UNIVERSIDAD DE CÓRDOBA

ESCUELA POLITÉCNICA SUPERIOR DE CÓRDOBA

DEPARTAMENTO DE
INFORMÁTICA Y ANÁLISIS NUMÉRICO

PROGRAMACIÓN DECLARATIVA

INGENIERÍA INFORMÁTICA

CUARTO CURSO

PRIMER CUATRIMESTRE

Tema 3.- Predicados y sentencias condicionales

Primera parte: Scheme

- Tema 1.- Introducción al Lenguaje Scheme
- Tema 2.- Expresiones y Funciones
- Tema 3.- Predicados y sentencias condicionales
- Tema 4.- Iteración y Recursión
- Tema 5.- Tipos de Datos Compuestos
- Tema 6.- Abstracción de Datos
- Tema 7.- Lectura y Escritura

Segunda parte: Prolog

- Tema 8.- Introducción al Lenguaje Prolog
- Tema 9.- Elementos Básicos de Prolog
- Tema 10.- Listas
- Tema 11.- Reevaluación y el “corte”
- Tema 12.- Entrada y Salida

Primera parte: Scheme

Índice

Índice

1. Predicados simbólicos

- **Descripción**
  - También se denominan “predicados de tipo”
  - Permite indicar la **naturaleza** de un objeto.
  - Un objeto solamente puede hacer verdadero a **uno** de los predicados.

1. Predicados simbólicos

- *boolean?*
- *number?*
- *char?*
- *string?*
- *procedure?*
- *symbol?*
- *pair?*
- *list?*
- *vector?*

1. Predicados simbólicos
   - *boolean?*
     - **Sintaxis**

       (boolean? objeto)
     - **Significado**
       - Devuelve verdadero *#t* si el valor de *objeto* es un valor lógico (verdadero *#t* o falso *#f*)
       - Devuelve falso *#f* en caso contrario
     - **Ejemplo**

       (boolean? 10) ➔ *#f*

       (boolean? (> 0 9)) ➔ *#t*

1. Predicados simbólicos
   - *number?*
     - **Sintaxis**

       (number? objeto)
     - **Significado**
       - Devuelve verdadero *#t* si el valor de *objeto* es numérico
       - Devuelve falso *#f* en caso contrario
     - **Ejemplo**

       (number? 10) ➔ *#t*

       (number? (> 0 9)) ➔ *#f*

1. Predicados simbólicos

- *char?*
  - **Sintaxis**

    (char? objeto)
  - **Significado**
    - Devuelve verdadero *#t* si el valor de *objeto* es un carácter
    - Devuelve falso *#f* en caso contrario
  - **Ejemplo**

    (char? 10) ➔ *#f*

    (char? #\a) ➔ *#t*

1. Predicados simbólicos

- *string?*
  - **Sintaxis**

    (string? objeto)
  - **Significado**
    - Devuelve verdadero *#t* si el valor de *objeto* es una cadena de caracteres
    - Devuelve falso *#f* en caso contrario
  - **Ejemplo**

    (string? 10) ➔ *#f*

    (string? "cadena") ➔ *#t*

1. Predicados simbólicos
   - *procedure?*
     - **Sintaxis**

       (procedure? objeto)
     - **Significado**
       - Devuelve verdadero *#t* si *objeto* es un procedimiento, función u operador
       - Devuelve falso *#f* en caso contrario
     - **Ejemplo**

       *(procedure? *)* ➔ *#t*

       (procedure? sqrt) ➔ *#t*

1. Predicados simbólicos

- *symbol?*
  - **Sintaxis**

    (symbol? objeto)
  - **Significado**
    - Devuelve verdadero *#t* si el valor de *objeto* es un símbolo (literal definido por el programador).
    - Devuelve falso *#f* en caso contrario
  - **Ejemplo**

    (symbol? 9) ➔ *#f*

    (symbol? 'a) ➔ *#t*

1. Predicados simbólicos

- *pair?*
  - **Sintaxis**

    (pair? objeto)
  - **Significado**
    - Devuelve verdadero *#t* si el valor de *objeto* es un par.
    - Devuelve falso *#f* en caso contrario
  - **Ejemplo**

    ***(cons** 1 2)* ➔ *(1 . 2)*

    *(pair? **(cons** 1 2))* ➔ *#t*

    (pair? 9) ➔ *#f*

1. Predicados simbólicos

- *list?*
  - **Sintaxis**

    (list? objeto)
  - **Significado**
    - Devuelve verdadero *#t* si el valor de *objeto* es una lista.
    - Devuelve falso *#f* en caso contrario
  - **Ejemplo**

    ***(list** 1 2 3 4)* ➔ *(1 2 3 4)*

    *(list? **(list** 1 2 3 4))* ➔ *#t*

    (list? 9) ➔ *#f*

1. Predicados simbólicos
   - *vector?*
     - **Sintaxis**

       (vector? objeto)
     - **Significado**
       - Devuelve verdadero *#t* si el valor de *objeto* es un vector
       - Devuelve falso *#f* en caso contrario
     - **Ejemplo**

       ***(vector** 1 2 3 4)* ➔ *#(1 2 3 4)*

       (vector? #(1 2 3 4)) ➔ *#t*

       (vector? 9) ➔ *#f*

Índice

2. Predicados y operadores relacionales numéricos

- Operadores relacionales
- Predicados de tipo numérico
- Predicados de signo
- Predicados de paridad
- Predicados de exactitud

