Especificación matemática ejercicios 51-70

José Simón Ramos Sandoval

Universidad Nacional de Colombia, Programación de computadores 2021-1s

1 Matrices

Ejercicio 51
Desarrollar un algoritmo que permita multiplicar dos matrices de números reales (enteros).
1. Análisis y especificación
1.1 Objetos conocidos:

- Se conocen dos matrices a multiplicar
- Para poder multiplicar dos matrices, el número de columnas de la primer matriz debe ser igual al número de filas de la segunda matriz

1.2 Objetos desconocidos:

- Se desconocen los elementos de la matriz resultante de la multiplicación

1.3 Relación entre los objetos:

- Para la lectura de las matrices, se hará mediante una cadena de texto dada por el usuario. Ya que las matrices son listas de listas, si tenemos la matriz (lista) A^n, entonces A_0será también una lista. Donde A_0 (y en general cualquier A_n) representaría las filas de la matriz, y los elementos con la misma posición en los A_n representarían las columnas. En este sentido, la cadena de texto correspondendiente a una matriz vendría de la forma "2 3 4, 5 6 7, 8 9 1". Donde la "," separa a las filas entre sí (las listas de la lista de listas) y los espacios entre los números separa a los números que conformarían a las columnas
- La multiplicación entre dos matrices solo es posible si la primera matriz tiene el msimo número de columnas que filas la segunda matriz. Además de esto, las dimensiones de la matriz resultante tendrá igual número de filas que la primera matriz e igual número de columnas que la segunda. Es decir, si tenemos una matriz $A^{n\times m}$ y la matriz $B^{m\times p}$, la matriz resultante de $A \times B$ será $C^{n\times p}$
- Al multiplicar la matriz, se multiplican elemento por elemento de la primera fila de la primera matriz con la primera columna de la segunda matriz,

y la suma de estos resultados se ubica en la posición fila 1 columna 1 de la matriz resultante. Posteriormente se multiplican la primera fila de la primera matriz con la segunda columna de la segunda matriz y se repite el anterior proceso. Así que se debe multiplicar la primera fila de la primera matriz por cada columna de la segunda matriz. Este proceso se repite para todas las filas de la primera matriz

2. Diseño y prueba conceptual
Funciones que hacen la lectura:

Función que separa por una “,” y crea las filas:

Entradas: Una cadena de texto de longitud n ($s \in \mathbb{ASCII}^n$) con la representación de la matriz, esta cadena de texto debe venir de la forma “2 3 4, 5 6 7, 8 9 1”

Salidas: Un arreglo de cadenas de caracteres ($C \in (\mathbb{ASCII}^*)^*$)

Relación:

$$\begin{aligned} filas : \mathbb{ASCII}^n &\rightarrow (\mathbb{ASCII}^*)^* \\ filas(s) &\rightarrow D \text{ tal que } \forall_{i=0}^{n-1} D_{kj} = s_i \end{aligned}$$

Donde $k =$número de ”,” desde la posición 0 hasta la posición i y $j = i-$la posición + 1 del último ”,” antes de la posición i o 0 si no había

Función que separa por columnas:

Entradas: Una cadena de caracteres de tamañon ($s \in \mathbb{ASCII}^n$)

Salidas: Un arreglo de cadenas de caracteres ($C \in (\mathbb{ASCII}^*)^*$)

Relación:

$$\begin{aligned} columnas : \mathbb{ASCII}^n &\rightarrow \mathbb{R}^* \\ columnas(s) &\rightarrow C \text{ tal que } C + [float(s_i)] \text{si } \forall_{i=0}^{n-1} s_i \neq \text{” ”} \end{aligned}$$

$$\begin{aligned} columnas : \mathbb{ASCII}^n &\rightarrow (\mathbb{ASCII}^*)^* \\ columnas(s) &\rightarrow C \text{ tal que } \forall_{i=0}^{n-1} C_{kj} = s_i \end{aligned}$$

Donde $k =$número de “ ”(espacios en blanco) desde la posición 0 hasta la posición i y $j = i-$la posición + 1 del último “ ” antes de la posición i o 0 si no había

Función que crea la matriz apartir de la cadena de caracteres:

Entradas: Una cadena de caracteres ($s \in \mathbb{ASCII}^*$)

Salidas: Un arreglo de tamaño m de arreglos de tamaño n de cadenas de caracteres ($D \in (\mathbb{ASCII}^*)^m$)

Relación:

$$\begin{aligned} matriz : \mathbb{ASCII}^* &\rightarrow (\mathbb{ASCII}^*)^m \\ matriz(s) &\rightarrow D \text{ tal que } \forall_{i=0}^{m-1} D_i = columnas(A_i) \end{aligned}$$

Donde $A = filas(s)$ (es decir, las filas que tendría la matriz) y $m =$cantidad de elementos del arreglo A

Funciones que convierte los elementos en reales:

Entradas: Un arreglo de tamaño n de arreglos de tamaño m de cadenas de caracteres ($A \in (\mathbb{ASCII}^m)^n$)

Salidas: Un arreglo de tamaño n de arreglos de reales de tamaño m ($A \in (\mathbb{R}^m)^n$)

Relación:

$$\begin{array}{rcc} conviertereal : \mathbb{ASCII}^{n\times m} & \to & \mathbb{R}^{n\times m} \\ conviertereal(A) & \to & B \text{tal que } \forall_{i=0}^{n-1}(\forall_{j=0}^{m-1} B_{ij} = float(A_{ij})) \end{array}$$

Funciones que convierte los elementos en enteros:

Entradas: Un arreglo de tamaño n de arreglos de tamaño m de cadenas de caracteres ($A \in (\mathbb{ASCII}^m)^n$)

Salidas: Un arreglo de tamaño n de arreglos de reales de tamaño m ($A \in (\mathbb{Z}^m)^n$)

Relación:

$$\begin{array}{rcc} conviertereal : \mathbb{ASCII}^{n\times m} & \to & \mathbb{R}^{n\times m} \\ conviertereal(A) & \to & B \text{tal que } \forall_{i=0}^{n-1}(\forall_{j=0}^{m-1} B_{ij} = int(A_{ij})) \end{array}$$

Función que hace la impresión del resultado:

Entradas: Un arreglo de arreglos (matriz) de enteros o reales $C \in \mathbb{R}^{n\times m}$

Salidas: Una cadena de caracteres $e \in \mathbb{ASCII}^*$

Relación: Se crea una cadena de caracteres que representa la matriz, en la cual se genera una nueva línea cada vez que se termina una fila de la matriz y todos los elementos de esta se separan por un espacio

$$\begin{array}{rcc} salida : \mathbb{R}^{n\times m} & \to & \mathbb{ASCII}^* \\ salida(C) & \to & e \text{ tal que } \forall_{i=0}^{n-1}(\forall_{j=0}^{m-1} e_{id} = str(C_{ij})) \end{array}$$

Donde $d = 0$ si $i = 0 \vee (2i + 1)$ En otro caso

Funciones que hacen la multiplicación:

Función que crea las filas de la nueva matriz:

Entradas: 3 variables: Dos matrices A, B de reales tales que ($A, B \in (\mathbb{R}^*)^*$) y A tiene igual número de columnas que B número de filas ($A^{n\times m}, B^{m\times p}$); Fila de A que será multiplicada ($k \in \mathbb{N}$) ***Nota:*** Se determina que las matrices son de reales ya que este conjunto contiene al conjunto de los número enteros

Salidas: Un arreglo de reales de tamaño p ($C \in \mathbb{R}^p$)

Relación:

$filmatriz : \mathbb{R}^{n\times m} \times \mathbb{R}^{m\times p} \times \mathbb{N} \to \mathbb{R}^p$

$filmatriz : (A, B, k) \to C$ tal que $\forall_{i=0}^{p-1}(\sum_{j=0}^{m-1}(A_{kj} * B_{ji}) = C_i)$

Función que crea la matriz producto:

Entradas: Dos matrices A, B de reales tales que $(A, B \in (\mathbb{R}^*)^*)$ y A tiene igual número de columnas que B número de filas $(A^{n\times m}, B^{m\times p})$ ***Nota:*** Se determina que las matrices son de reales ya que este conjunto contiene al conjunto de los número enteros

Salidas: Una matriz de reales $(D \in \mathbb{R}^{n\times p})$ ***Nota:*** Se determina que la matriz es de reales ya que este conjunto contiene al conjunto de los número enteros

Relación:

$colmatriz : \mathbb{R}^{n\times m} \times \mathbb{R}^{m\times p} \to \mathbb{R}^{n\times p}$

$colmatriz(A, B) \to D$ tal que $\forall_{i=0}^{n-1} D_i = filmatriz(A, B, i)$

==

Función que realiza la multiplicación:

Entradas: Dos matrices A, B de reales tales que $(A, B \in (\mathbb{R}^*)^*)$ y A tiene igual número de columnas que B número de filas $(A^{n\times m}, B^{m\times p})$ ***Nota:*** Se determina que las matrices son de reales ya que este conjunto contiene al conjunto de los número enteros

Salidas: Una matriz de reales $(D \in \mathbb{R}^{n\times p})$ ***Nota:*** Se determina que la matriz es de reales ya que este conjunto contiene al conjunto de los número enteros

Relación:

$multiplicación : \mathbb{R}^{n\times m} \times \mathbb{R}^{m\times p} \to \mathbb{R}^{n\times p}$

$multiplicación : (A, B) \to D$ tal que $\forall_{k=0}^{n-1}(\forall_{i=0}^{p-1}(\sum_{j=0}^{m-1}(A_{kj} * B_{ji}) = C_i) = D_k)$

==

$multiplicación : \mathbb{R}^{n\times m} \times \mathbb{R}^{m\times p} \to \mathbb{R}^{n\times p}$

$multiplicación : (A, B) \to D$ tal que $\forall_{k=0}^{n-1}(\forall_{i=0}^{p-1}(\sum_{j=0}^{m-1}(A_{kj} * B_{ji}) = D_{ki}))$

3. Codificación:

El código en el que se resuelve el ejercicio se encuentra en el archivo llamado «Ejercicio_51_SimonRamos.py»

Ejercicio 52

Desarrollar un programa que sume los elementos de una columna dada de una matriz.

