Lenguajes de Programación

Introducción a Racket

Tipos básicos y definición de funciones

Manuel Soto Romero

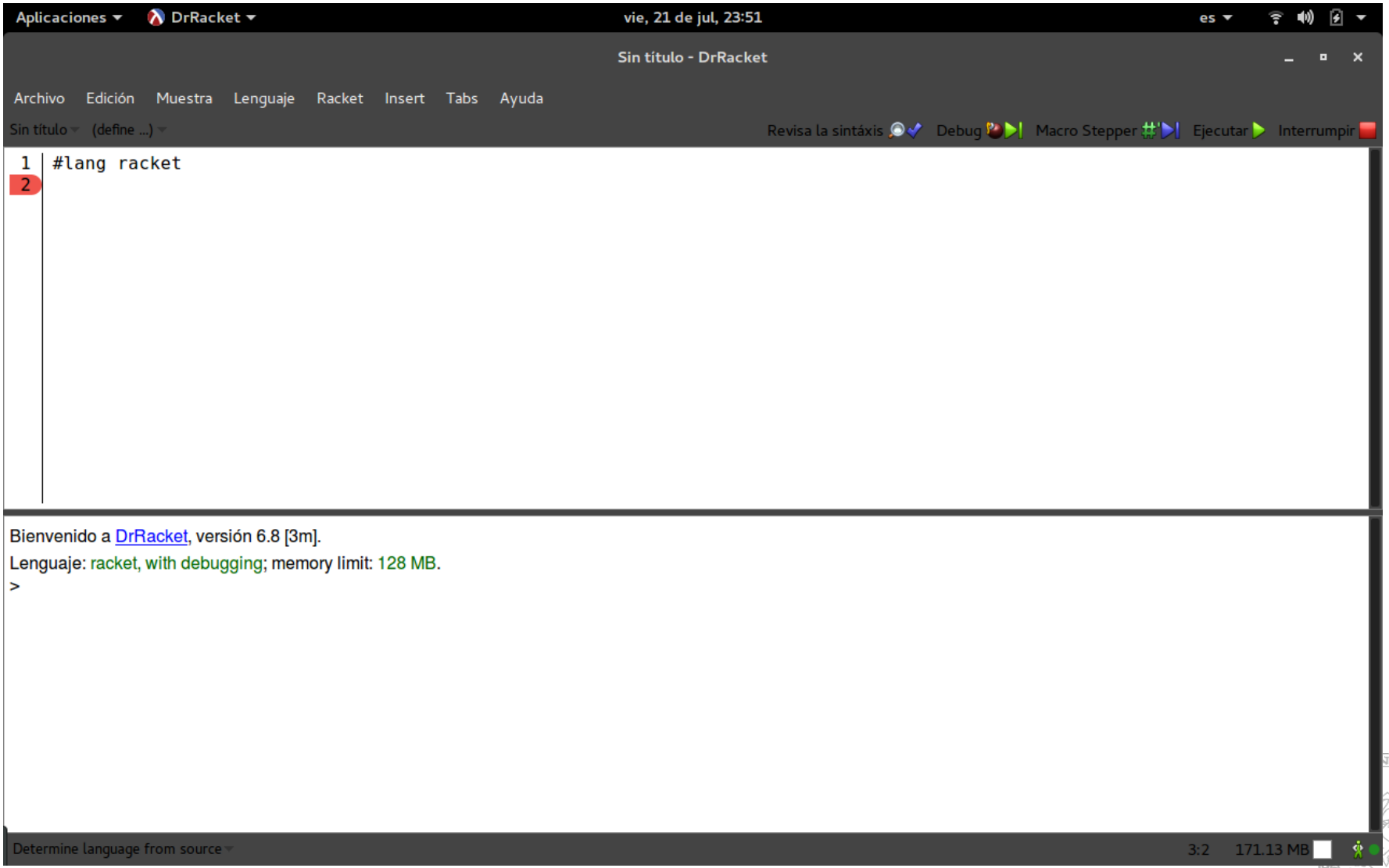
Universidad Nacional Autónoma de México
Facultad de Ciencias

