6)
"aca"
> (substring "Abracadabra" 3)
"acadabra"
> (substring "Abracadabra" 0)
"Abracadabra"
>
```

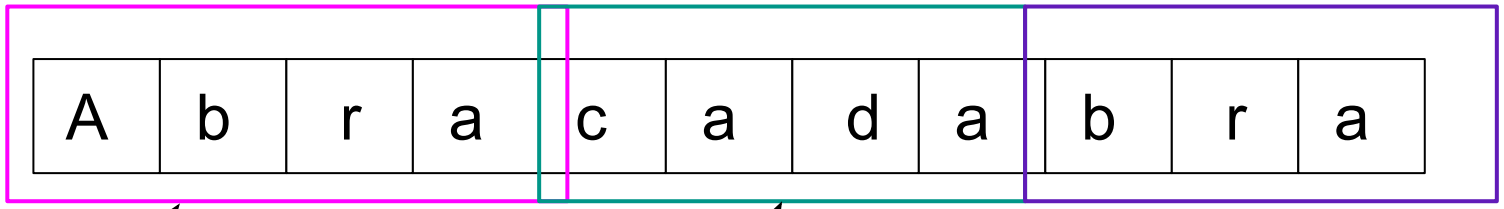
```
(substring s i j) → string                                    pro
  s : string
  i : natural-number
  j : natural-number
```

Extracts the substring starting at *i* up to *j* (or the end if *j* is not provided).

```
> (substring "hello world" 1 5)
"ello"
> (substring "hello world" 4)
"o world"
```

- Realiza una extracción de un “segmento” de una cadena de caracteres desde el índice “i” hasta el índice “j-1” incluido.
- El índice final puede omitirse, en este caso se interpreta que se desea extraer hasta el final de la cadena de caracteres.
- ¿Qué sucede si los índice están invertidos?, es decir, ¿El primer índice es menor que el segundo?

Ejemplos de uso específicos de substring



PREFIJO, siempre una substring que comienza desde 0

(substring “Abracadabra” 0 4)

SEGMENTO: una substring que no es el prefijo, ni el sufijo de la cadena

(substring “Abracadabra” 4 8)

SUFIJO: una substring cuyo extremo final coincide con el de la cadena original.

(substring “Abracadabra” 8)

Combinados ...

- Recordemos: la composición de funciones es lo que nos permite computar en los lenguajes funcionales. Nos permite alcanzar el objetivo deseado para nuestro programa.

- Aprendimos expresiones numéricas y expresiones sobre strings, ¿podremos componerlas?

  Sí, siempre que los tipos se correspondan.

Ejemplos expresiones combinadas

- `(+ (string-length “Hello world”) 40)`
- `(+ (string-length (number->string 42)) 2)`
- `(string-append (substring “Hola Mundo” 0 4)`
  `(substring “Abracadabra 4)`
  `(string->number “25”)`
- `(string-append`
  `(number->string (strlength “Abracadabra”))`
  `“Abracadabra”)`

Evaluaciones paso a paso

- ```
  (string-append
        (number->string (strlength “Abracadabra”))
        “Abracadabra”)
  ```
- vamos a trabajar la expresión en un editor y luego chequearla en Dr. Racket
  ```

Tarea:

- Con lo visto pueden avanzar resolviendo práctica 0, ejercicios 3.1