Programación Declarativa

Ingeniería Informática
Cuarto curso. Primer cuatrimestre

Escuela Politécnica Superior de Córdoba
Universidad de Córdoba
Curso académico: 2023 – 2024

Práctica número 1.- Introducción al lenguaje Scheme

- **Observaciones:**
  - Sólo se han de presentar los ejercicios marcados con un **asterisco (*)**, que deberán estar **contenidos en un mismo fichero**.

  - **IMPORTANTE:**
    - Todas las funciones deberán tener un comentario de cabecera con la siguiente estructura:
      - Nombre de la función
      - Objetivo
      - Descripción de la solución (salvo que se deduzca de forma inmediata)
      - Significado de los parámetros de entrada.
      - Significado del resultado que devuelve.
      - Funciones auxiliares a las que llama.
    - Ejemplos de ejecución de las funciones
      - Después de cada función, se debe poner unos o varios comentarios con ejemplos de ejecución de dicha función.
      - Por ejemplo, si la función es *(cuadrado x)*
        ;; (cuadrado 2)
        ;; (cuadrado (cuadrado 2))

1. **Constantes y literales**: teclea las siguientes constantes y literales (creados con la forma especial **quote** o con la comilla simple) y comprueba el resultado devuelto por el intérprete de *Scheme*:

```
; Los comentarios comienzan con el símbolo de "punto y coma"

#t                  ;; constante lógica de verdadero
3                   ;; número entero
20.5                ;; número real
"ejemplo de cadena" ;; se utilizan comillas dobles para delimitar las cadenas
"dato               ;; no debes olvidar las comillas de cierre
'dato               ;; se utiliza la comilla simple para crear un literal

(quote dato)        ;; también se puede utilizar quote para crear un literal
dato                ;; la variable dato no es un literal
                    ;; y producirá un error porque posee no todavía un valor
'#t                 ;; las constantes lógicas también son literales
(quote #t)
'3                  ;; los números también son literales
(quote 3)
```

'20.5
(quote 20.5)
*(**quote** "ejemplo de cadena") ;; una cadena también es un literal*

(+ 2 3) ;; expresión aritmética con notación prefija
' (+ 2 3) ;; la expresión aritmética se convierte en un literal y "no" se evalúa

*(**quote** (+ 2 3)) ;; la expresión aritmética se convierte en un literal y "no" se evalúa*
'(a b c) ;; lista de literales
(quote '(a b c)) ;; otra forma de crear una lista de literales
'(Ana Luis Juan) ;; lista de literales
(quote (Ana Luis Juan)) ;; otra forma de crear una lista de literales

2. Teclea las siguientes expresiones aritméticas y comprueba los resultados.

; **Siempre** se debe **separar** el operador de los argumentos
(+ 2 3)

;; Si no se separa el operador del argumento, se producirá un **error**
***(+2** 3)*

(+ 0.1)	*(+ 0.001)*	*(+ 0.00000001)*	*(+ 3)*
(+ 3 4)	*(+ 3 4 5)*	*(+ 3 4.)*	*(+ 3 4.0)*
(+)			
(- 2)	*(- 10 2)*	*(- 10 3 1)*	*(- 10 3. 1)*
(2)*	*(* 2 3 4)*	*(* 2.0 3 4)*	*(*)*
(/ 5)	*(/ 5.)*	*(/ 10 2)*	*(/ 8 3)*
(/ 8. 3)	*(/ 8 3.0)*		

;; Aproximación racional al número π
(/ 355 113)

;; Aproximación al número π con seis decimales exactos.
(/ 355.0 113)

;; Se divide el primer argumento por el producto de los demás
(/ 60 3 5 4)

;; Combinación de operadores
(/ (9 4 3) (+ 3 2))*

;; Expresión "sangrada" con tabuladores: más legible
```
(/
   (* 9 4 3)
   (+ 3 2)
)
```

3. Escribe la siguiente expresión aritmética con notación **prefija**:
   - $\frac{-b \pm \sqrt{b^2-4ac}}{2a}$

4. Utiliza la forma especial **define** para declarar las siguientes variables y asignarles los

valores que se indican:

Variable	Valor
iva	18
mayor-edad	18
meses	12
x	2.5
y	-12.3
z	2 x + y^3
partido1	36.5
partido2	30.75
blanco	2.55
nulo	0.34
;; comprueba si el intérprete admite variables acentuadas	
abstención	100 - partido1 - partido2 - blanco – nulo
celsius	19.5
fahrenheit	32.0 + (9.0/5.0) celsius

5. ¿Qué ocurre si se aplica **set!** sobre una variable no definida previamente?
   Por ejemplo:
   *(**set!** votantes 23732)*

6. Define las siguientes variables y escribe en *Scheme* las expresiones asociadas a las **funciones matemáticas predefinidas** que se indican:

Variable	Valor
a	1
b	2
c	-3
pi	(**acos** -1.0)