9 de febrero de 2018

Racket

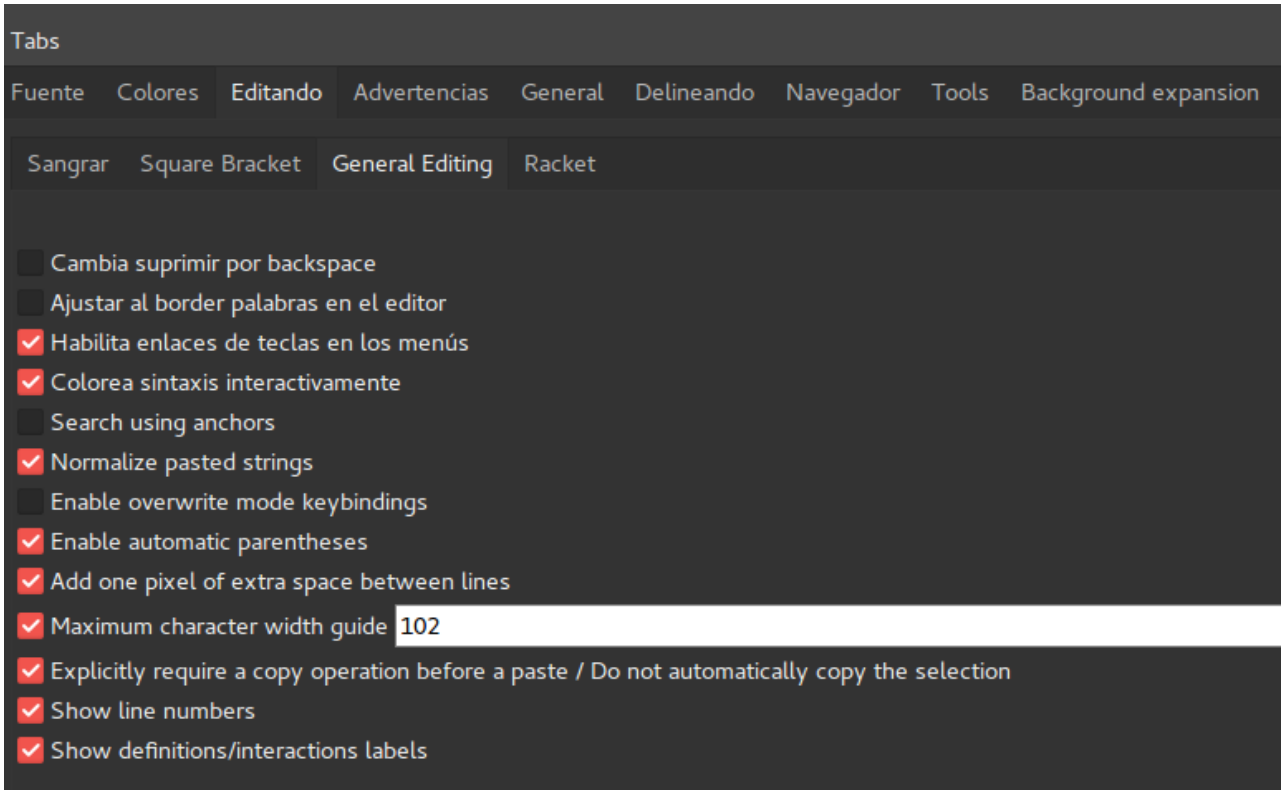
- Es un lenguaje de propósito general de la familia de lenguajes Lisp.
- Fue diseñado como una plataforma para la creación, diseño e implementación de lenguajes de programación.
- Incluye muchos dialectos, en el curso usaremos la variante `plai` que acompaña al libro de texto que usamos.
- Para escribir y ejecutar programas, usaremos el IDE DrRacket.