2. Predicados y operadores relacionales numéricos

- Operadores relacionales
- Predicados de tipo numérico
- Predicados de signo
- Predicados de paridad
- Predicados de exactitud

2. Predicados y operadores relacionales numéricos

- Operadores relacionales

Operador	Significado	Ejemplo
<	*Menor que*	(< exp_1 exp_2 … exp_n)
<=	*Menor o igual que*	(<= exp_1 exp_2 … exp_n)
>	*Mayor que*	(> exp_1 exp_2 … exp_n)
>=	*Mayor o igual que*	(>= exp_1 exp_2 … exp_n)
=	*Igual que*	(= exp_1 exp_2 … exp_n)

2. Predicados y operadores relacionales numéricos

- Operadores relacionales
  - **Observación**
    - Los argumentos deben ser **expresiones numéricas**
    - Más adelante se explicarán otros operadores de igualdad o **equivalencia**
      - ***eq?***
      - ***eqv?***
      - ***equal?***

2. Predicados y operadores relacionales numéricos

- Operadores relacionales
  - **Ejemplos**

(define a 1)

(define b 2)

(define c 3)

(= a b) ➔ *#f*

(< 0 a 10) ➔ *#t*

(<= b 10) ➔ *#t*

(> c b a 0) ➔ *#t*

(>= c 0) ➔ *#t*

2. Predicados y operadores relacionales numéricos

- Operadores relacionales
- Predicados de tipo numérico
- Predicados de signo
- Predicados de paridad
- Predicados de exactitud

2. Predicados y operadores relacionales numéricos

- Predicados de tipo numérico
  - complex?
  - real?
  - rational?
  - integer?

2. Predicados y operadores relacionales numéricos

- Predicados de tipo numérico
  - *complex?*
    - **Sintaxis**

      (complex? objeto)

    - **Significado**
      - ❑ Devuelve verdadero *#t* si el valor de *objeto* es un número (de cualquier tipo).
      - ❑ Devuelve falso *#f* en caso contrario
      - ❑ Este predicado es equivalente a *number?*

2. Predicados y operadores relacionales numéricos

- Predicados de tipo numérico
  - *complex?*
    - **Ejemplos**

(complex? 9) ➔ *#t*

(complex? 9/2) ➔ *#t*

(complex? 9.0) ➔ *#t*

(complex? 9+2i) ➔ *#t*

2. Predicados y operadores relacionales numéricos

- Predicados de tipo numérico
  - *real?*
    - **Sintaxis**

      (real? objeto)

    - **Significado**

      ❑ Devuelve verdadero *#t* si el valor de *objeto* es un número real (incluidos racionales y enteros)

      ❑ Devuelve falso *#f* en caso contrario

2. Predicados y operadores relacionales numéricos

- Predicados de tipo numérico
  - *real?*
    - **Ejemplos**

(real? 9) ➔ *#t*

(real? 9/2) ➔ *#t*

(real? 9.0) ➔ *#t*

(real? 9+2i) ➔ *#f*

2. Predicados y operadores relacionales numéricos

- Predicados de tipo numérico
  - *rational?*
    - **Sintaxis**

      (rational? objeto)

    - **Significado**
      - ❑ Devuelve verdadero *#t* si el valor de *objeto* es un número racional (incluidos los enteros)
      - ❑ Devuelve falso *#f* en caso contrario
      - ❑ En muchas implementaciones, ***real?*** y ***rational?*** son iguales

2. Predicados y operadores relacionales numéricos

- Predicados de tipo numérico
  - *rational?*
    - **Ejemplos**

(rational? 9) ➔ *#t*

(rational? 9/2) ➔ *#t*

(rational? 9.0) ➔ *#t*

(rational? 9+2i) ➔ *#f*

2. Predicados y operadores relacionales numéricos

- Predicados de tipo numérico
  - *integer?*
    - **Sintaxis**

      (integer? objeto)

    - **Significado**
      - ❑ Devuelve verdadero *#t* si el valor de *objeto* es un número entero
      - ❑ Devuelve falso *#f* en caso contrario

2. Predicados y operadores relacionales numéricos

- Predicados de tipo numérico
  - *integer?*
    - **Ejemplos**

(integer? 9) ➔ *#t*

(integer? 9/2) ➔ *#f*

(integer? 9.0) ➔ *#t*

(integer? 9.5) ➔ *#f*

(integer? 9+2i) ➔ *#f*

2. Predicados y operadores relacionales numéricos

- Operadores relacionales
- Predicados de tipo numérico
- Predicados de signo
- Predicados de paridad
- Predicados de exactitud

2. Predicados y operadores relacionales numéricos

- Predicados de signo
  - positive?
  - negative?
  - zero?