1. Análisis y especificación

1.1 Objetos conocidos:

- Se conoce una matriz con sus elementos, además se sabe cuál es la columna que se quiere sumar

1.2 Objetos desconocidos:

- Se desconoce el resultado de sumar los elementos de la columna de la matriz

1.3 Relación entre los objetos:

- Para la lectura de las matrices, se hará mediante una cadena de texto dada por el usuario. Ya que las matrices son listas de listas, si tenemos la matriz (lista) A^n, entonces A_0será también una lista. Donde A_0 (y en general cualquier A_n) representaría las filas de la matriz, y los elementos con la misma posición en los A_n representarían las columnas. En este sentido, la cadena de texto correspondendiente a una matriz vendría de la forma “2 3 4, 5 6 7, 8 9 1”. Donde la “,” separa a las filas entre sí (las listas de la lista de listas) y los espacios entre los números separa a los números que conformarían a las columnas

- Para hallar el resultado de esta suma, se debe saber cuál columna se desea sumar. Una vez teniendo esta información, el índice que da la posición de la columna se mantiene constante mientras que el índice que indica la fila va cambiando. El resultado se obtendrá al ir sumando todos esos elementos que se encuentran en los subíndices señalados

2. Diseño y prueba conceptual

Funciones que hacen la lectura:

Función que separa por una “,” y crea las filas:

Entradas: Una cadena de texto de longitud n ($s \in \mathbb{ASCII}^n$) con la representación de la matriz, esta cadena de texto debe venir de la forma “2 3 4, 5 6 7, 8 9 1”

Salidas: Un arreglo de cadenas de caracteres ($C \in (\mathbb{ASCII}^*)^*$)

Relación:

$$\begin{array}{rcc} filas : \mathbb{ASCII}^n & \rightarrow & (\mathbb{ASCII}^*)^* \\ filas(s) & \rightarrow & D \text{ tal que } \forall_{i=0}^{n-1} D_{kj} = s_i \end{array}$$

Donde $k =$número de ”,” desde la posición 0 hasta la posición i y $j = i-$la posición + 1 del último ”,” antes de la posición i o 0 si no había

Función que separa por columnas:

Entradas: Una cadena de caracteres de tamañon ($s \in \mathbb{ASCII}^n$)

Salidas: Un arreglo de cadenas de caracteres ($C \in (\mathbb{ASCII}^*)^*$)

Relación:

$$\begin{array}{rcc} columnas : \mathbb{ASCII}^n & \rightarrow & \mathbb{R}^* \\ columnas(s) & \rightarrow & C \text{ tal que } C + [float(s_i)]\text{si } \forall_{i=0}^{n-1} s_i \neq \text{” ”} \end{array}$$

$$\begin{array}{rcl} columnas : \mathbb{ASCII}^n & \rightarrow & (\mathbb{ASCII}^*)^* \\ columnas(s) & \rightarrow & C \text{ tal que } \forall_{i=0}^{n-1} C_{kj} = s_i \end{array}$$

Donde k =número de “ ”(espacios en blanco) desde la posición 0 hasta la posición i y $j = i-$la posición + 1 del último “ ” antes de la posición i o 0 si no había

Función que aplica la funcion columna y convierte a reales:

Entradas: Una lista de cadena de caractes de tamaño m ($A \in (\mathbb{ASCII}^*)^m$)

Salidas: Un arreglo de tamaño m de arreglos de tamaño k de reales ($D \in (\mathbb{R}^k)^m$)

Relación:

$$\begin{array}{rcl} matrizreales : \mathbb{ASCII}^* & \rightarrow & (\mathbb{ASCII}^*)^m \\ matrizreales(A) & \rightarrow & D \text{ tal que } (\forall_{i=0}^{m-1} (\forall_{j=0}^{k} D_{ij} = float(k_j))) \end{array}$$

$k = columnas(A_i)$ (es decir, los elementos de las columnas de la matriz) y m =cantidad de elementos del arreglo A

Funcion final que crea la matriz:

Entradas: Una cadena de caracteres ($s \in \mathbb{ASCII}^*$)

Salidas: Un arreglo de tamaño m de arreglos de reales de tamaño k ($A \in (\mathbb{R}^k)^m$)

Relación:

$$\begin{array}{rcl} matriz : \mathbb{ASCII}^* & \rightarrow & (\mathbb{R}^k)^m \\ matriz(s) & \rightarrow & matrizreales(filas(s)) \end{array}$$

Función que hace la suma de la columna

Entradas: 2 variables: Una matriz $A \in \mathbb{R}^{n \times m}$ y el índice de la columna que se desa sumar ($k : k \in \mathbb{N} \wedge \in [1, m]$) ($k \in \mathbb{N}^{[1,m]}$)

Salidas: 1 variable: La suma de los elementos de la columna ($suma \in \mathbb{R}$)

Relación:

$sumacol : \mathbb{R}^{n \times m} \times \mathbb{N}^{[1,m]} \rightarrow \mathbb{R}$

$sumacol(A, k) \rightarrow \sum_{i=0}^{n-1} A_{i,k-1}$

3. Codificación:

El código en el que se resuelve el ejercicio se encuentra en el archivo llamado «Ejercicio_52_SimonRamos.py»

Ejercicio 53

Desarrollar un programa que sume los elementos de una fila dada de una matriz.

1. Análisis y especificación

1.1 Objetos conocidos:

- Se conoce una matriz con sus elementos, además se sabe cuál es la fila que se quiere sumar

1.2 Objetos desconocidos:

- Se desconoce el resultado de sumar los elementos de la fila de la matriz

1.3 Relación entre los objetos:

- Para la lectura de las matrices, se hará mediante una cadena de texto dada por el usuario. Ya que las matrices son listas de listas, si tenemos la matriz (lista) A^n, entonces A_0será también una lista. Donde A_0 (y en general cualquier A_n) representaría las filas de la matriz, y los elementos con la misma posición en los A_n representarían las columnas. En este sentido, la cadena de texto correspondendiente a una matriz vendría de la forma “2 3 4, 5 6 7, 8 9 1”. Donde la “,” separa a las filas entre sí (las listas de la lista de listas) y los espacios entre los números separa a los números que conformarían a las columnas

- Para hallar el resultado de esta suma, se debe saber cuál fila se desea sumar. Una vez teniendo esta información, el índice que da la posición de la fila se mantiene constante mientras que el índice que indica la columna va cambiando. El resultado se obtendrá al ir sumando todos esos elementos que se encuentran en los subíndices señalados

2. Diseño y prueba conceptual
Funciones que hacen la lectura:

Función que separa por una “,” y crea las filas:

Entradas: Una cadena de texto de longitud n ($s \in \mathbb{ASCII}^n$) con la representación de la matriz, esta cadena de texto debe venir de la forma “2 3 4, 5 6 7, 8 9 1”

Salidas: Un arreglo de cadenas de caracteres ($C \in (\mathbb{ASCII}^*)^*$)

Relación:

$$\begin{array}{rcc} filas : \mathbb{ASCII}^n & \to & (\mathbb{ASCII}^*)^* \\ filas(s) & \to & D \text{ tal que } \forall_{i=0}^{n-1} D_{kj} = s_i \end{array}$$

Donde k =número de ”,” desde la posición 0 hasta la posición i y $j = i-$la posición + 1 del último ”,” antes de la posición i o 0 si no había

Función que separa por columnas:

Entradas: Una cadena de caracteres de tamañon ($s \in \mathbb{ASCII}^n$)

Salidas: Un arreglo de cadenas de caracteres ($C \in (\mathbb{ASCII}^*)^*$)

Relación:

$$\begin{array}{rcc} columnas : \mathbb{ASCII}^n & \to & \mathbb{R}^* \\ columnas(s) & \to & C \text{ tal que } C + [float(s_i)] \text{si } \forall_{i=0}^{n-1} s_i \neq \text{" "} \end{array}$$

$$\begin{array}{rcl} columnas : \mathbb{ASCII}^n & \rightarrow & (\mathbb{ASCII}^*)^* \\ columnas(s) & \rightarrow & C \text{ tal que } \forall_{i=0}^{n-1} C_{kj} = s_i \end{array}$$

Donde k =número de “ ”(espacios en blanco) desde la posición 0 hasta la posición i y $j = i-$la posición + 1 del último “ ” antes de la posición i o 0 si no había

Función que aplica la funcion columna y convierte a reales:

Entradas: Una lista de cadena de caractes de tamaño m ($A \in (\mathbb{ASCII}^*)^m$)

Salidas: Un arreglo de tamaño m de arreglos de tamaño k de reales ($D \in (\mathbb{R}^k)^m$)

Relación:

$$\begin{array}{rcl} matrizreales : \mathbb{ASCII}^* & \rightarrow & (\mathbb{ASCII}^*)^m \\ matrizreales(A) & \rightarrow & D \text{ tal que } (\forall_{i=0}^{m-1}(\forall_{j=0}^{k} D_{ij} = float(k_j))) \end{array}$$

$k = columnas(A_i)$ (es decir, los elementos de las columnas de la matriz) y m =cantidad de elementos del arreglo A

Funcion final que crea la matriz:

Entradas: **U**na cadena de caracteres ($s \in \mathbb{ASCII}^*$)

Salidas: Un arreglo de tamaño m de arreglos de reales de tamaño k ($A \in (\mathbb{R}^k)^m$)

Relación:

$$\begin{array}{rcl} matriz : \mathbb{ASCII}^* & \rightarrow & (\mathbb{R}^k)^m \\ matriz(s) & \rightarrow & matrizreales(filas(s)) \end{array}$$

Función que hace la suma de la fila

Entradas: 2 variables: Una matriz $A \in \mathbb{R}^{n\times m}$ y el índice de la fila que se desa sumar ($k : k \in \mathbb{N} \wedge \in [1, n]$) ($k \in \mathbb{N}^{[1,n]}$)

Salidas: 1 variable: La suma de los elementos de la fila ($suma \in \mathbb{R}$)

Relación:

$sumafil : \mathbb{R}^{n\times m} \times \mathbb{N}^{[1,n]} \rightarrow \mathbb{R}$

$sumafil(A, k) \rightarrow \sum_{j=0}^{m-1} A_{k-1,j}$

3. Codificación:

El código en el que se resuelve el ejercicio se encuentra en el archivo llamado «Ejercicio_53_SimonRamos.py»

Ejercicio 54

Desarrollar un algoritmo que determine si una matriz es mágica. Se dice que una matriz cuadrada es mágica si la suma de cada una de sus filas, de cada una de sus columnas y de cada diagonal es igual.