Función	Significado	Ejemplo
(abs x)	Valor absoluto de x	*abs (a^2 - b^2)*
(sqrt x)	Raíz cuadrada de x	$\sqrt{b^2-4ac}$
(sqr x)	Cuadrado de x	*$(3a-2b+c)^2$*
(exp x)	Exponencial de x	*e^{2a}*
(log x)	Logaritmo neperiano de x	*log(e^a)*
(expt x y)	Potencia: x^y	*$(2a-b)^c$*
(sin x)	Seno de x	*sin(2 pi)*
(cos x)	Coseno de x	*cos(pi/2)*
(tan x)	Tangente de x	*tan(2 pi)*
(asin x)	Arco seno de x	*asin(- 0.5)*
(acos x)	Arco coseno de x	*acos(0.5)*
(atan x)	Arco tangente de x	*atan(1.0)*
(atan x y)	Arco tangente de x/y	*atan(a/b)*
(max x_1 x_2 …)	Máximo de x_1 x_2 …	*max(a,b,c)*
(min x_1 x_2 …)	Mínimo de x_1 x_2 …	*min(2a,3b,4c)*
(gcd x_1 x_2 …)	Máximo común divisor	*gcd(12,15,-18)*
(lcm x_1 x_2 …)	Mínimo común múltiplo	*lcm(12,15,-18)*

Función	Significado	Ejemplo
(floor x)	Mayor entero no más grande que x	*floor(-2.7)*
		floor(7.5)
(ceiling x)	Menor entero no más pequeño que x	*ceiling(-2.7)*
		ceiling(7.5)
(truncate x)	Entero más próximo a x cuyo valor absoluto no es más grande que el valor absoluto de x	*truncate(-2.7)*
		truncate(7.5)
(round x)	Entero más próximo a x; redondeando a un número par si x está justo entre dos enteros.	*round(-2.5)*
		Round(7.5)
(modulo x y)	Resto de la división entera (Signo del divisor)	*modulo (12, 5)*
		modulo(12, -5)
		modulo(-12, 5)
(quotient x y)	Cociente de la división entera	*quotient(12,5)*
(remainder x y)	Resto de la división entera (Signo del dividendo)	*remainder(12, 5)*
		remainder(12,-5)
		remainder(-12,5)

7. (*) Codifica funciones que permitan calcular el valor del término general de las siguientes sucesiones numéricas:
   a. $a_n = C\left(1 + \frac{i}{100}\right)^n$
      - Esta sucesión numérica permite calcular la cantidad que se obtiene al depositar una cantidad *C* durante *n* años con un interés del *i%*.
   b. $a_n = \frac{\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n}{\sqrt{5}}$
      - Comprueba que a_n es el n-ésimo término de la sucesión de Fibonacci.
   c. $b_n = \frac{a_{n+1}}{a_n}$
      - Donde a_n es el n-ésimo término de la sucesión de Fibonacci.
      - Comprueba que b_n converge hacia el número áureo: ϕ = 1.6180339887…

8. (*) Codifica funciones de conversión entre las siguientes unidades de medida:
   a. Millas a kilómetros.
      - 1 milla = 1,60934 kilómetros
   b. Kilómetros a millas.
   c. Grados Celsius a grados Fahrenheit.
      - Ejemplos: 0ºC → 32ºF, 100ºC → 212ºF
   d. Grados Fahrenheit en grados Celsius.

9. (*) Cálculo de la fecha del Domingo de Pascua o Domingo de Resurrección.
   - Codifica una función que utilice el algoritmo de Meuus-Jones-Butcher para calcular el día y mes del Domingo de Resurrección de un año y mostrarlos por pantalla.
   - Por ejemplo, el Domingo de Resurrección del año 2024 será el 31 de marzo.

- Referencias
  - Date of Easter.
    - Wikipedia.
    - https://en.wikipedia.org/wiki/Date_of_Easter#Meeus.2FJones.2FButcher_Gregorian_algorithm. Consultado el 12 de septiembre de 2023
  - Algorithm For Calculating The Date Of Easter Sunday
    - https://dzone.com/articles/algorithm-calculating-date
    - Consultado el 12 de septiembre de 2023
    - Pseudocódigo

```
int Y = year;
int a = Y % 19;
int b = Y / 100;
int c = Y % 100;
int d = b / 4;
int e = b % 4;
int f = (b + 8) / 25;
int g = (b - f + 1) / 3;
int h = (19 * a + b - d - g + 15) % 30;
int i = c / 4;
int k = c % 4;
int L = (32 + 2 * e + 2 * i - h - k) % 7;
int m = (a + 11 * h + 22 * L) / 451;
int month = (h + L - 7 * m + 114) / 31;
int day = ((h + L - 7 * m + 114) % 31) + 1;
```

- **Observaciones**
  - Al codificar en scheme el algoritmo, se debe tener en cuenta las operaciones deben ser con números enteros. Debido a ello, se debe usar *quotient* y *modulo* para calcular el cociente y el resto de la división entera.
  - El identificador "e" es una constante en el lenguaje Scheme: número "e" de Euler. Por tanto, se debe cambiar su nombre en le pseudocódigo anterior.