2. Predicados y operadores relacionales numéricos

- Predicados de signo
  - *positive?*
    - **Sintaxis**

      (positive? objeto)

    - **Significado**

      ❑ Devuelve verdadero *#t* si el valor de *objeto* es un número positivo

      ❑ Devuelve falso *#f* en caso contrario

2. Predicados y operadores relacionales numéricos

- Predicados de signo
  - *positive?*
    - **Ejemplos**

      (positive? 9) ➔ *#t*

      (positive? -9) ➔ *#f*

      (positive? 0) ➔ *#f*

2. Predicados y operadores relacionales numéricos

- Predicados de signo
  - *negative?*
    - **Sintaxis**

      (negative? objeto)

    - **Significado**

      ❑ Devuelve verdadero *#t* si el valor de *objeto* es un número negativo

      ❑ Devuelve falso *#f* en caso contrario

2. Predicados y operadores relacionales numéricos

- Predicados de signo
  - *negative?*
    - **Ejemplos**

(negative? 9) ➔ *#f*

(negative? -9) ➔ *#t*

(negative? 0) ➔ *#f*

2. Predicados y operadores relacionales numéricos

- Predicados de signo
  - *zero?*
    - **Sintaxis**

      (zero? objeto)

    - **Significado**
      - ❑ Devuelve verdadero *#t* si el valor de *objeto* es el número cero
      - ❑ Devuelve falso *#f* en caso contrario

2. Predicados y operadores relacionales numéricos

- Predicados de signo
  - *zero?*
    - **Ejemplos**

(zero? 9) ➔ *#f*

(zero? -9) ➔ *#f*

(zero? 0) ➔ *#t*

2. Predicados y operadores relacionales numéricos

- Operadores relacionales
- Predicados de tipo numérico
- Predicados de signo
- Predicados de paridad
- Predicados de exactitud

2. Predicados y operadores relacionales numéricos

- Predicados de paridad
  - even?
  - odd?

2. Predicados y operadores relacionales numéricos

- Predicados de paridad
  - *even?*
    - **Sintaxis**

      (even? objeto)
    - Significado
      - ❑ Devuelve verdadero *#t* si el valor de *objeto* es un número par.
      - ❑ Devuelve falso *#f* en caso contrario
    - **Ejemplo**

      (even? 2 ➔ #t

      (even? 9) ➔ #f

2. Predicados y operadores relacionales numéricos

- Predicados de paridad
  - *odd?*
    - **Sintaxis**

      (odd? objeto)

    - **Significado**
      - ❑ Devuelve verdadero *#t* si el valor de *objeto* es un número impar.
      - ❑ Devuelve falso *#f* en caso contrario
    - **Ejemplo**

      (odd? 2) ➔ *#f*

      (odd? 9) ➔ *#t*

2. Predicados y operadores relacionales numéricos

- Operadores relacionales
- Predicados de tipo numérico
- Predicados de signo
- Predicados de paridad
- Predicados de exactitud

2. Predicados y operadores relacionales numéricos

- Predicados de exactitud
  - exact?
  - inexact?

2. Predicados y operadores relacionales numéricos

- Predicados de exactitud
  - *exact?*
    - **Sintaxis**

      (exact? objeto)

    - **Significado**

      ❑ Devuelve verdadero *#t* si el valor de *objeto* es un número exacto:

      ✓ escrito como una constante exacta

      ✓ u obtenido al aplicar operaciones exactas

      ❑ Devuelve falso *#f* en caso contrario

2. Predicados y operadores relacionales numéricos

- Predicados de exactitud
  - *exact?*
    - **Ejemplos**

(exact? 2) ➔ *#t*

(max 9 2.0) ➔ *9.0*

(exact? (max 9 2.0)) ➔ *#f*

2. Predicados y operadores relacionales numéricos

- Predicados de exactitud
  - *inexact?*
    - **Sintaxis**

      (inexact? objeto)

    - **Significado**
      - ❑ Devuelve verdadero *#t* si el valor de *objeto* es un número inexacto
      - ❑ Devuelve falso *#f* en caso contrario

2. Predicados y operadores relacionales numéricos

- Predicados de exactitud
  - *inexact?*
    - **Ejemplos**

(inexact? 2 ➔ #f

(max 9 2.0) ➔ 9.0

(inexact? (max 9 2.0)) ➔ #t

Índice

3. Predicados alfanuméricos

- Predicados de caracteres
- Predicados de cadenas

3. Predicados alfanuméricos
   - Predicados de caracteres
   - Predicados de cadenas

3. Predicados alfanuméricos

- Predicados de caracteres
  - Predicados relacionales de caracteres
  - Predicados de tipo de carácter

3. Predicados alfanuméricos
   - Predicados de caracteres
     - Predicados relacionales de caracteres
     - Predicados de tipo de carácter

3. Predicados alfanuméricos

- Predicados de caracteres
  - Predicados relacionales de caracteres
    - Distinguen mayúsculas y minúsculas
      - char<?, char<=?,
      - char>?, char>=?
      - char=?
    - No distinguen mayúsculas y minúsculas
      - char-ci<?, char-ci<=?
      - char-ci>?, char-ci>=?
      - char-ci=?

ci: case insensitive

3. Predicados alfanuméricos

- Predicados de caracteres
  - Predicados relacionales de caracteres
    - Distinguen mayúsculas y minúsculas

Operador	Significado	Ejemplo
char<?	*Menor que*	*(char<? $char_1$ $char_2$ … $char_n$)*
char<=?	*Menor o igual que*	*char<=? $char_1$ $char_2$ … $char_n$)*
char>?	*Mayor que*	*(char>? $char_1$ $char_2$ … $char_n$)*
char>=?	*Mayor o igual que*	*(char>=? $char_1$ $char_2$ … $char_n$)*
char=?	*Igual que*	*(char=? $char_1$ $char_2$ … $char_n$)*