1. Análisis y especificación

1.1 Objetos conocidos:

- Se conoce una matriz cuadrada dada de número reales

1.2 Objetos desconocidos:

- Se desconoce si esta matriz es mágica o no

1.3 Relación entre los objetos:

- Para la lectura de las matrices, se hará mediante una cadena de texto dada por el usuario. Ya que las matrices son listas de listas, si tenemos la matriz (lista) A^n, entonces A_0será también una lista. Donde A_0 (y en general cualquier A_n) representaría las filas de la matriz, y los elementos con la misma posición en los A_n representarían las columnas. En este sentido, la cadena de texto correspondendiente a una matriz vendría de la forma "2 3 4, 5 6 7, 8 9 1". Donde la "," separa a las filas entre sí (las listas de la lista de listas) y los espacios entre los números separa a los números que conformarían a las columnas

- Una matriz es mágica si la suma de cada una de sus filas, de cada una de sus columnas y de cada diagonal es igual. Para comporbar esto, en primer lugar hallar la suma de las dos diagonales y colocarlas en un arreglo, este arreglo tendría solo dos elementos. Si estos dos elementos son iguales, entonces ahora comprobar si la suma de cada fila y de cada columna es igual al primer elemento de este arreglo creado. Esto comporbarlo apotandose con las funciones que se crearon para los dos puntos anteriores. SI los dos elementos no son iguales, entonces la matriz no es mágica

2. Diseño y prueba conceptual
Funciones que hacen la lectura:

Función que separa por una "," y crea las filas:

Entradas: Una cadena de texto de longitud n ($s \in \mathbb{ASCII}^n$) con la representación de la matriz, esta cadena de texto debe venir de la forma "2 3 4, 5 6 7, 8 9 1"

Salidas: Un arreglo de cadenas de caracteres ($C \in (\mathbb{ASCII}^*)^*$)

Relación:

$$\begin{array}{lcc} filas : \mathbb{ASCII}^n & \rightarrow & (\mathbb{ASCII}^*)^* \\ filas(s) & \rightarrow & D \text{ tal que } \forall_{i=0}^{n-1} D_{kj} = s_i \end{array}$$

Donde k =número de "," desde la posición 0 hasta la posición i y $j = i-$la posición + 1 del último "," antes de la posición i o 0 si no había

Función que separa por columnas:

Entradas: Una cadena de caracteres de tamañon ($s \in \mathbb{ASCII}^n$)

Salidas: Un arreglo de cadenas de caracteres ($C \in (\mathbb{ASCII}^*)^*$)

Relación:

$$\begin{array}{rcl} columnas : \mathbb{ASCII}^n & \to & \mathbb{R}^* \\ columnas(s) & \to & C \text{ tal que } C + [float(s_i)] \text{si } \forall_{i=0}^{n-1} s_i \neq \text{" "} \end{array}$$

$$\begin{array}{rcl} columnas : \mathbb{ASCII}^n & \to & (\mathbb{ASCII}^*)^* \\ columnas(s) & \to & C \text{ tal que } \forall_{i=0}^{n-1} C_{kj} = s_i \end{array}$$

Donde k =número de " "(espacios en blanco) desde la posición 0 hasta la posición i y $j = i-$la posición + 1 del último " " antes de la posición i o 0 si no había

Función que aplica la funcion columna y convierte a reales:

Entradas: Una lista de cadena de caractes de tamaño m ($A \in (\mathbb{ASCII}^*)^m$)

Salidas: Un arreglo de tamaño m de arreglos de tamaño k de reales ($D \in (\mathbb{R}^k)^m$)

Relación:

$$\begin{array}{rcl} matrizreales : \mathbb{ASCII}^* & \to & (\mathbb{ASCII}^*)^m \\ matrizreales(A) & \to & D \text{ tal que } (\forall_{i=0}^{m-1}(\forall_{j=0}^{k} D_{ij} = float(k_j))) \end{array}$$

$k = columnas(A_i)$ (es decir, los elementos de las columnas de la matriz) y m =cantidad de elementos del arreglo A

Funcion final que crea la matriz:

Entradas: **U**na cadena de caracteres ($s \in \mathbb{ASCII}^*$)

Salidas: Un arreglo de tamaño m de arreglos de reales de tamaño k ($A \in (\mathbb{R}^k)^m$)

Relación:

$$\begin{array}{rcl} matriz : \mathbb{ASCII}^* & \to & (\mathbb{R}^k)^m \\ matriz(s) & \to & matrizreales(filas(s)) \end{array}$$

Función que hace la suma de la columna

Entradas: 2 variables: Una matriz $A \in \mathbb{R}^{n\times m}$ y el índice de la columna que se desa sumar ($k : k \in \mathbb{N} \wedge \in [1, m]$) ($k \in \mathbb{N}^{[1,m]}$)

Salidas: 1 variable: La suma de los elementos de la columna ($suma \in \mathbb{R}$)

Relación:

$sumacol : \mathbb{R}^{n\times m} \times \mathbb{N}^{[1,m]} \to \mathbb{R}$

$sumacol(A, k) \to \sum_{i=0}^{n-1} A_{i,k-1}$

Función que hace la suma de la fila

Entradas: 2 variables: Una matriz $A \in \mathbb{R}^{n\times m}$ y el índice de la fila que se desa sumar ($k : k \in \mathbb{N} \wedge \in [1, n]$) ($k \in \mathbb{N}^{[1,n]}$)

Salidas: 1 variable: La suma de los elementos de la fila ($suma \in \mathbb{R}$)

Relación:

$sumafil : \mathbb{R}^{n\times m} \times \mathbb{N}^{[1,n]} \to \mathbb{R}$

$sumafil(A,k) \to \sum_{j=0}^{m-1} A_{k-1,j}$

Función que crea el arreglo con la suma de las diagonales:

Entradas: Una matriz cuadrada de reales ($A \in \mathbb{R}^{n\times n}$)

Salidas: Un arreglo de dos elementos con la suma de las diagonales ($B \in \mathbb{R}^2$)

Relación:

$diagonal : \mathbb{R}^{n\times n} \to \mathbb{R}^2$

$diagonal(A) \to B$ tal que$B = [\sum_{i=0}^{n-1}(A_{i,i})] + [\sum_{i=0}^{n-1} A_{i,-(i+1)}]$

Función que comprueba si las filas y columnas son iguales a un número:

Entradas: Dos variables: Una matriz cuadrada de reales $A \in \mathbb{R}^{n\times n}$ y un número reales $k \in \mathbb{R}$

Salidas: Un valor de verdad $v \in \mathbb{B}$, que indica si la suma de cada fila y de cada columna es igual al número dado

Relación:

$iguales : \mathbb{R}^{n\times n} \times \mathbb{R} \to \mathbb{B}$

$$iguales(A,k) \to \begin{cases} True \quad si & \forall_{i=0}^{n-1}(sumacol(A,i) = k \wedge sumafil(A,i) = x) \\ False & \text{En otro caso} \end{cases}$$

Función que determina si la suma de las diagonales son iguales a las columnas y filas:

Entradas: Dos variables: Una matriz cuadrada de reales $A \in \mathbb{R}^{n\times n}$ y un arreglo de dos elementos reales $B \in \mathbb{R}^2$

Salidas: Un valor de verdad $v \in \mathbb{B}$, que indica si la suma de las diagonales es igual a la suma de las columnas y de las filas

Relación:

$sumaigual : \mathbb{R}^{n\times n} \times \mathbb{R}^2 \to \mathbb{B}$

$$sumaigual : (A,B) \to \begin{cases} iguales(A,B_0) \quad si & B_0 = B_1 \\ False & \text{En otro caso} \end{cases}$$

Función que determina si es mágica o no:

Entradas: Una matriz cuadrada de reales $A \in \mathbb{R}^{n\times n}$

Salidas: Un valor de verdad $v \in \mathbb{B}$, que indica si la matriz es mágica o no

Relación:

$magica : \mathbb{R}^{n\times n} \to \mathbb{B}$

$magica(A) \to sumaigual(A, diagonal(A))$

3. Codificación:

El código en el que se resuelve el ejercicio se encuentra en el archivo llamado «Ejercicio_54_SimonRamos.py»

Ejercicio 55
Desarrollar un algoritmo que dado un entero, reemplace en una matriz todos los números mayores a un número dado por un uno y todos los menores o iguales por un cero.
1. Análisis y especificación
1.1 Objetos conocidos:

- Se conocen una matriz de reales y un número real

1.2 Objetos desconocidos:

- Se desconoce cuales números de la matriz son menores o iguales que el real dado para que sean reemplazados por un cero y cuales son mayores para que sean reemplazados por un uno

1.3 Relación entre los objetos:

- Se pasará por todos los números de la matriz, aquellos que sean mayores que el número dado se convertirán en un 1 y todos los menores o iguales se convertirán en un 0

2. Diseño y prueba conceptual
Funciones que hacen la lectura:

Función que separa por una "," y crea las filas:

Entradas: Una cadena de texto de longitud n ($s \in \mathbb{ASCII}^n$) con la representación de la matriz, esta cadena de texto debe venir de la forma "2 3 4, 5 6 7, 8 9 1"

Salidas: Un arreglo de cadenas de caracteres ($C \in (\mathbb{ASCII}^*)^*$)

Relación:

$$\begin{array}{lcl} filas : \mathbb{ASCII}^n & \to & (\mathbb{ASCII}^*)^* \\ filas(s) & \to & D \text{ tal que } \forall_{i=0}^{n-1} D_{kj} = s_i \end{array}$$

Donde k =número de "," desde la posición 0 hasta la posición i y $j = i-$la posición + 1 del último "," antes de la posición i o 0 si no había

Función que separa por columnas:

Entradas: Una cadena de caracteres de tamañon ($s \in \mathbb{ASCII}^n$)

Salidas: Un arreglo de cadenas de caracteres ($C \in (\mathbb{ASCII}^*)^*$)

Relación:

$$\begin{array}{lcl} columnas : \mathbb{ASCII}^n & \to & \mathbb{R}^* \\ columnas(s) & \to & C \text{ tal que } C + [float(s_i)] \text{si } \forall_{i=0}^{n-1} s_i \neq \text{" "} \end{array}$$

$$\begin{array}{rcl} columnas : \mathbb{ASCII}^n & \rightarrow & (\mathbb{ASCII}^*)^* \\ columnas(s) & \rightarrow & C \text{ tal que } \forall_{i=0}^{n-1} C_{kj} = s_i \end{array}$$

Donde k =número de “ ”(espacios en blanco) desde la posición 0 hasta la posición i y $j = i-$la posición + 1 del último “ ” antes de la posición i o 0 si no había

Función que aplica la funcion columna y convierte a reales:

Entradas: Una lista de cadena de caractes de tamaño m ($A \in (\mathbb{ASCII}^*)^m$)

Salidas: Un arreglo de tamaño m de arreglos de tamaño k de reales ($D \in (\mathbb{R}^k)^m$)

Relación:

$$\begin{array}{rcl} matrizreales : \mathbb{ASCII}^* & \rightarrow & (\mathbb{ASCII}^*)^m \\ matrizreales(A) & \rightarrow & D \text{ tal que } (\forall_{i=0}^{m-1}(\forall_{j=0}^{k} D_{ij} = float(k_j))) \end{array}$$

$k = columnas(A_i)$ (es decir, los elementos de las columnas de la matriz) y m =cantidad de elementos del arreglo A

Funcion final que crea la matriz:

Entradas: **U**na cadena de caracteres ($s \in \mathbb{ASCII}^*$)

Salidas: Un arreglo de tamaño m de arreglos de reales de tamaño k ($A \in (\mathbb{R}^k)^m$)

Relación:

$$\begin{array}{rcl} matriz : \mathbb{ASCII}^* & \rightarrow & (\mathbb{R}^k)^m \\ matriz(s) & \rightarrow & matrizreales(filas(s)) \end{array}$$

Función que hace la impresión del resultado:

Entradas: Un arreglo de arreglos (matriz) de reales $C \in \mathbb{R}^{n\times m}$

Salidas: Una cadena de caracteres $e \in \mathbb{ASCII}^*$

Relación: Se crea una cadena de caracteres que representa la matriz, en la cual se genera una nueva línea cada vez que se termina una fila de la matriz y todos los elementos de esta se separan por un espacio

$$\begin{array}{rcl} salida : \mathbb{R}^{n\times m} & \rightarrow & \mathbb{ASCII}^* \\ salida(C) & \rightarrow & e \text{ tal que } \forall_{i=0}^{n-1}(\forall_{j=0}^{m-1} e_{id} = str(C_{ij})) \end{array}$$

Donde $d = 0$ si $i = 0 \wedge (2i + 1)$ En otro caso

Funcion que reeemplaza los mayores:

Entradas: Dos variables: Una matriz de reales $A \in \mathbb{R}^{n\times m}$y un número real $x \in \mathbb{R}$

Salidas: Una matriz de unos y ceros $B \in \{0, 1\}^{n\times m}$

Relación:

$mayores : \mathbb{R}^{n\times m} \times \mathbb{R} \rightarrow \{0, 1\}^{n\times m}$

$$mayores(A,x) \rightarrow B \text{ tal que } \forall_{i=0}^{n-1}(\forall_{j=0}^{m-1} \begin{cases} B_{ij} = 1 \quad si & B_{ij} > x \\ B_{ij} = 0 & \text{En otro caso} \end{cases})$$

Nota: En este caso en la codificación las modificaciones se hacen sobre el mismo arreglo A, por lo que en un inicio A = B, pero posteriormente las modificaciones de la función llevan a que A se componga exclusivamente de unos y ceros

3. Codificación:

El código en el que se resuelve el ejercicio se encuentra en el archivo llamado «Ejercicio_55_SimonRamos.py»

Ejercicio 56
Desarrollar un programa que calcule el determinante de una matriz cuadrada.
1. Análisis y especificación
1.1 Objetos conocidos:

- Se conocen una matriz de reales

1.2 Objetos desconocidos:

- Se desconoce el determinante de esta matriz de reales

1.3 Relación entre los objetos:

- El determinante se calculará mediante Gauss-Jordan. Esto quiere decir, los elementos pivote serán los elementos de la diagonal de la matriz, yendo de fila en fila, primero se dividirán todos lo elementos de esta fila por el pivote correspondendiente. Posteriormente se reducíran los elementos de la columna del pivote a 0, haciendo que todos los elementos sean cero y el pivote 1. Si en algún momento un elemento pivote es cero, se buscará si alguna de las filas siguientes tiene un elemento distinto de cero en esta posición pivote y se intercambiaran filas. Este procedimiento se hará para todas las filas. . Al final, el determinante será igual a la multiplicación de todos los pivotes en el proceso de reducción de la matriz, si el número de intercambios de filas es par el determinante será positivo, de lo contrario será negativo

2. Diseño y prueba conceptual
Funciones que hacen la lectura:

Función que separa por una "," y crea las filas:

Entradas: Una cadena de texto de longitud n ($s \in \mathbb{ASCII}^n$) con la representación de la matriz, esta cadena de texto debe venir de la forma "2 3 4, 5 6 7, 8 9 1"

Salidas: Un arreglo de cadenas de caracteres ($C \in (\mathbb{ASCII}^*)^*$)

Relación:

$$\begin{array}{lcl} filas : \mathbb{ASCII}^n & \to & (\mathbb{ASCII}^*)^* \\ filas(s) & \to & D \text{ tal que } \forall_{i=0}^{n-1} D_{kj} = s_i \end{array}$$

Donde k =número de ”,” desde la posición 0 hasta la posición i y $j = i-$la posición + 1 del último ”,” antes de la posición i o 0 si no había

Función que separa por columnas:

Entradas: Una cadena de caracteres de tamañon ($s \in \mathbb{ASCII}^n$)

Salidas: Un arreglo de cadenas de caracteres ($C \in (\mathbb{ASCII}^*)^*$)

Relación:

$$\begin{array}{lcl} columnas : \mathbb{ASCII}^n & \to & \mathbb{R}^* \\ columnas(s) & \to & C \text{ tal que } C + [float(s_i)] \text{si } \forall_{i=0}^{n-1} s_i \neq \text{” ”} \end{array}$$

$$\begin{array}{lcl} columnas : \mathbb{ASCII}^n & \to & (\mathbb{ASCII}^*)^* \\ columnas(s) & \to & C \text{ tal que } \forall_{i=0}^{n-1} C_{kj} = s_i \end{array}$$

Donde k =número de “ ”(espacios en blanco) desde la posición 0 hasta la posición i y $j = i-$la posición + 1 del último “ ” antes de la posición i o 0 si no había

Función que aplica la funcion columna y convierte a reales:

Entradas: Una lista de cadena de caractes de tamaño m ($A \in (\mathbb{ASCII}^*)^m$)

Salidas: Un arreglo de tamaño m de arreglos de tamaño k de reales ($D \in (\mathbb{R}^k)^m$)

Relación:

$$\begin{array}{lcl} matrizreales : \mathbb{ASCII}^* & \to & (\mathbb{ASCII}^*)^m \\ matrizreales(A) & \to & D \text{ tal que } (\forall_{i=0}^{m-1} (\forall_{j=0}^{k} D_{ij} = float(k_j))) \end{array}$$

$k = columnas(A_i)$ (es decir, los elementos de las columnas de la matriz) y m =cantidad de elementos del arreglo A

Funcion final que crea la matriz:

Entradas: **U**na cadena de caracteres ($s \in \mathbb{ASCII}^*$)

Salidas: Un arreglo de tamaño m de arreglos de reales de tamaño k ($A \in (\mathbb{R}^k)^m$)

Relación:

$$\begin{array}{lcl} matriz : \mathbb{ASCII}^* & \to & (\mathbb{R}^k)^m \\ matriz(s) & \to & matrizreales(filas(s)) \end{array}$$

Funciones que calculan el determinante:

Función que hace los intercambios de filas:

Entradas: Una matriz cuadrada de reales ($A \in \mathbb{R}^{n\times n}$), la posición de la fila donde el pivote es cero ($i \in \mathbb{N}$)

Salidas: Una matriz cuadrada de reales ($A \in \mathbb{R}^{n\times n}$)

Relación:

$inter : \mathbb{R}^{n\times n} \times \mathbb{N} \rightarrow \mathbb{R}^{n\times n}$

$inter(A, i) \rightarrow$ tal que $\exists_{j=i}^{n-1} A_{ji} \neq 0$ Entonces $A_i \longleftrightarrow A_j$

Función que reduce la fila y columna del pivote:

Entradas: Una matriz cuadrada de reales ($A \in \mathbb{R}^{n\times n}$), posición de la fila y columna a reducir ($i \in \mathbb{N}$)

Salidas: Una matriz cuadrada de reales ($D \in \mathbb{R}^{n\times n}$), el pivote de esa fila ($piv \in \mathbb{R}$)

Relación:

$reduce : \mathbb{R}^{n\times n} \times \mathbb{N} \rightarrow \mathbb{R}^{n\times n} \times \mathbb{R}$

$reduce(A, i) \rightarrow D$ tal que $\forall_{j=0}^{m-1}(D_{ij} = \frac{A_{ij}}{A_{ii}}) \wedge \forall_{k=0}^{n-1}(D_{kj} = A_{kj} - (A_{ki} * A_{ij}))$ si $A_{ki} \neq 0 \wedge k \neq i, piv = A_{ii}$

Nota: En la codificación el procedimiento se hará sobre la misma matriz A

Funsión que comprueba si hay que intercambiar filas

Entradas: Una matriz cuadrada de reales ($A \in \mathbb{R}^{n\times n}$), variable acumuladora de veces que se hace el intercambio de filas ($x \in \mathbb{N}$), fila y columna a la que se le hará el proceso ($i \in \mathbb{N}$)

Salidas: Una matriz cuadrada de reales ($A \in \mathbb{R}^{n\times n}$), pivote de la fila a la que se le hizo el proceso ($piv \in \mathbb{R}$), veces que se hace el intercambio de filas ($x_{(j)} \in \mathbb{N}$),

Relación:

$integra : \mathbb{R}^{n\times n} \times \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{R}^{n\times n} \times \mathbb{R} \times \mathbb{N}$

$$integra(A, x, i) \rightarrow \begin{cases} reduce(inter(A, i), i), x_{(j)} = x_{(0)} + 1 \quad si & A_{ii} = 0 \\ reduce(A, i), x & \text{En otro caso} \end{cases}$$

Función que calcula el determinante:

Entradas: Dos variables: cantidad de veces que se hicieron los intercambios de filas ($x \in \mathbb{N}$), productoria de todos los pivotes de la matriz (determinante) ($det \in \mathbb{R}$)

Salidas: Un real que corresponde con el determinante de la matriz $\mathbb{R}$

Relación:

$determinante : \mathbb{N} \times \mathbb{R} \rightarrow \mathbb{R}$

$$determinante(x, det) \rightarrow \begin{cases} det \quad si & x \bmod 2 = 0 \\ -det & \text{En otro caso} \end{cases}$$

Función que reduce la matriz de forma recursiva

Entradas: Una matriz cuadrada de reales ($A \in \mathbb{R}^{n\times n}$), variable acumuladora de veces que se hace el intercambio de filas ($x \in \mathbb{N}$), fila y columna a la que se le hará el proceso ($i \in \mathbb{N}$), variable acumuladora de los pivotes de las filas ($det \in \mathbb{R}$)

Salidas: Determinante de la matriz $\mathbb{R}$

Relación:

$toda2 : \mathbb{R}^{n\times n} \times \mathbb{N} \times \mathbb{N} \times \mathbb{R}$

$$toda2(A, x, det, i) \to \begin{cases} determinante(x, det) & si \\ A, piv, x = integra(A, x, |A| - i) \to toda2(A, x, i-1, det_j = piv * det_0) \end{cases}$$

Función final que calcula el determinante con las anteriores funciones

Entradas: Una matriz cuadrada de reales ($A \in \mathbb{R}^{n\times n}$)

Salidas: un real que se corresponde con el determinante de la matriz $det \in \mathbb{R}$

Relación:

$final : \mathbb{R}^{n\times n} \to \mathbb{R}^n$

$final(A) \to toda2(A, x, |A|, det)$

Donde $x = 0$ y $det = 1$

3. Codificación

El código en el que se resuelve el ejercicio se encuentra en el archivo llamado «Ejercicio_56_SimonRamos.py»

Ejercicio 57:

Desarrollar un programa que dadas una matriz cuadrada A y un arreglo de números reales del mismo tamaño B, calcule una solución x para el sistema de ecuaciones lineales Ax = B.