DrRacket


Aplicaciones
DrRacket
vie, 21 de jul, 23:51
es
Sin título - DrRacket
Archivo
Edición
Muestra
Lenguaje
Racket
Insert
Tabs
Ayuda
Sin título
(define ...)
Revisa la sintáxis
Debug
Macro Stepper
Ejecutar
Interrumpir
1
2
#lang racket
Bienvenido a DrRacket, versión 6.8 [3m].
Lenguaje: racket, with debugging; memory limit: 128 MB.
>
Determine language from source
3:2
171.13 MB

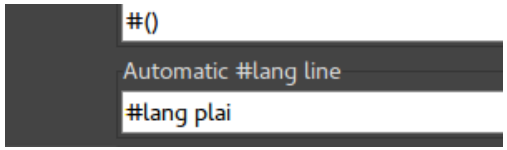
Configuraciones

CTRL + ;



Lenguaje por defecto

CTRL + I



Programming Languages Application and Interpretation

Tipos básicos

BOOLEANOS

Constantes lógicas para verdadero y falso

CADENAS

Secuencias de caracteres, se delimitan por comillas dobles

NÚMEROS

Exactos e Inexactos

SÍMBOLOS

Valores atómicos, usan el mecanismo de *citado* (`quote`)

CARACTERES

Codificación Unicode, se delimitan por #\

Estas primitivas pueden probarse en el área de interacción de DrRacket

Ejemplos - Tipos básicos

```
> #t
```

Ejemplos - Tipos básicos

```
> #t
#t
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
> 7
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
> 7
7
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
> 7
7
> 1/7
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
> 7
7
> 1/7
1/7
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
> 7
7
> 1/7
1/7
> 1+7i
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
> 7
7
> 1/7
1/7
> 1+7i
1+7i
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
> 7
7
> 1/7
1/7
> 1+7i
1+7i
```

```
> 5.5
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
> 7
7
> 1/7
1/7
> 1+7i
1+7i
```

```
> 5.5
5.5
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
> 7
7
> 1/7
1/7
> 1+7i
1+7i
```

```
> 5.5
5.5
> 5.25e8
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
> 7
7
> 1/7
1/7
> 1+7i
1+7i
```

```
> 5.5
5.5
> 5.25e8
5.25e8
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
> 7
7
> 1/7
1/7
> 1+7i
1+7i
```

```
> 5.5
5.5
> 5.25e8
5.25e8
> 98245798457982758927509823 47
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
> 7
7
> 1/7
1/7
> 1+7i
1+7i
```

```
> 5.5
5.5
> 5.25e8
5.25e8
> 98245798457982758927509823 47
98245798457982758927509823 47
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
> 7
7
> 1/7
1/7
> 1+7i
1+7i
```

```
> 5.5
5.5
> 5.25e8
5.25e8
> 98245798457982758927509823 47
98245798457982758927509823 47
> #\a
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
> 7
7
> 1/7
1/7
> 1+7i
1+7i
```

```
> 5.5
5.5
> 5.25e8
5.25e8
> 98245798457982758927509823 47
98245798457982758927509823 47
> #\a
#\a
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
> 7
7
> 1/7
1/7
> 1+7i
1+7i
```

```
> 5.5
5.5
> 5.25e8
5.25e8
> 98245798457982758927509823 47
98245798457982758927509823 47
> #\a
#\a
> #\E
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
> 7
7
> 1/7
1/7
> 1+7i
1+7i
```

```
> 5.5
5.5
> 5.25e8
5.25e8
> 98245798457982758927509823 47
98245798457982758927509823 47
> #\a
#\a
> #\E
#\E
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
> 7
7
> 1/7
1/7
> 1+7i
1+7i
```

```
> 5.5
5.5
> 5.25e8
5.25e8
> 98245798457982758927509823 47
98245798457982758927509823 47
> #\a
#\a
> #\E
#\E
> ”Hola Mundo”
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
> 7
7
> 1/7
1/7
> 1+7i
1+7i
```

```
> 5.5
5.5
> 5.25e8
5.25e8
> 98245798457982758927509823 47
98245798457982758927509823 47
> #\a
#\a
> #\E
#\E
> ''Hola Mundo''
''Hola Mundo''
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
> 7
7
> 1/7
1/7
> 1+7i
1+7i
```

```
> 5.5
5.5
> 5.25e8
5.25e8
> 98245798457982758927509823 47
98245798457982758927509823 47
> #\a
#\a
> #\E
#\E
> ''Hola Mundo''
''Hola Mundo''
> 'manzana
```