3. Predicados alfanuméricos

- Predicados de caracteres
  - Predicados relacionales de caracteres
    - No distinguen mayúsculas y minúsculas

ci: case insensitive

Operador	Significado	Ejemplo
char-ci<?	*Menor que*	*(char-ci<? $char_1$ $char_2$ … $char_n$)*
char-ci<=?	*Menor o igual que*	*(char-ci<=? $char_1$ $char_2$ … $char_n$)*
char-ci>?	*Mayor que*	*(char-ci>? $char_1$ $char_2$ … $char_n$)*
char-ci>=?	*Mayor o igual que*	*(char-ci>=? $char_1$ $char_2$ … $char_n$)*
char-ci=?	*Igual que*	*(char-ci=? $char_1$ $char_2$ … $char_n$)*

3. Predicados alfanuméricos

- Predicados de caracteres
  - Predicados relacionales de caracteres
    - **Ejemplos**

(char=? #\a #\A) ➔ #f

(char-ci=? #\a #\A) ➔ #t

3. Predicados alfanuméricos
   - Predicados de caracteres
     - Predicados relacionales de caracteres
     - Predicados de tipo de carácter

3. Predicados alfanuméricos

- Predicados de caracteres
  - Predicados de tipo de carácter
    - char-whitespace?
    - char-numeric?
    - char-alphabetic?
    - char-upper-case?
    - char-lower-case?

3. Predicados alfanuméricos

- Predicados de caracteres
  - Predicados de tipo de carácter
    - *char-whitespace?*
    - **Sintaxis**

      (*char-whitespace? objeto*)
    - **Significado**
      - Comprueba si el objeto es un espacio un espacio, tabulador o salto de línea.
    - **Ejemplos**

      (char-whitespace? #\ ) ➔ *#t*

3. Predicados alfanuméricos

- Predicados de caracteres
  - Predicados de tipo de carácter
    - *char-whitespace?*
    - Espacios en blanco en *Scheme*
      - Espacio: **#** o **#\space**
      - Tabulador: **#\tab**
      - Salto de línea: **#\newline**

3. Predicados alfanuméricos

- Predicados de caracteres
  - Predicados de tipo de carácter
    - *char-numeric?*
    - **Sintaxis**

      (char-numeric? objeto)
    - **Significado**
      - ❑ Comprueba si es un carácter numérico
    - Ejemplos

      (char-numeric? #\8) ➔ *#t*

      (char-numeric? 8) ➔ *#f*

3. Predicados alfanuméricos

- Predicados de caracteres
  - Predicados de tipo de carácter
    - *char-alphabetic?*
    - **Sintaxis**

      (char-alphabetic? objeto)
    - **Significado**
      - ❑ Comprueba si el objeto es una letra
    - Ejemplos

      (char-alphabetic? #\7) ➔ *#f*

      (char-alphabetic? #\B) ➔ *#t*

3. Predicados alfanuméricos

- Predicados de caracteres
  - Predicados de tipo de carácter
    - *char-upper-case?*
    - **Sintaxis**

      (char-upper-case? objeto)
    - **Significado**
      - ❑ Comprueba si el objeto es una letra mayúscula
    - Ejemplos

      (char-upper-case? #\a) ➔ *#f*

      (char-upper-case? #\B) ➔ *#t*

3. Predicados alfanuméricos

- Predicados de caracteres
  - Predicados de tipo de carácter
    - *char-lower-case?*
    - **Sintaxis**

      (char-lower-case? objeto)
    - **Significado**
      - ❑ Comprueba si el objeto es una letra minúscula
    - Ejemplos

      (char-lower-case? #\A) ➔ *#f*

      (char-lower-case? #\b) ➔ *#t*

3. Predicados alfanuméricos

- Predicados de caracteres
- Predicados de cadenas

3. Predicados alfanuméricos

- Predicados de cadenas
  - Predicados relacionales de cadenas
    - Distinguen mayúsculas de minúsculas
      - string<?, string<=?
      - string>?, string>=?
      - string =?
    - No distinguen mayúsculas de minúsculas
      - string-ci<?, string-ci<=?
      - string-ci>?, string-ci>=?
      - string-ci=?

ci: case insensitive

3. Predicados alfanuméricos
   - Predicados de cadenas
     - Predicados relacionales de cadenas
       - Distinguen mayúsculas de minúsculas

Operador	Significado	Ejemplo
string<?	*Menor que*	*(string<? cad_1 cad_2 … cad_n)*
string<=?	*Menor o igual que*	*string<=? cad_1 cad_2 … cad_n)*
string>?	*Mayor que*	*(string>? cad_1 cad_2 … cad_n)*
string>=?	*Mayor o igual que*	*(string>=? cad_1 cad_2 … cad_n)*
string=?	*Igual que*	*(string=? cad_1 cad_2 … cad_n)*

3. Predicados alfanuméricos

- Predicados de cadenas
  - Predicados relacionales de cadenas
    - No distinguen mayúsculas de minúsculas

Operador	Significado	Ejemplo
string-ci<?	*Menor que*	*(string-ci<? cad_1 cad_2 … cad_n)*
string-ci<=?	*Menor o igual que*	*(string-ci<=? cad_1 cad_2 … cad_n)*
string-ci>?	*Mayor que*	*(string-ci>? cad_1 cad_2 … cad_n)*
string-ci>=?	*Mayor o igual que*	*(string-ci>=? cad_1 cad_2 … cad_n)*
string-ci=?	*Igual que*	*(string-ci=? cad_1 cad_2 … cad_n)*

ci: case insensitive

3. Predicados alfanuméricos

- Predicados de cadenas
  - Predicados relacionales de cadenas
    - **Ejemplos**