1. Análisis y especificación

1.1 Objetos conocidos:

- Se conocen una matriz de reales que se corresponde a los coeficientes de las variables en el sistema de ecuaciones y un arreglo de reales que se corresponde con la solución de cada ecuación en el sistema

1.2 Objetos desconocidos:

- Se desconoce el valor de cada variable que da solución al sistema de ecuaciones

1.3 Relación entre los objetos:

- Al tener el sistema de ecuaciones lineales $Ax = B$, para poder resolverlo se debe primero tener una matriz extendida de A y B, posteriormente llevar a A a una matriz identidad mientras se modifica a toda la matriz extendida utilizando Gauss-Jordan. El arreglo resultante B después de las modificaciones que llevaron a A a convertirse en una matriz extendida, se corresponderá con las soluciones del sistema de ecuaciones lineales. Cuando un elemento pivote es cero, se buscará en las siguientes filas aquella que no tenga cero en la posición pivote y se intercambiarán. AL final, el valor B_0 será igual al valor de la primera variable del sistema, B_1 de la segunda y así sucesivamente hasta B_{n-1}

2. Diseño y prueba conceptual
Funciones que hacen la lectura:

Función que separa por una "," y crea las filas:

Entradas: Una cadena de texto de longitud n ($s \in \mathbb{ASCII}^n$) con la representación de la matriz, esta cadena de texto debe venir de la forma "2 3 4, 5 6 7, 8 9 1"

Salidas: Un arreglo de cadenas de caracteres ($C \in (\mathbb{ASCII}^*)^*$)

Relación:

$$\begin{array}{rcc} filas : \mathbb{ASCII}^n & \to & (\mathbb{ASCII}^*)^* \\ filas(s) & \to & D \text{ tal que } \forall_{i=0}^{n-1} D_{kj} = s_i \end{array}$$

Donde $k =$número de "," desde la posición 0 hasta la posición i y $j = i-$la posición + 1 del último "," antes de la posición i o 0 si no había

Función que separa por columnas:

Entradas: Una cadena de caracteres de tamañon ($s \in \mathbb{ASCII}^n$)

Salidas: Un arreglo de cadenas de caracteres ($C \in (\mathbb{ASCII}^*)^*$)

Relación:

$$\begin{array}{rcc} columnas : \mathbb{ASCII}^n & \to & \mathbb{R}^* \\ columnas(s) & \to & C \text{ tal que } C + [float(s_i)] \text{si } \forall_{i=0}^{n-1} s_i \neq \text{" "} \end{array}$$

$$\begin{array}{rcc} columnas : \mathbb{ASCII}^n & \to & (\mathbb{ASCII}^*)^* \\ columnas(s) & \to & C \text{ tal que } \forall_{i=0}^{n-1} C_{kj} = s_i \end{array}$$

Donde $k =$número de " "(espacios en blanco) desde la posición 0 hasta la posición i y $j = i-$la posición + 1 del último " " antes de la posición i o 0 si no había

Función que aplica la funcion columna y convierte a reales:

Entradas: Una lista de cadena de caractes de tamaño m ($A \in (\mathbb{ASCII}^*)^m$)

Salidas: Un arreglo de tamaño m de arreglos de tamaño k de reales ($D \in (\mathbb{R}^k)^m$)

Relación:

$$\begin{array}{rcc} matrizreales : \mathbb{ASCII}^* & \to & (\mathbb{ASCII}^*)^m \\ matrizreales(A) & \to & D \text{ tal que } (\forall_{i=0}^{m-1} (\forall_{j=0}^{k} D_{ij} = float(k_j))) \end{array}$$

$k = columnas(A_i)$ (es decir, los elementos de las columnas de la matriz) y $m =$cantidad de elementos del arreglo A

Función que crea un arreglo de reales a partir de la separación de columnas (arreglos):

Entradas: Un arreglo de cadenas de caracteres de tamaño n $(A \in (\mathbb{ASCII}^*)^n)$

Salidas: Un arreglo de reales $B \in \mathbb{R}^n$

Relación:

$arregloreal : (\mathbb{ASCII}^*)^n \to \mathbb{R}^n$

$arregloreal(A) \to B$ tal que $\forall_{i=0}^{n-1} B_i = float(A_i)$

Nota: En la codificación será sobre el mismo arreglo A que se harán las modificaciones

Función que crea el arreglo a partir de la cadena de texto:

Entradas: Una cadena de caracteres de tamaño $(s \in \mathbb{ASCII}^*)$

Salidas: Un arreglo de reales $A \in \mathbb{R}^n$

Relación:

$arreglo : \mathbb{ASCII}^* \to \mathbb{R}^n$

$arreglo(s) \to arregloreal(columnas(s))$

Funcion final que crea la matriz:

Entradas: **U**na cadena de caracteres $(s \in \mathbb{ASCII}^*)$

Salidas: Un arreglo de tamaño m de arreglos de reales de tamaño k $(A \in (\mathbb{R}^k)^m)$

Relación:

$$\begin{array}{rcl} matriz : \mathbb{ASCII}^* & \to & (\mathbb{R}^k)^m \\ matriz(s) & \to & matrizreales(filas(s)) \end{array}$$

Función que hace los intercambios de filas:

Entradas: Una matriz cuadrada de reales $(A \in \mathbb{R}^{n \times n})$, un arreglo de reales de tamaño n $(B \in \mathbb{R}^n)$, la posición de la fila donde el pivote es cero $(i \in \mathbb{N})$

Salidas: Una matriz cuadrada de reales $(A \in \mathbb{R}^{n \times n})$, un arreglo de reales de tamaño n $(B \in \mathbb{R}^n)$

Relación:

$inter : \mathbb{R}^{n \times n} \times \mathbb{R}^n \times \mathbb{N} \to \mathbb{R}^{n \times n} \times \mathbb{R}^n$

$inter(A, B, i) \to$ tal que $\exists_{j=i}^{n-1} A_{ji} \neq 0$ Entonces $A_i \longleftrightarrow A_j, B_i \longleftrightarrow B_j$

Función que reduce la fila y columna del pivote:

Entradas: Una matriz cuadrada de reales ($A \in \mathbb{R}^{n\times n}$), un arreglo de reales de tamaño n ($B \in \mathbb{R}^n$), posición de la fila y columna a reducir ($i \in \mathbb{N}$)

Salidas: Una matriz cuadrada de reales ($D \in \mathbb{R}^{n\times n}$), un arreglo de reales de tamaño n ($C \in \mathbb{R}^n$)

Relación:

$reduce : \mathbb{R}^{n\times n} \times \mathbb{R}^n \times \mathbb{N} \to \mathbb{R}^{n\times n} \times \mathbb{R}^n$

$reduce(A, B, i) \to D, C$ tal que $\forall_{j=0}^{m-1}(D_{ij} = \frac{A_{ij}}{A_{ii}} \wedge C_i = \frac{B_i}{A_{ii}}) \wedge \forall_{k=0}^{n-1}(D_{kj} = A_{kj} - (A_{ki} * A_{ij}) \wedge C_k = B_k - (A_{ki} * C_i))$ si $A_{ki} \neq 0 \wedge k \neq i$

Función que repite para todos los pivotes e intercambia:

Entradas: Una matriz cuadrada de reales ($A \in \mathbb{R}^{n\times n}$), un arreglo de reales de tamaño n ($B \in \mathbb{R}^n$)

Salidas: un arreglo de reales de tamaño n ($\mathbb{R}^n$)

Relación:

$Gauss : \mathbb{R}^{n\times n} \times \mathbb{R}^n \to \mathbb{R}^n$

$$Gauss(A, B) \to C \text{ tal que } \forall_{i=0}^{n-1} \begin{cases} A, C = reduce(inter(A, C, i), i) & si \quad A_{ii} = 0 \\ A, C = reduce(A, B, i) & \text{En otro caso} \end{cases}$$

Nota: *En la codificación, será al mismo arreglo B al que se le harán todas las modificaciones a lo largo de la función*

Especificación de forma recursiva

Entradas: Una matriz cuadrada de reales ($A \in \mathbb{R}^{n\times n}$), un arreglo de reales de tamaño n ($B \in \mathbb{R}^n$), filay columna a la que se le hará el proceso ($i \in \mathbb{N}$)

Salidas: Una matriz cuadrada de reales ($A \in \mathbb{R}^{n\times n}$) y un arreglo de reales de tamaño n ($B \in \mathbb{R}^n$)

Relación:

$integra : \mathbb{R}^{n\times n} \times \mathbb{R}^n \times \mathbb{N} \to \mathbb{R}^{n\times n} \times \mathbb{R}^n$

$$integra(A, B, i) \to \begin{cases} reduce(inter(A, B, i), i) & si \quad A_{ii} = 0 \\ reduce(A, B, i) & \text{En otro caso} \end{cases}$$

Entradas: Una matriz cuadrada de reales ($A \in \mathbb{R}^{n\times n}$), un arreglo de reales de tamaño n ($B \in \mathbb{R}^n$), filay columna a la que se le hará el proceso ($i \in \mathbb{N}$)

Salidas: un arreglo de reales de tamaño n ($B \in \mathbb{R}^n$)

Relación:

$$toda(A, B, i) \to \begin{cases} B & si \quad i < 1 \\ toda(integra(A, B, len(B) - i), i - 1) & \text{En otro caso} \end{cases}$$

Entradas: Una matriz cuadrada de reales ($A \in \mathbb{R}^{n\times n}$), un arreglo de reales de tamaño n ($B \in \mathbb{R}^n$)

Salidas: un arreglo de reales de tamaño n ($\mathbb{R}^n$)

Relación:

$Gauss2 : \mathbb{R}^{n\times n} \times \mathbb{R}^n \to \mathbb{R}^n$

$Gauss2(A, B) \to toda(A, B, |B|)$

3. Codificación:

El código en el que se resuelve el ejercicio se encuentra en el archivo llamado «Ejercicio_57_SimonRamos.py»

Ejercicio 58
Desarrollar un programa que calcule la inversa de una matriz cuadrada.
1. Análisis y especificación
1.1 Objetos conocidos:

- Se conoce una matriz cuadrada de reales $A \in \mathbb{R}^{n\times n}$

1.2 Objetos desconocidos:

- Se desconoce la matriz inversa de esta matriz dada

1.3 Relación entre los objetos:

- Al tener una matriz cuadrada $A^{n\times n}$, su inversa se puede calcular mediante Gauss-Jordan. En primer lugar se dispone una matriz extendida con la matriz identidad. La matriz A que es a la cual se desea calcular la inversa, se llevará a una matriz identidad mediante Gauss-jordan, mientras que todo el proceso se le hará de manera análoga también a la matriz identidad. Al final, la matriz A terminará siendo una matriz identidad, mientras que la matriz identidad dispuesta al principio se convertirá en la inversa de la matriz A.

