Programación Declarativa

Ingeniería Informática
Cuarto curso. Primer cuatrimestre.

Escuela Politécnica Superior de Córdoba
Universidad de Córdoba

Curso académico: 2021 – 2022

Práctica número 4.- Tipos compuestos de datos y funciones con argumentos obligatorios y opcionales

VECTORES Y MATRICES

1. Codifica las siguientes funciones para el cómputo de **valores medios** de un vector $\vec{v} = (x_1, x_2, \ldots, x_n)$
   a) **Media aritmética**

$$\frac{x_1 + x_2 + \cdots + x_n}{n}$$

   - **Ejemplo**
     - *(**arithmeticMeanVector** #(1. 2. 3.))* ➔ *2.0*
     - *(**arithmeticMeanVector** #(1. 2. 3. 4. 5.))* ➔ *3.0*

   b) **Media geométrica**

$$\sqrt[n]{x_1 \times x_2 \times \ldots \times x_n}$$

   - **Ejemplo**
     - *(**geometricMeanVector** #(1. 2. 3.))* ➔ *1.817120593…*
     - *(**geometricMeanVector** #(1. 2. 3. 4. 5.))* ➔ *2.6051710…*

2. Codifica una función, denominada ***"miniMax"***, que reciba una matriz (no necesariamente cuadrada) y devuelva el mínimo de los valores máximos de las filas de la matriz.
   - **Ejemplo**
     - (***miniMax*** *#(#(1. 2. 3. 2.) #(4. 5. 6. 4.) #(7. 8. 9. 7.)))* ➔ *3.*

3. Determinante de una matriz de dimensión 3 x 3 y área del triángulo.
   a) Codifica una función que permita calcular el **determinante** de una matriz de dimensión 3 x 3:
   - $\begin{vmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{vmatrix} = x_1y_2z_3 + x_2y_3z_1 + x_3y_1z_2 - x_1y_3z_2 - x_2y_1z_3 - x_3y_2z_1$

   b) Utiliza la función anterior para calcular el **área de un triángulo** definido por sus vértices: $P_1=(x_1,y_1)$, $P_2=(x_2,y_2)$ y $P_3=(x_3,y_3)$
   - $área(P1, P2, P3) = valor\_absoluto\left(\frac{1}{2}\begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}\right)$

LISTAS Y PARÁMETROS OBLIGATORIOS U OPCIONALES

4. Codifica una función, denominada ***primosMenores***, que reciba un número natural y genere una lista con los números **primos menores o iguales** que dicho número.
    - **Ejemplos**
        - *(**primosMenores** 9)* ➔ *(2 3 5 7)*
        - *(**primosMenores** 11)* ➔ *(2 3 5 7 11)*
    - **Nota**:
        - Utiliza el predicado "***primo?***" codificado en la práctica nº 3.

5. Codifica una función, denominada ***descomposiciónEnPrimos*****,** que reciba un número natural y genere una lista con su **descomposición en números primos**.
    - **Ejemplos**
        - *(**descomposiciónEnPrimos** 2)* ➔ *(2)*
        - *(**descomposiciónEnPrimos** 12)* ➔ *(2 2 3)*
        - *(**descomposiciónEnPrimos** 60)* ➔ *(2 2 3 5)*

6. Codifica una función, denominada ***filtrarListaPrimos*****,** que reciba una **lista** de números naturales y devuelva otra lista compuesta por los números primos.
    - **Ejemplos**
        - (***filtrarListaPrimos*** **'*()***) → ()
        - (***filtrarListaPrimos*** **'(**2 3 4 5 6)) → (2 3 5)
        - (***filtrarListaPrimos*** **'(**2 3 4 5 6 7 8 9 10)) → (2 3 5 7)

7. Codifica una función denominada ***filtrarPrimos*** que reciba un número variable de números naturales y devuelva una lista compuesta por los números primos.
    - **Ejemplos**
        - (***filtrarPrimos***) → ()
        - (***filtrarPrimos*** 2 3 4 5 6) → (2 3 5)
        - (***filtrarPrimos*** 2 3 4 5 6 7 8 9 10 11 12) → (2 3 5 7 11)

8. Codifica una función denominada ***filtrarPrimosDelimitados*** que reciba dos números **"inicial"** y **"final"** (parámetros obligatorios) y un número variable de números naturales y devuelva una lista compuesta por los números **primos** delimitados por **"inicial"** y **"final"**.
    - **Ejemplos**
        - *(**filtrarPrimosDelimitados** 2 10) → ()*
        - *(**filtrarPrimosDelimitados** 2 10 3 4 5 6) → (3 5)*
        - *(**filtrarPrimosDelimitados** 2 10 3 4 5 6 7 8 9 10 11 12) → (3 5 7)*
        - *(**filtrarPrimosDelimitados** 10 2 3 4 5 6 7 8 9 10 11 12) → ()*

9. Codifica una función **recursiva**, denominada ***separar***, que reciba como parámetro una lista de números y los reparta en dos listas, dependiendo de

que ocupen un "lugar o posición" par o impar.

- Ejemplos
  - ***(separar** '()) → (() ())*
  - ***(separar** '(2)) → ((2) ())*
  - ***(separar** '(3 2)) → ((3) (2))*
  - ***(separar** '(1 3 2)) → ((1 2) (3))*
  - ***(separar** '(4 1 3 2)) → ((4 3) (1 2))*
  - ***(separar** '(5 4 1 3 2)) → → ((5 1 2) (4 3))*

10. Codifica una función **recursiva**, denominada ***unir***, que reciba como parámetros dos listas ordenadas de números y devuelva otra lista con los números ordenados
    - **Ejemplos**
      - ***(unir** '() '()) → ()*
      - ***(unir** '(1) '()) → (1)*
      - ***(unir** '(1) '(2)) → (1 2)*
      - ***(unir** '(1 3) '(2)) → (1 2 3)*
      - ***(unir** '(1 3) '(2 4 5)) → (1 2 3 4 5)*
      - ***(unir** '(1 3 5) '(2 4)) → (1 2 3 4 5)*
      - ***(unir** '() '(1 2 3 4 5))→ (1 2 3 4 5)*

11. Método de ordenación ***mergeSort***
    - **Descripción**
      - Datos de entrada : 5 4 1 3 2
        - División
          - ✓ Primera: 5 1 2 / 4 3
          - ✓ Segunda: 5 2 / 1 // 4 / 3
          - ✓ Tercera: 5 / 2 // 1 /// 4 / 3
        - Fusión:
          - ✓ Primera: 2 5 / 1 // 3 4
          - ✓ Segunda: 1 2 5 / 3 4
          - ✓ Tercera: 1 2 3 4 5
    - Codifica una función que permita ordenar una **lista** de números utilizando el método ***mergeSort***.
      - Ejemplo
        - ✓ (***mergeSort*** '(5 4 1 3 2)) → (1 2 3 4 5)
    - **Observación**
      - Utilizas las funciones "***separar***" y "***unir***" de los ejercicios anteriores.

12. Codifica una función denominada ***mergeSortDatos*** que permita ordenar una **cantidad variable** de números utilizando el método ***mergeSort***.
    - **Ejemplo**
      - (***mergeSortDatos***) → ()
      - (***mergeSortDatos*** 2) → (2)
      - (***mergeSortDatos*** 1 3 2) → 1 2 3)
      - (***mergeSortDatos*** 5 4 1 3 2) → (1 2 3 4 5)