(string=? "Hola" "hola") ➔ *#f*

(string-ci=? "Hola" "hola") ➔ *#t*

(define nombre1 "Alicia")

(define nombre2 "Ana")

(define nombre3 "Pedro")

(string-ci<? nombre1 nombre2 nombre3) ➔ *#t*

Índice

4. Predicados de igualdad o equivalencia

- ***eq?***
- ***eqv?***
- ***equal?***
- **=**
  - **Observación**
    - El operador “=” es exclusivo de datos numéricos

4. Predicados de igualdad o equivalencia

- ***eq?***
  - **Sintaxis**

    *(**eq?** objeto$_1$ objeto$_2$)*

  - **Significado**
    - Devuelve verdadero si los dos argumentos son el **mismo objeto** (ocupan la misma posición en memoria)
    - Su implementación depende del intérprete

4. Predicados de igualdad o equivalencia

- ***eq?***
  - **Ejemplos**

(eq? 'yes 'yes) ➔ *#t*

(eq? “yes” “yes”) ➔ *#f*

(eq? 3 3.) ➔ *#f*

(eq? 3 3.0) ➔ *#f*

(eq? 3. 3.0) ➔ *#f*

(eq? 3.0 3.0) ➔ *#f*

4. Predicados de igualdad o equivalencia

- ***eqv?***
  - **Sintaxis**

    *(**eqv?** $objeto_1$ $objeto_2$)*
  - **Significado**
    - Devuelve
      - ❑ ***#t*** si y solamente si el ***operador eq?*** devuelve ***#t***, salvo para tipos de datos específicos
      - ❑ en caso contario, ***#f***
    - ***eq?*** y **eqv?** tienen un comportamiento **diferente** con datos numéricos o de tipo carácter

4. Predicados de igualdad o equivalencia

- ***eqv?***
  - **Ejemplos**

(eqv? 'yes 'yes) ➔ *#t*

(eqv? "yes" "yes") ➔ *#f*

(eqv? 3 3.) ➔ *#f*

(eqv? 3 3.0) ➔ *#f*

(eqv? 3. 3.0) ➔ *#t*

(eqv? 3.0 3.0) ➔ *#t*

4. Predicados de igualdad o equivalencia

- ***equal?***
  - **Sintaxis**

    *(**equal?** $objeto_1$ $objeto_2$)*
  - **Significado**
    - Devuelve ***#t*** si y solamente si el operador ***eqv?*** devuelve ***#t***, salvo para tipos de datos específicos:
      - ❑ cadenas
      - ❑ pares, listas
      - ❑ vectores
      - ❑ etc.

4. Predicados de igualdad o equivalencia

- ***equal?***
  - **Ejemplos**

(equal? 'yes 'yes) ➔ *#t*

(equal? "yes" "yes") ➔ *#t*

(equal? 3 3.) ➔ *#f*

(equal? 3 3.0) ➔ *#f*

(equal? 3. 3.0) ➔ *#t*

(equal? 3.0 3.0) ➔ *#t*

4. Predicados de igualdad o equivalencia

- **=**
  - **Sintaxis**

    (= exp_1 exp_2 … exp_n)
  - **Significado**
    - Los argumentos deben ser expresiones aritméticas
    - Comprueba la igualdad numérica de los argumentos

4. Predicados de igualdad o equivalencia

- **=**
  - **Ejemplos**

(= 3 3.) ➔ *#t*

(= 3 3.0) ➔ *#t*

(= 3. 3.0) ➔ *#t*

(= 3.0 3.0) ➔ *#t*

Índice

5. Operadores lógicos

- Disyunción lógica: *or*
- Conjunción lógica: *and*
- Negación lógica: *not*

5. Operadores lógicos

- Disyunción lógica: *or*
- Conjunción lógica: *and*
- Negación lógica: *not*

5. Operadores lógicos

- Disyunción lógica: or
  - **Sintaxis**

    *(**or** exp_1 exp_2 … exp_n)*

    - donde

      exp_1 exp_2 … exp_n

      pueden ser expresiones de cualquier tipo:

      - ❑ numéricas
      - ❑ alfanuméricas
      - ❑ …
      - ❑ pero generalmente son expresiones lógicas

5. Operadores lógicos

- Disyunción lógica: or
  - **Significado**
    - Se evalúan las expresiones lógicas de izquierda a derecha
    - **Evaluación en "corto circuito"**
      - ❑ Devuelve el valor de la primera expresión que no sea falsa (*#f*) y no evalúa el resto de expresiones.
      - ❑ Si todas las expresiones son falsas (*#f*) entonces devuelve el valor falso (*#f)*
      - ❑ Si no hay expresiones, el valor devuelto es falso *#f*

5. Operadores lógicos

- Disyunción lógica: or
  - **Ejemplos**

*(**define** a 1)*

*(**or** (> a 0) (< a 10))* ➔ *#t*

*(**or** (+ 2 a) "hola")* ➔ *3*

*(**or** "hola" (+ 2 a))* ➔ *"hola"*

5. Operadores lógicos

- Disyunción lógica: *or*
- Conjunción lógica: *and*
- Negación lógica: *not*

5. Operadores lógicos

- Conjunción lógica: and
  - **Sintaxis**

    (***and*** exp_1 exp_2 … exp_n)

    - donde

      exp_1 exp_2 … exp_n

      pueden ser expresiones de cualquier tipo:

      - ❑ numéricas
      - ❑ alfanuméricas
      - ❑ …
      - ❑ pero generalmente son expresiones lógicas.