2. Diseño y prueba conceptual
Funciones que hacen la lectura:

Función que separa por una "," y crea las filas:

Entradas: Una cadena de texto de longitud n ($s \in \mathbb{ASCII}^n$) con la representación de la matriz, esta cadena de texto debe venir de la forma "2 3 4, 5 6 7, 8 9 1"

Salidas: Un arreglo de cadenas de caracteres ($C \in (\mathbb{ASCII}^*)^*$)

Relación:

$$\begin{array}{lcc} filas : \mathbb{ASCII}^n & \to & (\mathbb{ASCII}^*)^* \\ filas(s) & \to & D \text{ tal que } \forall_{i=0}^{n-1} D_{kj} = s_i \end{array}$$

Donde k =número de "," desde la posición 0 hasta la posición i y $j = i-$la posición + 1 del último "," antes de la posición i o 0 si no había

Función que separa por columnas:

Entradas: Una cadena de caracteres de tamañon ($s \in \mathbb{ASCII}^n$)

Salidas: Un arreglo de cadenas de caracteres ($C \in (\mathbb{ASCII}^*)^*$)

Relación:

$$\begin{array}{rcc} columnas : \mathbb{ASCII}^n & \to & \mathbb{R}^* \\ columnas(s) & \to & C \text{ tal que } C + [float(s_i)] \text{si } \forall_{i=0}^{n-1} s_i \neq \text{" "} \end{array}$$

$$\begin{array}{rcc} columnas : \mathbb{ASCII}^n & \to & (\mathbb{ASCII}^*)^* \\ columnas(s) & \to & C \text{ tal que } \forall_{i=0}^{n-1} C_{kj} = s_i \end{array}$$

Donde k =número de " "(espacios en blanco) desde la posición 0 hasta la posición i y $j = i-$la posición + 1 del último " " antes de la posición i o 0 si no había

Función que aplica la funcion columna y convierte a reales:

Entradas: Una lista de cadena de caractes de tamaño m ($A \in (\mathbb{ASCII}^*)^m$)

Salidas: Un arreglo de tamaño m de arreglos de tamaño k de reales ($D \in (\mathbb{R}^k)^m$)

Relación:

$$\begin{array}{rcc} matrizreales : \mathbb{ASCII}^* & \to & (\mathbb{ASCII}^*)^m \\ matrizreales(A) & \to & D \text{ tal que } (\forall_{i=0}^{m-1} (\forall_{j=0}^{k} D_{ij} = float(k_j))) \end{array}$$

$k = columnas(A_i)$ (es decir, los elementos de las columnas de la matriz) y m =cantidad de elementos del arreglo A

Funcion final que crea la matriz:

Entradas: Una cadena de caracteres ($s \in \mathbb{ASCII}^*$)

Salidas: Un arreglo de tamaño m de arreglos de reales de tamaño k ($A \in (\mathbb{R}^k)^m$)

Relación:

$$\begin{array}{rcc} matriz : \mathbb{ASCII}^* & \to & (\mathbb{R}^k)^m \\ matriz(s) & \to & matrizreales(filas(s)) \end{array}$$

Función que hace la impresión del resultado:

Entradas: Un arreglo de arreglos (matriz) de enteros o reales $C \in \mathbb{R}^{n \times m}$

Salidas: Una cadena de caracteres $e \in \mathbb{ASCII}^*$

Relación: Se crea una cadena de caracteres que representa la matriz, en la cual se genera una nueva línea cada vez que se termina una fila de la matriz y todos los elementos de esta se separan por un espacio

$$\begin{array}{rcc} salida : \mathbb{R}^{n \times m} & \to & \mathbb{ASCII}^* \\ salida(C) & \to & e \text{ tal que } \forall_{i=0}^{n-1} (\forall_{j=0}^{m-1} e_{id} = str(C_{ij})) \end{array}$$

Donde $d = 0$ si $i = 0 \vee (2i + 1)$ En otro caso

Función que crea la matriz identidad:

Entradas: Una matriz cuadrada de reales $A \in \mathbb{R}^{n\times n}$

Salidas: Una matriz identidad de las mismas dimensiones que la matriz A ($B \in \{0,1\}^{n\times n}$)

Relación:

$identidad : \mathbb{R}^{n\times n} \to \{0,1\}^{n\times n}$

$identidad(A) \to B$ tal que $\forall_{j=0}^{n-1}(\forall_{i=0}^{n-1} \begin{cases} B_{ij} = 0 \quad si & i \neq j \\ B_{ij} = 1 & \text{En otro caso} \end{cases})$

Función que hace los intercambios de filas:

Entradas: Dos matrices cuadradas de reales ($A, B \in \mathbb{R}^{n\times n}$), la posición de la fila donde el pivote es cero ($i \in \mathbb{N}$)

Salidas: Dos matrices cuadradas de reales ($A, B \in \mathbb{R}^{n\times n}$),

Relación:

$inter : \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times n} \times \mathbb{N} \to \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times n}$

$inter(A, B, i) \to$ tal que $\exists_{j=i}^{n-1} A_{ji} \neq 0$ Entonces $A_i \longleftrightarrow A_j, B_i \longleftrightarrow B_j$

Función que reduce la fila y columna del pivote:

Entradas: Dos matrices cuadradas de reales ($A, B \in \mathbb{R}^{n\times n}$), posición de la fila y columna a reducir ($i \in \mathbb{N}$)

Salidas: Dos matrices cuadradas de reales ($D, C \in \mathbb{R}^{n\times n}$)

Relación:

$reduce : \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times n} \times \mathbb{N} \to \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times n}$

$reduce(A, B, i) \to D, C$ tal que $\forall_{j=0}^{m-1}(D_{ij} = \frac{A_{ij}}{A_{ii}} \wedge C_{ij} = \frac{B_{ij}}{A_{ii}}) \wedge \forall_{k=0}^{n-1}(D_{kj} = A_{kj} - (A_{ki} * A_{ij}) \wedge C_{kj} = B_{kj} - (A_{ki} * B_{ij}))$ si $A_{ki} \neq 0 \wedge k \neq i$

Nota: En la codificación, las modificaciones que llevan a obtener el arreglo D se harán sobre el arreglo A y las modificaciones que llevan a obtener el arreglo C se harán sobre el arreglo B

Función que comprueba si hay que intercambiar o no:

Entradas: Dos matrices cuadradas de reales ($A, B \in \mathbb{R}^{n\times n}$), fila y columna a la que se le hará el proceso ($i \in \mathbb{N}$)

Salidas: Dos matrices cuadradas de reales ($A, B \in \mathbb{R}^{n\times n}$)

Relación:

$integra : \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times n} \times \mathbb{N} \to \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times n}$

$$integra(A, B, i) \to \begin{cases} reduce(inter(A, B, i), i) \quad si & A_{ii} = 0 \\ reduce(A, B, i) & \text{En otro caso} \end{cases}$$

Funció que aplica las funciones anteriores para todas las filas y columnas:

Entradas: Dos matrices cuadradas de reales ($A, B \in \mathbb{R}^{n\times n}$), fila y columna a la que se le hará el proceso ($i \in \mathbb{N}$)

Salidas: Una matriz cuadrada de reales ($B \in \mathbb{R}^{n\times n}$)

Relación:

$$toda(A,B,i) \rightarrow \begin{cases} B & si \quad i < 1 \\ toda(integra(A,B,len(B)-i), i-1) & \text{En otro caso} \end{cases}$$

Función final que calcula la inversa de una amtriz:

Entradas: Una matriz cuadrada de reales ($A \in \mathbb{R}^{n\times n}$)

Salidas: Una matriz cuadrada de reales ($B \in \mathbb{R}^{n\times n}$)

Relación:

$inversa : \mathbb{R}^{n\times n} \rightarrow \mathbb{R}^{n\times n}$

$inversa(A,B) \rightarrow toda(A, indetidad(A), |B|)$

3. Codificación

El código en el que se resuelve el ejercicio se encuentra en el archivo llamado «Ejercicio_58_SimonRamos.py»

Ejercicio 59

Desarrollar un programa que tome un arreglo de tamaño n^2 y llene en espiral hacia adentro una matriz cuadrada de tamaño n.

1. Análisis y especificación

1.1 Objetos conocidos:

- Se conoce un arreglo de números reales de tamaño n^2, además, la matriz donde se colocaran los números es cuadrada y de orden $A^{n\times n}$

1.2 Objetos desconocidos:

- Se desconoce la matriz resultante al colocar los números del arreglo dado en forma de espiral dentro de ella

1.3 Relación entre los objetos:

- Al tener una matriz cuadrada $A^{n\times n}$ y un arreglo r^{n^2} en primer lugar hay que crear la matriz donde se dispondrán los números dle arrreglo, esta matriz en un principio se compondrá de ceros. Posteriormente, el problema se puede resolver de forma recursiva si llenamos primero el perimetro de la matriz cuadrada de tamaño n, después el perimetro de tamaño $n-2$ y así sucesivamente hasta que ya no queden más perimetros. Los números del arreglo que se utiilizaran para llenar estos bordes vendrán de los $4(n-1)$elementos del arreglo dado. Es decir, si tenemos una matriz

4×4 (el arreglo dado tiene 16 elementos) para el primer borde se utilizarán los primeros $4(n-1)$ elementos, osea los primeros 12 elementos. Para el siguiente borde, que será 2×2 , se utilizarán los siguientes $4(n-1)$ elementos, es decir los siguientes 4 elementos. Al terminar con todos los números del arreglo y ya no tener más bordes posibles por llenar se terminará con la función recursiva

2. Diseño y prueba conceptual
Funciones que hacen la lectura:
Al hacer la lectura del arreglo los elementos que hagan parte de él deben ir separados por un espacio, de esta forma la entrada "2 4 6 7 8", se corresponde con el arreglo [2,4,6,7,8]

Función que separa por espacios y convierte de texto a real:

Entradas: Una cadena de caracteres de tamañon ($s \in \mathbb{ASCII}^n$)

Salidas: Un arreglo de cadenas de caracteres ($C \in (\mathbb{ASCII}^*)^*$)

Relación: Esta función separa las cadenas de caracter ingresadas por medio de los espacios, de esta forma, a medida que aparece un espacio en la cadena los caracteres que había antes de este se convierten en un número real y se guardan en un arreglo. No obstante, si la hacer la separación por alguna razón queda una cadena vacia, esta no será tenida en cuenta. Finalmente, si tenemos una entrada por teclado del tipo "2.5 4.2 6 7 8", el arreglo final será [2.5,4.2,6,7,8]

$$\begin{aligned} arreglosreal : \mathbb{ASCII}^n &\rightarrow \mathbb{R}^* \\ arreglosreal(s) &\rightarrow C \text{ tal que } C + [float(s_i)] \text{si } \forall_{i=0}^{n-1} s_i \neq \text{" "} \wedge s_i \neq \text{""} \end{aligned}$$

$$\begin{aligned} arreglosreal : \mathbb{ASCII}^n &\rightarrow (\mathbb{ASCII}^*)^* \\ arreglosreal(s) &\rightarrow C \text{ tal que } \forall_{i=0}^{n-1} C_{kj} = float(s_i) \end{aligned}$$

