Introducción a la Programación Funcional
Nota de clase 02: Definición de Funciones*

Karla Ramírez Pulido Manuel Soto Romero

21 de enero de 2020

Adicional a las funciones predefinidas sobre los tipos de datos básicos, es posible definir funciones propias. El diseño de funciones es un proceso que debe seguir una serie de pasos ordenados y bien especificados para garantizar su correcto funcionamiento, a continuación se presenta un método de definición de funciones y se presentan algunos ejemplos junto a su solución.

2.1. Método de definición de funciones

Cuando se define una función que cumple cierto objetivo, se recomienda seguir los siguientes pasos en el orden en que se indica[1]:

- Entender lo que la función tiene que hacer.
- Escribir la descripción de la función a través de los comentarios.
- Escribir su contrato, es decir, su tipo (cuántos parámetros recibe, de qué tipo son, cuál es el valor de regreso) a través de los comentarios.
- Escribir pruebas asociadas a esta función, sobre los distintos posibles datos de entradas que pueda tener (casos significativos).
- Finalmente, implementar el cuerpo de la función.

Para especificar el contrato y la descripción de la función, se usan comentarios. En Racket existen dos tipos de comentarios:

De una línea

```
;; Comentario de una línea
```

De varias líneas

```
#| Este es un comentario
   de varias líneas |#
```

Por otro lado, escribir pruebas, obliga al programador a definir desde un inicio cómo se va a usar la función. Para definir pruebas, simplemente se escriben casos significativos y se especifica qué deben devolver. Por ejemplo, sabemos que la suma de 1 con 2 es 3, por lo tanto se escribe la prueba:

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

Finalmente, para definir una función, se usa la siguiente sintaxis:

```
(define (<nombre> <parámetro>*)
   <cuerpo>)
```

Se debe especificar un nombre para la función, una secuencia de parámetros de entrada separados por espacios (puede no haber parámetros) y un cuerpo. Es importante recordar, que las definiciones de funciones deben realizarse dentro de un archivo con extensión `.rkt` o escribirlas directamente en el área de definiciones de DrRacket.

*Adaptación de las Notas de Laboratorio del Curso de Lenguajes de Programación.

[1] `https://users.dcc.uchile.cl/~etanter/preplai/defun.html`

2.2. Ejemplos de definición de funciones

Ejemplo 2.1. Definir la función `promedio-3` tal que `(promedio-3 x y z)` es el promedio de los números `x`, `y` y `z`. Por ejemplo:

```
(promedio-3 1 3 8)  = 4
(promedio-3 -1 0 7) = 2
(promedio-3 -3 0 3) = 0
```

Solución:

```
1 ;; Función que calcula el promedio de tres números.
2 ;; promedio-3: number number number → number
3 (define (promedio-3 x y z)
4    (/ (+ x y z) 3))
```

Código 1: Promedio de tres números

□

Ejemplo 2.2. Definir la función `suma-monedas` tal que `(sumaMonedas a b c d e)` es la suma de los pesos correspondiente a monedas de 50 centavos, 1 peso, 2 pesos, 5 pesos y 10 pesos respectivamente. Por ejemplo:

```
(suma-monedas 0 0 0 0 1) = 10
(suma-monedas 0 0 8 0 3) = 46
(suma-monedas 1 1 1 1 1) = 18.50
```

Solución:

```
1 ;; Función que calcula la suma de los pesos correspondiente a monedas
2 ;; de 50 centavos, 1 peso, 2 pesos, 5 pesos y 10 pesos
3 ;; respectivamente.
4 ;; suma-monedas: number number number number number → number
5 (define (suma-monedas a b c d e)
6    (+ (* a 0.5) (* b 1) (* c 2) (* d 5) (* e 10)))
```

Código 2: Suma de monedas

□

Ejemplo 2.3. Definir la función `volumen-esfera` tal que `(volumen-esfera r)` calcula el volumen de una esfera de radio `r`. Por ejemplo:

```
(volumen-esfera 10) = 4188.7902
```

Solución:

```
1 ;; Función que calcula el volumen de la esfera de radio r.
2 ;; volumen-esfera: number → number
3 (define (volumen-esfera r)
4    (* 4/3 pi (expt r 3)))
```

Código 3: Volumen de esfera

□

Ejemplo 2.4. Definir la función `area-circulo` tal que `(area-circulo d)` calcula el área de un círculo de diámetro `d`. Por ejemplo:

```
(area-circulo 10) = 78.53
(area-circulo 4)  = 12.56
(area-circulo 16) = 201.06
```

Solución:

```
1 ;; Función que calcula el área de un círculo dado su diámetro.
2 ;; area-circulo: number → number
3 (define (area-circulo d)
4    (* pi (/ d 2) (/ d 2)))
```

Código 4: Área de círculo

□

Referencias

[1] Matthias Felleisen, Conrad Barski, et.al., *Realm of Racket*, No Starch Press, 2013.

[2] Matthias Felleisen, Robert B. Findler, et.al., *How to Desing Programs*, MIT Pres, 2019.

[3] Matthew Flatt, Robert B. Findler, et. al., *The Racket Guide*, Versión 7.5

[4] Éric Tanter, *PrePLAI: Scheme y Programación Funcional*, 2014.