5. Operadores lógicos

- Conjunción lógica: and
  - **Significado**
    - Se evalúan las expresiones lógicas de izquierda a derecha
    - **Evaluación en "corto circuito"**
      - Devuelve el valor el valor falso (*#f*) tan pronto como una expresión se evalúe a falso (*#f*), no evaluando el resto de expresiones.
      - Si todas las expresiones tienen un valor distinto de falso (*#f*) entonces **devuelve** el valor de la última expresión
      - Si no hay expresiones, el valor devuelto es falso *#t*

5. Operadores lógicos

- Conjunción lógica: and
  - **Ejemplos**

*(**define** a 1)*

*(**define** b 2)*

*(**and** (> a 0) (< a 10))* ➔ *#t*

Forma equivalente *(< 0 a 10)* ➔ *#t*

*(**and** (+ 2 a) "hola")* ➔ *"hola"*

*(**and** "hola" (+ 2 a))* ➔ *3*

*(**and** (> a 3) (< b 10))* ➔ *#f*

5. Operadores lógicos

- Conjunción lógica: and y begin
  - ***begin***
    - sentencia similar a "***and***" que poseen **algunos** intérpretes de *scheme*
  - **Sintaxis**

    (***begin*** exp_1 exp_2 … exp_n)
  - **Significado**
    - Se evalúan **todas** las expresiones o sentencias de **izquierda a derecha**
    - y se devuelve el resultado de la **última**.

5. Operadores lógicos

- Conjunción lógica: and y begin
  - **Ejemplo**

    *(**begin** (< 10 0) (display "valor ") (display 10))*

    ➔ *valor 10*

    (and (< 10 0) (display "valor ") (display 10))

    ➔ *#f*

  - **Observación**
    - Se suele usar *begin* junto con *if*, pero no es imprescindible.
    - La sentencia *begin* **no** es imprescindible, porque se puede usar la sentencia **cond**.

5. Operadores lógicos

- Disyunción lógica: *or*
- Conjunción lógica: *and*
- Negación lógica: *not*

5. Operadores lógicos

- Negación lógica: not
  - **Sintaxis**

    (***not*** *objeto)*

    - donde

      objeto

      puede ser cualquier elemento del lenguaje

      ❑ literal

      ❑ variable

      ❑ etc.

5. Operadores lógicos

- Negación lógica: not
  - **Significado**
    - Devuelve verdadero *#t* si el objeto tiene el valor falso *#f*
    - Devuelve falso *#f* en caso contrario

5. Operadores lógicos

- Negación lógica: not
  - **Ejemplos**

*(**define** a 1)*

(not a) ➔ *#f*

(not (= a 0)) ➔ *#t*

(not 10) ➔ *#f*

(not (< 0 a 10)) ➔ *#f*

5. Operadores lógicos

- Negación lógica: not
  - **Ejemplos**

```
(define a 1)
(define (distinto? x y)
    (not (= x y))
)
(distinto? a 0) ➔ #t
```

5. Operadores lógicos

- Negación lógica: not
  - **Ejemplos**

```
(define a 1)
(define (!= x y)
    (not (= x y))
)
(!= a 0) ➔ #t
```

Índice

6. Otros predicados

- Lista vacía
- Puertos de entrada y salida
- Fin de fichero

6. Otros predicados

- Lista vacía
- Puertos de entrada y salida
- Fin de fichero

6. Otros predicados

- Lista vacía
  - ***null?***
  - **Sintaxis**

    *(**null?** objeto)*
  - **Significado**
    - Devuelve *#t* si y solamente si el objeto es la lista vacía; en caso contario, devuelve *#f*
  - **Ejemplos**

    (null? '()) ➔ *#t*

    (null? '(a b c)) ➔ *#f*

6. Otros predicados

- Lista vacía
- Puertos de entrada y salida
- Fin de fichero

6. Otros predicados

- Puertos de entrada y salida
  - ***input-port?***
  - ***output-port?***
    - **Significado**
      - ❑ Comprueban sin un objeto es un puerto de entrada o salida, respectivamente.
    - **Observación**
      - Estos predicados se explicarán en el tema nº 7

6. Otros predicados

- Lista vacía
- Puertos de entrada y salida
- Fin de fichero

6. Otros predicados

- Fin de fichero
  - ***eof-object?***
  - **Sintaxis**

    *(**eof-object?** objeto)*
  - **Significado**
    - Devuelve
      - ✓ *#t* si el argumento es el **objeto fin de fichero #<eof>**
      - ✓ en caso contario, devuelve *#f*
  - **Ejemplos**
    - Se mostrarán en el tema nº 7

Índice

7. Sentencias condicionales

- if
- cond
- case

7. Sentencias condicionales

- if
- cond
- case

7. Sentencias condicionales

- if
  - **Sintaxis**

    *(**if** <condición>*

    <consecuente>

    [<alternativa>]

)

    - donde
      - ❑ *<condición>: expresión (generalmente, lógica)*
      - ❑ *<consecuente>: una sentencia de scheme*
      - ❑ *<alternativa>: una sentencia opcional*

7. Sentencias condicionales

- if
  - **Significado**
    - Si la <*condición*>
      - ❑ **no** es *#f* , se ejecuta la sentencia del <*consecuente*>
      - ❑ en caso contrario, se ejecuta la sentencia de la <*alternativa*>, si existe.