Donde $k =$número de " "(espacios en blanco) desde la posición 0 hasta la posición i y $j = i -$la posición + 1 del último " " antes de la posición i o 0 si no había

Función que hace la impresión del resultado:

Entradas: Un arreglo de arreglos (matriz) de enteros o reales $C \in \mathbb{R}^{n\times m}$

Salidas: Una cadena de caracteres $e \in \mathbb{ASCII}^*$

Relación: Se crea una cadena de caracteres que representa la matriz, en la cual se genera una nueva línea cada vez que se termina una fila de la matriz y todos los elementos de esta se separan por un espacio

$$\begin{aligned} salida : \mathbb{R}^{n\times m} &\rightarrow \mathbb{ASCII}^* \\ salida(C) &\rightarrow e \text{ tal que } \forall_{i=0}^{n-1} (\forall_{j=0}^{m-1} e_{id} = str(C_{ij})) \end{aligned}$$

Donde $d = 0$ si $i = 0 \vee (2i + 1)$ En otro caso

Función que crea el arreglo con los elementos que se usarán para llenar el borde:

Entradas: 2 variables: 1 arreglo b de reales de tamaño n^2(es decir, la cantidad de elementos del arreglo tiene raíz cuadrada exacta) $b \in \mathbb{R}^{n^2}$, tamaño del borde que se va a llenar $k \in \mathbb{N}$

Salidas: Un arreglo d de reales de tamaño $4(k-1)$

Relación:

$arreglo : \mathbb{R}^{n^2} \times \mathbb{N} \to \mathbb{R}^*$

$arreglo(b,k) \to d$ tal que $\forall_{i=0}^{4(k-1)-1} d_i = b_{i+x}$

Donde $x =$ desde donde se deben tomar los elementos del arreglo b debido a la cantidad de veces que se ha repetido el ciclo hasta el momento, es decir $x = n^2 - k^2$

Función que llena los bordes:

Entradas: Matriz cuadrada A de reales ($A \in \mathbb{R}^{n\times n}$), arreglo de reales d de tamaño $m = 4(k-1)$ del cual se sacarán los números para llenar el recuadro de la matriz ($d \in \mathbb{R}^m$), tamaño del borde que se debe llenar ($k \in \mathbb{N}$), cantidad de veces que se ha ejecutado el ciclo ($j \in \mathbb{N}$)

Salidas: Una matriz cuadrada de reales $B \in \mathbb{R}^{n\times n}$

Relación:

$llena : \mathbb{R}^{n\times n} \times \mathbb{R}^m \times \mathbb{N} \times \mathbb{N} \to \mathbb{R}^{n\times n}$

$llena(A,d,k,j) \to B$ tal que $\forall_{i=0}^{k-1} B_{j.i+j} = d_i \wedge B_{i+j,x+j} = d_{x+i} \wedge B_{x+j,x-i+j} = d_{2x+i} \wedge B_{i+j,j} = d_{-i}$

Donde $x = (k-1)$

Nota: La razón por la cual se recibe como argumento una matriz de reales A es porque será sobe esta que se añadan los nuevos elementos tal y como lo especifica la función, esto quiere decir que la matriz B será construida sobre la matriz A. Adicionalmente, los elementos que serán modificados en la matriz A en un principio son 0, por lo que al final en la nueva matriz B esta contendrá en las posiciones especificadas números reales dintintos pero en las otras posiciones tendrá los mismos elementos que A

1^{00}	2^{01}	3^{02}	4^{03}	5^{04}
16^{10}	17^{11}	18^{12}	19^{13}	6^{14}
15^{20}	24^{21}	25^{22}	20^{23}	7^{24}
14^{30}	23^{31}	22^{32}	21^{33}	8^{34}
13^{40}	12^{41}	11^{42}	10^{43}	9^{44}

Función que integra:

Entradas: Una matriz cuadrada de reales ($A \in \mathbb{R}^{n\times n}$), un arreglo de reales cuya cantidad de elementos tiene raíz cuadrada finita ($b \in \mathbb{R}^{n^2}$) , cantidad de filas de la matriz (que se corresponde con la raíz cuadrada de la cantidad de elementos del arreglo, $n \in \mathbb{N}$), cantidad de veces que se ha repetido el ciclo ($j \in \mathbb{R}$)

Salidas: Una matriz cuadrada de reales $\mathbb{R}^{n\times n}$

Relación:

$integra : \mathbb{R}^{n\times n} \times \mathbb{R}^{n^2} \times \mathbb{N} \times \mathbb{N} \to \mathbb{R}^{n\times n}$

$$integra : (A, b, n, j) \to \begin{cases} A & si \quad n \leq 0 \\ A \text{ tal que } A_{j,j} = b_{-1} & si \quad n = 1 \\ integra(llena(A, arreglo(b, n), n, j), b, n-2, j+1)) & \text{En otro caso} \end{cases}$$

Función que llena e espiral usando las funciones anteriores:

Entradas: Un arreglo de reales cuya cantidad de elementos tiene raíz cuadrada exacta ($b \in \mathbb{R}^{n^2}$)

Salidas: Una matriz cuadrada de reales $C \in \mathbb{R}^{n\times n}$

Relación:

$espiral : \mathbb{R}^{n^2} \to \mathbb{R}^{n\times n}$

$espiral(b) \to integra(A, b, n, n-n)$

Nota: A es una matriz cuadrada de ceros de dimensiones n A es$\{0\}^{n\times n}$ (matriz nula). Adicionalmente, para sacar la raíz cuadrada de n se utiliza la función integrada de python para potencias, donde un número elevado a la $\frac{1}{2}$ es igual a la raíz cuadrada de este número

3. Codificación

El código en el que se resuelve el ejercicio se encuentra en el archivo llamado «Ejercicio_59_SimonRamos.py»

2 Relaciones binarias como matrices

Una matriz se puede usar para representar una relación entre dos conjuntos A y B. Esta representación es como sigue: Si $A = \{x_0, x_1, x_2, \ldots, x_{n-1}\}$ y $B = \{y_0, y_1, y_2, \ldots y_{m-1}\}$ una relación R de A en B se representa mediante una matriz de unos y ceros de tamaño n×m, donde $A_{ij} = 1$ si el elemento x_i se relaciona con el elemento y_i, en caso contrario $A_{ij} = 0$
Usando esta representación hacer un programa que le permita al usuario leer dos relaciones entre dos conjuntos y escoger mediante un menú, una de las siguientes operaciones sobre dichas relaciones:
Funciones que hacen la lectura:

Función que separa por una "," y crea las filas:

Entradas: Una cadena de texto de longitud n ($s \in \mathbb{ASCII}^n$) con la representación de la matriz, esta cadena de texto debe venir de la forma "2 3 4, 5 6 7, 8 9 1"

Salidas: Un arreglo de cadenas de caracteres ($C \in (\mathbb{ASCII}^*)^*$)

Relación:

$$\begin{array}{ccc} filas : \mathbb{ASCII}^n & \to & (\mathbb{ASCII}^*)^* \\ filas(s) & \to & D \text{ tal que } \forall_{i=0}^{n-1} D_{kj} = s_i \end{array}$$

Donde $k =$número de "," desde la posición 0 hasta la posición i y $j = i-$la posición + 1 del último "," antes de la posición i o 0 si no había

Función que separa por columnas:

Entradas: Una cadena de caracteres de tamañon ($s \in \mathbb{ASCII}^n$)

Salidas: Un arreglo de cadenas de caracteres ($C \in (\mathbb{ASCII}^*)^*$)

Relación:

$$\begin{array}{ccc} columnas : \mathbb{ASCII}^n & \to & \mathbb{R}^* \\ columnas(s) & \to & C \text{ tal que } C + [float(s_i)] \text{si } \forall_{i=0}^{n-1} s_i \neq \text{" "} \end{array}$$

$$\begin{array}{ccc} columnas : \mathbb{ASCII}^n & \to & (\mathbb{ASCII}^*)^* \\ columnas(s) & \to & C \text{ tal que } \forall_{i=0}^{n-1} C_{kj} = s_i \end{array}$$

Donde $k =$número de " "(espacios en blanco) desde la posición 0 hasta la posición i y $j = i-$la posición + 1 del último " " antes de la posición i o 0 si no había

Función que aplica la funcion columna y convierte a reales:

Entradas: Una lista de cadena de caractes de tamaño m ($A \in (\mathbb{ASCII}^*)^m$)

Salidas: Un arreglo de tamaño m de arreglos de tamaño k de reales ($D \in (\mathbb{R}^k)^m$)

Relación:

$$\begin{array}{ccc} matrizbinaria : \mathbb{ASCII}^* & \to & (\mathbb{ASCII}^*)^m \\ matrizrbinaria(A) & \to & D \text{ tal que } (\forall_{i=0}^{m-1} (\forall_{j=0}^{k} D_{ij} = int(k_j))) \end{array}$$

$k = columnas(A_i)$ (es decir, los elementos de las columnas de la matriz) y $m =$cantidad de elementos del arreglo A

Funcion final que crea la matriz:

Entradas: **U**na cadena de caracteres ($s \in \mathbb{ASCII}^*$)

Salidas: Un arreglo de tamaño m de arreglos de reales de tamaño k ($A \in (\mathbb{R}^k)^m$)

Relación:

$$\begin{array}{ccc} matriz : \mathbb{ASCII}^* & \to & (\mathbb{R}^k)^m \\ matriz(s) & \to & matrizbinaria(filas(s)) \end{array}$$

Función que hace la impresión del resultado:

Entradas: Un arreglo de arreglos (matriz) de enteros o reales $C \in \mathbb{R}^{n\times m}$

Salidas: Una cadena de caracteres $e \in \mathbb{ASCII}^*$

Relación: Se crea una cadena de caracteres que representa la matriz, en la cual se genera una nueva línea cada vez que se termina una fila de la matriz y todos los elementos de esta se separan por un espacio

$$\begin{array}{rcl} salida : \mathbb{R}^{n\times m} & \rightarrow & \mathbb{ASCII}^* \\ salida(C) & \rightarrow & e \text{ tal que } \forall_{i=0}^{n-1}(\forall_{j=0}^{m-1} e_{id} = str(C_{ij})) \end{array}$$

Donde $d = 0$ si $i = 0 \vee (2i + 1)$ En otro caso

3, Codificación

El código en el que se encuentran estas funciones esta disponible en el archivo llamado «Ejercicios_BINARIOS_SimonRamos.py». De la línea de código 1 a la 56

Ejercicio 60
UNION: Calcula e imprime la relación unión.
1. Análisis y especificación
1.1 Objetos conocidos:

- Se conocen dos relaciones binarias entre conjuntos, las cuales pueden verse como dos matrices de unos y ceros. Las dos matrices tienen las mismas dimensiones $A, B \in \{0, 1\}^{n\times m}$

1.2 Objetos desconocidos:

- Se desconoce la matriz resultante de la unión entre las matrices binarias

1.3 Relación entre los objetos:

- En general, la operación unión puede verse como un *or* $(\vee)$. Si analizamos dos matrices binarias, si alguna de las dos tiene un 1 en la misma posición, entonces la matriz resultante unión tendrá un 1 en esa posición. La única forma en la que en la matriz resultante se tenga un cero es que en ambas matrices dadas exista un cero en la misma posición. Esta interpretación puede llevarse a la codificación desde un enfoque algo distinto: si tenemos las dos matrices binarias A y B, y la matriz unión no la construiremos en una nueva matriz sino en la propia matriz A, entonces el elementoA_{ij} será igual a 1 si $A_{ij} \neq B_{ij}$, en caso contrario A_{ij} se mantendrá sin modificaciones. Esto se sustenta en que si los dos elementos en la misma posición son 1, entonces no es necesario cambiar el elemento en la matriz A pues ya contiene en sí misma la unión, los mismo sucede si hay un 0. Por otro lado, si alguno es 0 y el otro 1, entonces en la matriz unión el elemento de esa unión deberá ser 1 .