Ejemplos - Tipos básicos

```
> #t
#t
> #f
#f
> true
#t
> false
#f
> -1
-1
> 7
7
> 1/7
1/7
> 1+7i
1+7i
```

```
> 5.5
5.5
> 5.25e8
5.25e8
> 98245798457982758927509823 47
98245798457982758927509823 47
> #\a
#\a
> #\E
#\E
> ''Hola Mundo''
''Hola Mundo''
> 'manzana
'manzana
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
> (+ 1 (- 3 4))
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
> (+ 1 (- 3 4))
0
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
> (+ 1 (- 3 4))
0
> (sqrt -1)
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
> (+ 1 (- 3 4))
0
> (sqrt -1)
0+1i
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
> (+ 1 (- 3 4))
0
> (sqrt -1)
0+1i
> (or (< 5 4) (equal? 1
(- 6 5)))
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
> (+ 1 (- 3 4))
0
> (sqrt -1)
0+1i
> (or (< 5 4) (equal? 1
(- 6 5)))
#t
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
> (+ 1 (- 3 4))
0
> (sqrt -1)
0+1i
> (or (< 5 4) (equal? 1
(- 6 5)))
#t
> (and (not (zero? 10))
(+ 1 2 3))
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
> (+ 1 (- 3 4))
0
> (sqrt -1)
0+1i
> (or (< 5 4) (equal? 1
(- 6 5)))
#t
> (and (not (zero? 10))
(+ 1 2 3))
6
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
> (+ 1 (- 3 4))
0
> (sqrt -1)
0+1i
> (or (< 5 4) (equal? 1
(- 6 5)))
#t
> (and (not (zero? 10))
(+ 1 2 3))
6
```

```
> (string-append ''Ho'' ''la'')
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
> (+ 1 (- 3 4))
0
> (sqrt -1)
0+1i
> (or (< 5 4) (equal? 1
(- 6 5)))
#t
> (and (not (zero? 10))
(+ 1 2 3))
6
```

```
> (string-append ''Ho'' ''la'')
''Hola''
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
> (+ 1 (- 3 4))
0
> (sqrt -1)
0+1i
> (or (< 5 4) (equal? 1
(- 6 5)))
#t
> (and (not (zero? 10))
(+ 1 2 3))
6
```

```
> (string-append ”Ho” ”la”)
”Hola”
> (string-length ”hola mundo”)
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
> (+ 1 (- 3 4))
0
> (sqrt -1)
0+1i
> (or (< 5 4) (equal? 1
(- 6 5)))
#t
> (and (not (zero? 10))
(+ 1 2 3))
6
```

```
> (string-append ”Ho” ”la”)
”Hola”
> (string-length ”hola mundo”)
10
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
> (+ 1 (- 3 4))
0
> (sqrt -1)
0+1i
> (or (< 5 4) (equal? 1
(- 6 5)))
#t
> (and (not (zero? 10))
(+ 1 2 3))
6
```

```
> (string-append ”Ho” ”la”)
”Hola”
> (string-length ”hola mundo”)
10
> (substring ”Apple” 3)
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
> (+ 1 (- 3 4))
0
> (sqrt -1)
0+1i
> (or (< 5 4) (equal? 1
(- 6 5)))
#t
> (and (not (zero? 10))
(+ 1 2 3))
6
```

```
> (string-append ”Ho” ”la”)
”Hola”
> (string-length ”hola mundo”)
10
> (substring ”Apple” 3)
”le”
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
> (+ 1 (- 3 4))
0
> (sqrt -1)
0+1i
> (or (< 5 4) (equal? 1
(- 6 5)))
#t
> (and (not (zero? 10))
(+ 1 2 3))
6
```

```
> (string-append ''Ho'' ''la'')
''Hola''
> (string-length ''hola mundo'')
10
> (substring ''Apple'' 3)
''le''
> (string->symbol ''Apple'')
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
> (+ 1 (- 3 4))
0
> (sqrt -1)
0+1i
> (or (< 5 4) (equal? 1
(- 6 5)))
#t
> (and (not (zero? 10))
(+ 1 2 3))
6
```

```
> (string-append ''Ho'' ''la'')
''Hola''
> (string-length ''hola mundo'')
10
> (substring ''Apple'' 3)
''le''
> (string->symbol ''Apple'')
'Apple
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
> (+ 1 (- 3 4))
0
> (sqrt -1)
0+1i
> (or (< 5 4) (equal? 1
(- 6 5)))
#t
> (and (not (zero? 10))
(+ 1 2 3))
6
```

```
> (string-append ”Ho” ”la”)
”Hola”
> (string-length ”hola mundo”)
10
> (substring ”Apple” 3)
”le”
> (string->symbol ”Apple”)
’Apple
> (display ”hola\nmundo”)
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
> (+ 1 (- 3 4))
0
> (sqrt -1)
0+1i
> (or (< 5 4) (equal? 1
(- 6 5)))
#t
> (and (not (zero? 10))
(+ 1 2 3))
6
```

```
> (string-append ”Ho” ”la”)
”Hola”
> (string-length ”hola mundo”)
10
> (substring ”Apple” 3)
”le”
> (string->symbol ”Apple”)
’Apple
> (display ”hola\nmundo”)
hola
mundo
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
> (+ 1 (- 3 4))
0
> (sqrt -1)
0+1i
> (or (< 5 4) (equal? 1
(- 6 5)))
#t
> (and (not (zero? 10))
(+ 1 2 3))
6
```

```
> (string-append ”Ho” ”la”)
”Hola”
> (string-length ”hola mundo”)
10
> (substring ”Apple” 3)
”le”
> (string->symbol ”Apple”)
’Apple
> (display ”hola\nmundo”)
hola
mundo
> (+ 1 ”Hola”)
```

