CENTRO DE CIENCIAS BÁSICAS

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

LENGUAJES INTELIGENTES

5° "A"

ACTIVIDAD 18

Profesor: Alejandro Padilla Díaz

Alumno: Joel Alejandro Espinoza Sánchez

Fecha de Entrega: Aguascalientes, Ags.**, 24** de noviembre de 2020

Actividad 18

De los dos primeros ejercicios, se tienen en las actividades 16 y 17, observemos nuevamente:

Del primer archivo:

Tenemos cuatro ejercicios propuestos a resolver:

1) En el primer ejercicio se nos propone desarrollar un programa en LISP el cual convierta una cantidad de dólar a euros, por lo tanto, se desarrolla el siguiente código.

```
;Convertir una cantidad de dólar a euros
(define (DOLAR-EURO cd)
  (/ (* cd 20) 25))

(DOLAR-EURO 1)
(DOLAR-EURO 10)
(DOLAR-EURO 5)
(DOLAR-EURO 20)
(DOLAR-EURO 22)
(DOLAR-EURO 35)
```

Éste, junto con algunas ejecuciones de ejemplo produce el siguiente resultado al ejecutarse:

```
Bienvenido a DrScheme, versión 299.100.
Lenguaje: Estudiante Principiante.
0.8
8
4
16
17.6
28
> (DOLAR-EURO 8)
6.4
>
```

Observemos que también se realiza una ejecución ejemplo cuando $cd = 8$ para observar un funcionamiento correcto del programa.

2) En el segundo ejercicio, trabajamos sencillamente la definición recursiva de cómo calcular el factorial de un número, por lo tanto, se desarrolló el siguiente código junto con algunas ejecuciones de ejemplo:

```
(define fact
  (lambda (n)
    (if (zero? n) 1
        (* n (fact (sub1 n))))))

(fact 20)
(fact 5)
(fact 0)
(fact 70)
```

Este código produce a su vez el siguiente resultado:

```
Bienvenido a DrScheme, versión 299.100.
Lenguaje: Estudiante Principiante.
2432902008176640000
120
1
11978571669969891796072783721689098736458938142546425857555362864628009582789845319680000000000000000
> (fact 18)
6402373705728000
>
```

También probamos con otro ejemplo para revisar que trabaja correctamente.

3) En el tercer ejemplo, tenemos un ejercicio principal y tres incisos a trabajar:
   - En el primer ejercicio de este ejemplo, tratamos de ingresar tres números y el programa los “volteará” como si fuera un conversor de tres dígitos que cambia los números a la inversa, entonces desarrollamos el código:

```
;Definición de la función
(define (convert3 n1 n2 n3)
  (+ (* n3 100) (* n2 10) n1))

(convert3 1 1 1)
(convert3 5 3 4)
```

Observamos que los resultados obtenidos son de la ejecución completa del programa, pero señalamos con rojo aquellos resultados pertenecientes a este ejercicio:

```
Bienvenido a DrScheme, versión 299.100.
Lenguaje: Estudiante Principiante.
111
435
14
35
91
22
32.5
60.5
1.5
1.8
1.8
>
```

a. Para los ejemplos posteriores, se pedía determinar el valor en $n = 2$, $n = 5$ y $n = 9$ y expresarlo en forma de programa. Por lo tanto, para este inciso se pedía evaluar:

$$n^2 + 10$$

El código desarrollado es el siguiente:

```
;a) Evaluar n^2 + 10
;Definimos la función
(define (g n)
  (+ (* n n) 10))

(g 2)
(g 5)
(g 9)
```

Los resultados obtenidos nuevamente remarcados son los siguientes:

```
Bienvenido a DrScheme, versión 299.100.
Lenguaje: Estudiante Principiante.
111
435
14
35
91
22
32.5
60.5
1.5
1.8
1.8
>
```

b. Para este inciso se pedía evaluar:

$$\frac{n^2}{2} + 20$$

El código desarrollado es el siguiente:

```
;b) Evaluar (1/2) * n^2 + 20
;Definimos la función
(define (h n)
  (+ (* (/ 1 2) (* n n)) 20))

(h 2)
(h 5)
(h 9)
```