7. Sentencias condicionales

- if
  - **Ejemplos (1/5)**

(if 1

2

3

) ➔ 2

7. Sentencias condicionales

- if
  - **Ejemplos (2/5)**

```
(define x -9)
(if  (>= x 0)
     x
     (- x)
)  ➔ 9
```

7. Sentencias condicionales

- if
  - **Ejemplos (3/5)**

```
(define x 9)
(define y -3)
(if  (>= x 0)
     (sqrt x)
)  ➔ 3
(if  (>= y 0)
     (sqrt y)
)  ➔  no genera ningún resultado
```

7. Sentencias condicionales

- if
  - **Ejemplos (4/5)**

```
(define (valor_absoluto x)
   (if  (>= x 0)
            x
            (- x)
   )
)
```

Ejemplos (5/5): if anidado

```
(define (cuadrante x y)
  (if  (or (= x 0.0)  (= y 0.0))
       "eje de coordenadas"
       (if  (and (> x 0.0)  (> y 0.0))
            "primer cuadrante"
            (if (and (< x 0.0)  (> y 0.0))
                "segundo cuadrante"
                (if (and (< x 0.0)  (< y 0.0))
                    "tercer cuadrante"
                    (if (and (> x 0.0)  (< y 0.0))
                        "cuarto cuadrante"
                    )
                )
            )
       )
  )
)
```

7. Sentencias condicionales

- if y begin
  - **Observación**
    - Se puede usar ***begin*** para que el consecuente y la alternativa tengan **más** de una sentencia
    - En ese caso, es preferible usar la sentencia ***cond***
  - **Ejemplo**

```
(if (> x 0)
   (begin (display "número positivo o cero ") (display x))
   (begin (display "número negativo ") (display x))
 )
```

7. Sentencias condicionales

- if
- cond
- case

7. Sentencias condicionales

- cond
  - **Sintaxis**

*(**cond***

<cláusula$_1$>

[<cláusula$_2$>

…

<cláusula$_n$>

*<cláusula de **else**>]*

)

7. Sentencias condicionales

- cond
  - **Sintaxis**
    - donde
      - ❑ *<cláusula>*

        (<condición> => <sentencia(s)>)

        ✓ *Los símbolos "=>" son opcionales*

      - ❑ *<cláusula de else>*

        *(**else** <sentencia(s)>)*

7. Sentencias condicionales

- cond
  - **Sintaxis**
    - Versión completa **con** =>

*(**cond***

(<condición$_1$> => <sentencia(s)$_1$>)

[(<condición$_2$> => <sentencia(s)$_2$>)

…

(<condición$_n$> => <sentencia(s)$_n$>)

*(**else** <sentencia(s) $_{n+1}$>)]*

)

7. Sentencias condicionales

- cond
  - **Sintaxis**
    - Versión completa sin =>

*(**cond***

(<condición$_1$> <sentencia(s)$_1$>)

[(<condición$_2$> <sentencia(s)$_2$>)

…

(<condición$_n$> <sentencia(s)$_n$>)

*(**else** <sentencia(s) $_{n+1}$>)]*

)

7. Sentencias condicionales

- cond
  - **Significado**
    - Se evalúan consecutivamente las condiciones de las cláusulas
    - Si una condición no es *#f* entonces
      - ❑ se ejecutan las sentencias asociadas
      - ❑ y se devuelve el **valor** de la **última sentencia**
    - Si todas las condiciones tienen el valor *#f* entonces
      - ❑ se ejecutan las sentencias de la cláusula **else**, si existe,
      - ❑ y se devuelve el **valor** de la **última sentencia**

7. Sentencias condicionales

- con
  - **Ejemplos (1/8)**

```
(define (valor_absoluto x)
        (cond
                ((>= x 0)          x)
                (else              (- x))
        )
)
```

(valor_absoluto -9) ➔ *9*

7. Sentencias condicionales

- cond
  - **Ejemplos (2/8)**

```
(define (estado temperatura)
  (cond
    ((< temperatura 0.0)          "hielo")
    ((<= 0.0 temperatura 100.0)   "agua")
    ((> temperatura 100.0)        "vapor")
    )
  )
```

    - *Las cláusulas se pueden intercambiar*

7. Sentencias condicionales

- cond
  - **Ejemplos (3/8)**

```
(define (estado temperatura)
  (cond
    ((< temperatura 0.0)        "hielo")
    ((< temperatura 100.0)      "agua")
    ((< 100.0 temperatura)      "vapor")
    )
  )
```

    - *Las cláusulas no se pueden intercambiar*

7. Sentencias condicionales

- cond
  - **Ejemplos (4/8):** cláusulas intercambiables

```
(define (calificacion nota)
  (cond  ((<= 0.0 nota 4.9)   “suspenso”)
         ((<= 5.0 nota 6.9)   “aprobado”)
         ((<= 7.0 nota 8.9)   “notable”)
         ((<= 9.0 nota 9.9)   “sobresaliente”)
         ((= nota 10.0)       “matrícula de honor”)
         (else                (display “calificación incorrecta"))
  )
)
```

7. Sentencias condicionales

- cond

  - **Ejemplos (5/8):** cláusulas no intercambiables

```
(define (calificacion nota)
  (cond  ((<= 0.0 nota 4.9)   "suspenso")
         ((<=  nota 6.9)      "aprobado")
         ((<=  nota 8.9)      "notable")
         ((<= nota 9.9)       "sobresaliente")
         ((= nota 10.0)       "matrícula de honor")
         (else                (display "calificación incorrecta"))
  )
)
```