Ejemplos - Funciones predefinidas

```
> (+ 1 2)
3
> (- 1 1/4)
3/4
> (+ 1 (- 3 4))
0
> (sqrt -1)
0+1i
> (or (< 5 4) (equal? 1
(- 6 5)))
#t
> (and (not (zero? 10))
(+ 1 2 3))
6
```

```
> (string-append ”Ho” ”la”)
”Hola”
> (string-length ”hola mundo”)
10
> (substring ”Apple” 3)
”le”
> (string->symbol ”Apple”)
’Apple
> (display ”hola\nmundo”)
hola
mundo
> (+ 1 ”Hola”)
+: contract violation
```

Definición de funciones

Al definir funciones se recomienda seguir los siguientes pasos:

1. Entender qué tiene que hacer la función.
2. Escribir la descripción de la función.
3. Escribir su contrato, o sea su tipo (cuantos parámetros, de qué tipo, qué regresa).
4. Escribir las pruebas asociadas a esta función sobre los posibles datos de entrada (casos significativos).
5. Implementar el cuerpo de la función.

Ejemplo - Definición de funciones

```
;; Función que calcula el área de un círculo dado su
;; diámetro
;; area: number -> number
(define(area diametro)
    (* pi (/ diametro 2) (/ diametro 2)))

(test(area 10) 78.53)
(test(area 4) 12.56)
(test(area 16) 201.06)
```

Asignaciones locales

Permiten guardar cálculos u otras funciones en variables para usarlas en una determinada región del programa.

```
;; Función que calcula el área de un círculo dado su
;; diámetro
;; area: number -> number
(define(area diametro)
    (let([r (/ diametro 2)])
        (* pi r r)))
```

Asignaciones locales anidadas

```
(let ([a 2] [b (+ a a)])
    b)
```

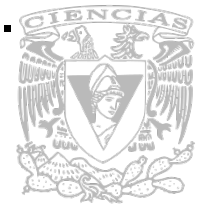
Este tipo de expresiones no funcionan. Necesitamos anidar:

```
(let ([a 2])
    (let ([b (+ a a)])
        b))
```

O usar la versión con azúcar sintáctica, `let*`:

```
(let* ([a 2] [b (+ a a)])
    b)
```

Existe otra variante para expresiones recursivas, llamada `letrec`.



Condicionales

```
;; Función que calcula el valor absoluto de un número
;; absoluto: number -> number
(define(absoluto n)
    (if(< n 0)
        (* n -1)
        n))

;; Función que regresa la representación en cadena de un mes
;; pasado como número
;; mes: number -> string
(define(mes n)
    (cond
        [(= n 1) ''Enero'']
        [(= n 2) ''Febrero'']
        ...
        [else (error 'mes ''Número inválido'']))
```