Los resultados generados son los siguientes:

```
Bienvenido a DrScheme, versión 299.100.
Lenguaje: Estudiante Principiante.
111
435
14
35
91
22
32.5
60.5
1.5
1.8
1.8
>
```

c. Para este inciso se pedía evaluar:

$$2 - \frac{1}{n}$$

El código desarrollado es el siguiente:

```
;c) Evaluar 2 - (1/n)
;Definimos la función
(define (b n)
  (- 2 (/ 1 n)))

(b 2)
(b 5)
(b 9)
```

Los resultados son los siguientes:

```
Bienvenido a DrScheme, versión 299.100.
Lenguaje: Estudiante Principiante.
111
435
14
35
91
22
32.5
60.5
1.5
1.8
1.8
>
```

4) En el último ejemplo, se pedía hacer un ejemplo donde se sumaran monedas de cierto peso, en este caso de $1, $2, $5 y $10 en una bolsa. El código es el siguiente:

```
;Definir el programa suma-monedas. Consume cuatro números:
;el número de monedas de $1, $2, $5, $10 en la bolsa y
;produce la cantidad de dinero en la bolsa

;Definición de la función
(define (suma-monedas m1 m2 m3 m4)
  (+ (* m1 1) (* m2 2) (* m3 5) (* m4 10)))

(suma-monedas 100 40 30 20)
(suma-monedas 23 40 200 10)
```

El resultado de la ejecución del código anterior junto con una prueba extra, es el siguiente:

```
Bienvenido a DrScheme, versión 299.100.
Lenguaje: Estudiante Principiante.
530
1203
> (suma-monedas 1 1 1 1)
18
>
```

Del segundo archivo:

Tenemos cuatro ejercicios propuestos a resolver:

1) En el primer ejercicio se nos propone desarrollar un programa en LISP para calcular el área de un cilindro dado el radio de su base y su altura:

```
;Desarrollar el programa para hallar el área de un cilindro
;Entradas: Radio de la base y altura
;Salida: El área del cilíndro

;Definimos las funciones
(define (areacilindro r h)
  (+ (* 2 PI r h) (* 2 PI (* r r))))

(define PI 3.14)

;Ejemplos
(areacilindro 3 2)
(areacilindro 4 6)
(areacilindro 6 3)
```

Los resultados fueron los siguientes:

```
Bienvenido a DrScheme, versión 299.100.
Lenguaje: Estudiante Principiante.
94.2
251.2
339.12
>
```

2) Como segundo ejercicio nos pedían hallar el volumen de un cilindro bajo las mismas entradas que el ejercicio anterior:

```
;Desarrollar el programa para hallar el volumen de un cilindro
;Entradas: Radio de la base y altura
;Salida: El volumen del cilindro

;Definimos las funciones
(define (volcilindro r h)
  (* PI r r h))

(define PI 3.14)

;Ejemplos
(volcilindro 3 2)
(volcilindro 4 6)
(volcilindro 6 3)
```

Los resultados son los siguientes:

```
Bienvenido a DrScheme, versión 299.100.
Lenguaje: Estudiante Principiante.
56.52
301.44
339.12
>
```

3) En este ejercicio se nos pidió evaluar tres condiciones, las cuales son:

```
;Condiciones en DrScheme

;1) 4 > 3 AND 10 <= 100
(and (> 4 3) (<= 10 100))

;2) 4 > 3 OR 10 = 100
(or (> 4 3) (= 10 100))

;3) NOT 2 > 3
(not (> 2 3))
```

Los resultados correspondientes son:

```
Bienvenido a DrScheme, versión 299.100.
Lenguaje: Estudiante Principiante.
true
true
true
>
```

De igual forma, otros ejercicios de comparación fueron los siguientes:

```
;Condiciones en DrScheme

;1) 4 > 3
(> 4 3)

;2) 4 > 3 AND 2 > 3
(and (> 4 3) (> 2 3))

;3) (7/2) * 3.5 = 3.5 * (14/4)
(= (* (/ 7 2) 3.5) (* 3.5 (/ 14 4)))
```