7. Sentencias condicionales

- cond
  - **Ejemplos (6/8):** cláusulas con varias sentencias

```
(define (calificacion-baremo nota)
  (cond
    ((<= 0.0 nota 4.9) (display "suspenso")                (newline)   0)
    ((<= 5.0 nota 6.9) (display "aprobado")                (newline)   1)
    ((<= 7.0 nota 8.9) (display "notable")                 (newline)   2)
    ((<= 9.0 nota 9.9) (display "sobresaliente")           (newline)   3)
    ((= nota 10.0)     (display "matrícula de honor") (newline)   4)
    (else              (display "calificación incorrecta") (newline))
    )
  )
```

7. Sentencias condicionales

- cond
  - **Ejemplos (7/8):** cláusulas intercambiables

```
(define (cuadrante x y)
  (cond
    ((or  (= x 0.0)  (= y 0.0))   "eje de coordenadas")
    ((and (> x 0.0)  (> y 0.0))  "primer cuadrante")
    ((and (< x 0.0)  (> y 0.0))  "segundo cuadrante")
    ((and (< x 0.0)  (< y 0.0))  "tercer cuadrante")
    ((and (> x 0.0)  (< y 0.0))  "cuarto cuadrante")
    )
  )
```

Ejemplos (8/8): “cond” anidado

```
(define (cuadrante-bis x y)
  (cond
    ((or  (= x 0.0)  (= y 0.0))  “eje de coordenadas”)
    ((> x 0.0)
          (cond
            ((> y 0.0)  “primer cuadrante”)
            ((< y 0.0)  “cuarto cuadrante”)
          )
     )
    ((< x 0.0)
          (cond
            ((> y 0.0)   “segundo cuadrante”)
            ((< y 0.0)   “tercer cuadrante”)
            )
          )
    )
  )
```

7. Sentencias condicionales

- if
- cond
- case

7. Sentencias condicionales

- case
  - **Sintaxis**

*(**case** <expresión>*

<cláusula$_1$>

[<cláusula$_2$>

…

<cláusula$_n$>

*<cláusula de **else**>]*

)

7. Sentencias condicionales

- case
  - **Sintaxis**
    - donde
      - ❑ *<cláusula>*

        ((<dato$_1$> <dato$_2$> … <dato$_n$>) <sentencia(s)>)

      - ❑ *<cláusula de else>*

        *(**else** <sentencia(s)>)*

7. Sentencias condicionales

- case
  - **Sintaxis**
    - Versión completa

*(**case** <expresión>*

((<dato$_{1,1}$> <dato$_{1,2}$> … <dato$_{1,k_1}$>) <sentencia(s)$_1$>)

[((<dato$_{2,1}$> <dato$_{2,2}$> … <dato$_{2,k_2}$>) <sentencia(s)$_2$>)

…

((<dato$_{n,1}$> <dato$_{n,2}$> … <dato$_{n,k_n}$>) <sentencia(s)$_n$>)

*(**else** <sentencia(s) $_{n+1}$>)]*

)

7. Sentencias condicionales

- case
  - **Significado**
    - Se obtiene el valor de la expresión
    - Se comprueba consecutivamente si el valor está en la lista de datos de cada cláusula usando el predicado eqv?
    - Si el valor **aparece** en una cláusula de datos entonces
      - ❑ se ejecutan las sentencias asociadas
      - ❑ y se devuelve el **valor** de la **última sentencia**
    - Si el valor no aparece en ninguna cláusula de datos entonces
      - ❑ se ejecutan las sentencias de la cláusula **else**, si existe,
      - ❑ y se devuelve el **valor** de la **última sentencia**

7. Sentencias condicionales

- case

  - **Ejemplos (1/3)**

```
(define (calificacion nota)
  (case nota
    ((0 1 2 3 4) "suspenso")
    ((5 6)       "aprobado")
    ((7 8)       "notable")
    ((9)         "sobresaliente")
    ((10)        "matrícula de honor")
    (else         (display "calificación incorrecta"))
    )
  )
```

7. Sentencias condicionales

- case
  - **Ejemplos (2/3)**

```
(define (control clave)
    (case (remainder clave 4)
        ((0)  "zona 0")
        ((1)  "zona 1")
        ((2)  "zona 2")
        ((3)  "zona 3")
        )
    )
```

7. Sentencias condicionales

- case

Ejercicio: codifica la función redondear

Ejemplos (3/3)

```
(define (calificacion-media teoria practica)
  (case (redondear (/ (+ teoria practica) 2.0))
    ((0. 1. 2. 3. 4.) "suspenso")
    ((5. 6.)          "aprobado")
    ((7. 8.)          "notable")
    ((9.)             "sobresaliente")
    ((10.)            "matrícula de honor")
    (else             (display "calificación incorrecta"))
    )
  )
```

UNIVERSIDAD DE CÓRDOBA

ESCUELA POLITÉCNICA SUPERIOR DE CÓRDOBA

DEPARTAMENTO DE
INFORMÁTICA Y ANÁLISIS NUMÉRICO

PROGRAMACIÓN DECLARATIVA

INGENIERÍA INFORMÁTICA

CUARTO CURSO

PRIMER CUATRIMESTRE

Tema 3.- Predicados y sentencias condicionales