10. **(*)** Dado un polígono regular de "n" lados de longitud "l", codifica funciones que permitan calcular los siguientes valores:
    a. Perímetro = n * l
    b. Ángulo central: $\alpha = \frac{360^o}{n}$
    c. $Apotema = \frac{l}{2\ \tan\left(\frac{\alpha}{2}\right)}$
    d. $Área = \frac{perímetro * apotema}{2}$

11. **(*)** Codifica las siguientes funciones que calculan áreas de figuras geométricas del plano:
    a. areaTriangulo
    - Calcula el área del triángulo a partir de sus lados usando la fórmula de Herón.
      - $área = \sqrt{s\ (s-a)(s-b)(s-c}$
      - donde *s* es el semiperímetro: $s = \frac{a+b+c}{2}$

    b. areaRombo
    - Calcula el área del rombo a partir de sus diagonales.
      - $área = \frac{d1\ d2}{2}$
      - donde d_1 y d_2 son las diagonales del rombo.

    c. areaTrapecio
    - Calcula el área del trapecio a partir de sus bases y altura.

- $área = \frac{(b1+b2)}{2} \times h$
- donde b_1 y b_2 son las bases y h es la altura del trapecio.

12. **(*)** Codifica las siguientes funciones de distancias entre puntos del plano:
    a. **D2: distancia euclidiana o distancia L_2** entre dos puntos $P_1 = (x_1, y_1)$ y $P_2 = (x_2, y_2)$.
    $$D2(P_1, P_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$
    b. **D1: distancia de Manhattan, distancia de la ciudad de los bloques o distancia L_1** entre dos puntos $P_1 = (x_1, y_1)$ y $P_2 = (x_2, y_2)$.
    $$D1(P_1, P_2) = |x_2-x_1|+|y_2-y_1|$$
    c. **Dmax: distancia de ajedrez, distancia de Chebyshev o distancia L_∞** entre dos puntos $P_1 = (x_1, y_1)$ y $P_2 = (x_2, y_2)$.
    $$Dmax(P_1, P_2) = max(|x_2-x_1|, |y_2-y_1|)$$

13. **(*)** Codifica la siguiente función que calcula el área del triángulo plano a partir de sus vértices:
- **areaTrianguloVertices**
    - La función debe recibir como argumentos a las coordenadas de los vértices.
    - Utiliza las siguientes funciones auxiliares:
        - **areaTriangulo:** área del triángulo conocidos sus lados (ejercicio 11).
        - **D2***:* distancia euclidiana entre dos vértices (ejercicio 12).

14. **(*)** Utiliza la forma especial ***let*** para codificar una función que calcule el área de un rombo a partir de sus vértices.
- **areaRomboVerticesLet**
    - La función recibirá como argumentos las coordenadas de los vértices del rombo.
    - Utiliza los comentarios para indicar en qué "orden relativo" se han de introducir las coordenadas de los puntos del rombo para formar las diagonales.
    - Utiliza las siguientes funciones auxiliares:
        - **areaRombo:** área del rombo conocidas sus diagonales (ejercicio 11).
        - **D2***:* distancia euclidiana entre dos vértices (ejercicio 12).

15. **(*)** Codifica las siguientes funciones:
    a. Función denominada **distanciaPuntoRecta**
        - Ha de calcular la distancia de un punto $P = (x_0,y_0)$ a una recta $r \equiv a\,x + b\,y + c = 0$ mediante la siguiente fórmula
        $$d(P,r) = \frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}}$$
    b. Utiliza la forma especial ***let*** para codificar la función denominada **distanciaPuntoRecta2**
        - La función ha de calcular la distancia de un punto $P = (x_0,y_0)$ a la recta que pasa por otros dos puntos $P_1 = (x_1,y_1)$ y $P_2 = (x_2,y_2)$.
        - **Sugerencia**:
            - En primer lugar, determina los coeficientes de la recta $r \equiv a\,x + b\,y + c = 0$ que pasa por los puntos P_1 y P_2.
            - A continuación, utiliza la función del apartado "a".

16. **(*)** Utiliza la forma especial ***let*** para codificar una función que calcule el área de un trapecio a partir de sus vértices
    - **areaTrapecioLetVertices**
        - La función recibirá como argumentos las coordenadas de los vértices del trapecio.
        - Utiliza los comentarios para indicar en qué "orden relativo" se han de

introducir las coordenadas de los puntos del trapecio para formar las bases.

- Utiliza las siguientes funciones auxiliares:
  - **D2***:* distancia euclidiana entre dos puntos o vértices.
  - **distanciaPuntoRecta2:** distancia de un punto a una recta definida por dos puntos.
  - **areaTrapecio:** área del trapecio conocidas las bases y la altura.