Con resultados:

```
Bienvenido a DrScheme, versión 299.100.
Lenguaje: Estudiante Principiante.
true
false
true
>
```

4) Otro ejercicio más era programar un verificador de intervalos en el que se pudiera determinar si un número se encontraba dentro de este rango. Por lo que las funciones son las siguientes:

```
;Teniendo en cuenta algunos intervalos,
;observamos si un número puede encontrarse dentro de estos

;1) El intervalo (3,7]
(define (int3a7c n)
  (and (> n 3) (<= n 7)))

;2) El intervalo [3,7]
(define (int3c7c n)
  (and (>= n 3) (<= n 7)))

;3) El intervalo [3,9)
(define (int3c9a n)
  (and (>= n 3) (< n 9)))

;4) El intervalo (1,3) U (9,11)
(define (int1a3a-9a11a n)
  (or (and (> n 1) (< n 3)) (and (> n 9) (< n 11))))
```

```
;5) El rango de números fuera del intervalo [1,3]
(define (notint1c3c n)
  (not (and (>= 1 n) (<= 3 n))))
```

Las pruebas fueron las siguientes:

```
Bienvenido a DrScheme, versión 299.100.
Lenguaje: Estudiante Principiante.
> (int3a7c 5)
true
> (int3a7c 7)
true
> (int3c7c 3)
true
> (int3c9a 9)
false
> (int1a3a-9a11a 2)
true
> (int1a3a-9a11a 6)
false
> (notint1c3c 2)
false
> (notint1c3c 0)
true
> (notint1c3c 4)
true
>
```

Del tercer archivo:

Se tendrá el siguiente código:

```
;Traslade las siguientes tres funciones de LISP a Scheme de intervalos
;sobre la recta real, tambi?n formule el contrato y el prop?sito
;para cada una:

;1.
(define (in-interval-1? x)
       (and (< -3 x) (< x 0)))

;Ejemplo 1, Resultado esperado: VERDADERO
(in-interval-1? -2.9)

;Ejemplo 2, Resultado esperado: VERDADERO
(in-interval-1? -0.1)
```

```
;Ejemplo 3, Resultado esperado: FALSO
(in-interval-1? 3)

;Ejemplo 4, Resultado esperado: FALSO
(in-interval-1? 0)

;El intervalo en la recta real es: (-3,0).

;CONTRATO
;in-interval-1? :  n?mero -> boolean

;PROP?SITO
;Determinar el intervalo soluci?n para la funci?n expresada.

;2.
(define (in-interval-2? x)
       (or (< x 1) (> x 2)))

;Ejemplo 1, Resultado esperado: FALSO
(in-interval-2? 1)

;Ejemplo 2, Resultado esperado: FALSO
(in-interval-2? 2)

;Ejemplo 3, Resultado esperado: VERDADERO
(in-interval-2? 0.9)

;Ejemplo 4, Resultado esperado: VERDADERO
(in-interval-2? 2.1)

;La soluci?n en la recta real es el rango externo del intervalo
;[1,2].

;CONTRATO
;in-interval-2? :  n?mero -> boolean

;PROP?SITO
;Determinar el intervalo soluci?n para la funci?n expresada.

;3.
(define (in-interval-3? x)
```

```
       (not (and (<= 1 x) (<= x 5))))

;Ejemplo 1, Resultado esperado: FALSO
(in-interval-3? 1)

;Ejemplo 2, Resultado esperado: FALSO
(in-interval-3? 5)

;Ejemplo 3, Resultado esperado: VERDADERO
(in-interval-3? 0.9)

;Ejemplo 4, Resultado esperado: VERDADERO
(in-interval-3? 5.1)

;La soluci?n en la recta real es el rango externo del intervalo
;[1,5].

;CONTRATO
;in-interval-3? :  n?mero -> boolean

;PROP?SITO
;Determinar el intervalo soluci?n para la funci?n expresada.
```

Sin embargo no se encuentra un error en el código, ya que la expresión de intervalos es correcta y los resultados esperados también.