2. Diseño y prueba conceptual
Función que calcula la unión:

Entradas: Dos matrices binarias con las mismas dimensiones ($A, B \in \{0,1\}^{n \times m}$)

Salidas: Una matriz binaria resultante de la unión entre A y B ($A \cup B$) con las mismas dimensiones que las matrices dadas ($U \in \{0,1\}^{n \times m}$)

Relación:

$union : \{0,1\}^{n\times m} \times \{0,1\}^{n\times m} \rightarrow \{0,1\}^{n\times m}$

$$union(A,B) \rightarrow U \text{ tal que } \forall_{i=0}^{n-1} (\forall_{j=0}^{m-1} \begin{cases} U_{ij} = 1 \quad si & A_{ij} = 1 \vee B_{ij} = 1 \\ U_{ij} = 0 & \text{En otro caso} \end{cases})$$

=============================

$union2 : \{0,1\}^{n\times m} \times \{0,1\}^{n\times m} \rightarrow \{0,1\}^{n\times m}$

$$union2(A,B) \rightarrow U \text{ tal que } \forall_{i=0}^{n-1} (\forall_{j=0}^{m-1} \begin{cases} U_{ij} = 1 \quad si & A_{ij} \neq B_{ij} \\ U_{ij} = A_{ij} & \text{En otro caso} \end{cases})$$

Nota: En la codificación, las modificaciones sobre el mismo arreglo A, por lo que $U = A$ en un comienzo

3, Codificación

El código en el que se encuentran estas funciones esta disponible en el archivo llamado «Ejercicios_BINARIOS_SimonRamos.py». De la línea de código 59 a la 76, y en la función principal estas se llaman de la línea de código 249 a la 253

Ejercicio 61
INTERSECCIÓN: Calcula e imprime la relación intersección
1. Análisis y especificación
1.1 Objetos conocidos:

- Se conocen dos relaciones binarias entre conjuntos, las cuales pueden verse como dos matrices de unos y ceros. Las dos matrices tienen las mismas dimensiones $A, B \in \{0,1\}^{n\times m}$

1.2 Objetos desconocidos:

- Se desconoce la matriz resultante de la intersección entre las matrices binarias

1.3 Relación entre los objetos:

- En general, la operación intersección puede verse como un *and* ($\wedge$). Si analizamos dos matrices binarias, si las dos tiene un 1 en la misma posición, entonces la matriz resultante intersección tendrá un 1 en esa posición, en caso contrario tendrá un 0. Esta interpretación de $A \cap B$ puede verse como una multiplicación elemento por elemento de cada matriz. Así, la única forma en que la matriz resultante tendría un 1 es cuando ambas matrices tienen un 1 en esa posición, en caso contrario daría 0 ya que cualquier número multiplicado por 0 da 0. Se estaría simulando la puerta lógica AND. Esto puede codificarse sobre la misma matriz A

2. Diseño y prueba conceptual
Función que hace la intersección de matrices binarias:

Entradas: Dos matrices binarias con las mismas dimensiones ($A, B \in \{0,1\}^{n\times m}$)

Salidas: Una matriz binaria resultante de la intersección entre A y B ($A \cap B$) con las mismas dimensiones que las matrices dadas ($N \in \{0,1\}^{n\times m}$)

Relación:

$intersección : \{0,1\}^{n\times m} \times \{0,1\}^{n\times m} \to \{0,1\}^{n\times m}$

$interseccion(A, B) \to N$ tal que $\forall_{i=0}^{n-1}(\forall_{j=0}^{m-1} N_{ij} = A_{ij} * B_{ij})$

Nota: En la codificación las modificaciones se harán sobre la matriz A

3, Codificación

El código en el que se encuentran estas funciones esta disponible en el archivo llamado «Ejercicios_BINARIOS_SimonRamos.py». De la línea de código 79 a la 86, y en la función principal estas se llaman de la línea de código 255 a la 260

Ejercicio 62
SIMETRIA: Determina si la primer relación es simétrica o no.
1. Análisis y especificación
1.1 Objetos conocidos:

- Se conoce una relación binaria entre conjuntos, la cual se representa mediante una matriz de unos y ceros. Para poder probar si una relación binaria (matriz) es simétrica o no, la relación debe ir sobre el mismo conjunto, es decir, de A en A. Esto implica que la matriz binaria debe ser cuadrada, en caso contrario, no se podrá porbar simetría

1.2 Objetos desconocidos:

- Se desconoce si la matriz dada es simétrica o no

1.3 Relación entre los objetos:

- La simetría se prueba sobre una relación que va de un conjunto A hacia el mismo conjunto A, por lo que la matriz resultante que representa esta relación binaria debe ser cuadrada. La simetría implica que si existe la pareja ordenada (x, y) en la relación, entonces existe la pareja (y, x). En términos de la matriz A, si $A_{ij} = 1$ entonces $A_{ji} = 1$. En un análisis más exhaustivo, se puede observar que si la pareja ordenada (x, y) no existe (en la matriz hay un cero en esa posición) entonces la pareja (y, x) tampoco debe existir (debe haber un cero en esa posicioón de la matriz). Caso contrario, si la pareja ordenada (x, y) existe entonces (y, x) también debe hacerlo (En ambas posiciones de la matriz debe haber un 1). Así que para comporbar la propiedad de simetría en la matriz binaria $A_{ij} = A_{ji}$

2. Diseño y prueba conceptual
Función que determina si la relación es simétrica:

Entradas: Una matriz binaria cuadrada que representa una relación binaria de un conjunto A hacia el mismo conjunto A $(A \in \{0, 1\}^{n \times n})$

Salidas: Un valor de verdad (True si la matriz (relación) es simétrica, False si no lo es) $v \in \mathbb{B}$

Relación:

$simetrica : \{0, 1\}^{n \times n} \to \mathbb{B}$

$$simetrica(A) \to \left\{ \begin{array}{lcc} True & si & \forall_{i=0}^{n-1}(\forall_{j=0}^{n-1} \exists A_{ij} = 1 \Rightarrow A_{ji} = 1) \\ False & & \text{En otro caso} \end{array} \right|$$

==

$simetrica2 : \{0, 1\}^{n \times n} \to \mathbb{B}$

$$simetrica2(A) \to \left\{ \begin{array}{lcc} True & si & \forall_{i=0}^{n-1}(\forall_{j=0}^{n-1} A_{ij} = A_{ji}) \\ False & & \text{En otro caso} \end{array} \right|$$

3, Codificación

El código en el que se encuentran estas funciones esta disponible en el archivo llamado «Ejercicios_BINARIOS_SimonRamos.py». De la línea de código 88 a la 113, y en la función principal estas se llaman de la línea de código 261 a la 263

Ejercicio 63
REFLEXIVIDAD: Determina si la primer relación es reflexiva o no.
1. Análisis y especificación
1.1 Objetos conocidos:

- Se conoce una relación binaria entre conjuntos, la cual se representa mediante una matriz de unos y ceros. Para poder probar si una relación binaria (matriz) es reflexiva o no, la relación debe ir sobre el mismo conjunto, es decir, de A en A. Esto implica que la matriz binaria debe ser cuadrada, en caso contrario, no se podrá probar reflexividad

1.2 Objetos desconocidos:

- Se desconoce si la matriz dada es reflexiva o no

1.3 Relación entre los objetos:

- La reflexividad se prueba sobre una relación que va de un conjunto A hacia el mismo conjunto A, por lo que la matriz resultante que representa esta relación binaria debe ser cuadrada. La reflexividad implica que todo elemento de la relación esta relacionado consigo mismo, es decir, si lo vemos como parejas ordenadas una relación reflexiva contiene a la pareja (x, x). En una representación matricial que un elemento se relacione consigo mismo llevaría a que en la diagonal de esta matriz se ponga un 1, es decir, $A_{ii} = 1$. Para comporbar la reflexividad todos los elementos de la diagonal de la matriz deben ser 1, pues todos los elementos de la relación se estarían relacionando consigo mismo.

2. Diseño y prueba conceptual
Función que determina si la relación es reflexiva:

Entradas: Una matriz binaria cuadrada que representa una relación binaria de un conjunto A hacia el mismo conjunto A $(A \in \{0,1\}^{n\times n})$

Salidas: Un valor de verdad (True si la matriz (relación) es reflexiva, False si no lo es) $v \in \mathbb{B}$

Relación:

$reflexiva : \{0,1\}^{n\times n} \to \mathbb{B}$

$$reflexvia(A) \to \left\{ \begin{array}{lll} True & si & \forall_{i=0}^{n-1} A_{ii} = 1 \\ False & & \text{En otro caso} \end{array} \right|$$

3, Codificación

El código en el que se encuentran estas funciones esta disponible en el archivo llamado «Ejercicios_BINARIOS_SimonRamos.py». De la línea de código 116 a la 125, y en la función principal estas se llaman de la línea de código 264 a la 266

Ejercicio 64
TRANSITIVIDAD: Determina si la primer relación leída es transitiva o no.
1. Análisis y especificación
1.1 Objetos conocidos:

- Se conoce una relación binaria entre conjuntos, la cual se representa mediante una matriz de unos y ceros. Para poder probar si una relación binaria (matriz) es transitiva o no, la relación debe ir sobre el mismo conjunto, es decir, de A en A. Esto implica que la matriz binaria debe ser cuadrada, en caso contrario, no se podrá probar transitividad

1.2 Objetos desconocidos:

- Se desconoce si la matriz dada es transitiva o no

1.3 Relación entre los objetos:

- La transitividad se prueba sobre una relación que va de un conjunto A hacia el mismo conjunto A, por lo que la matriz resultante que representa esta relación binaria debe ser cuadrada. La transitividad implica que si un elemento x se relaciona con un elemento y (visto como parejas ordenadas (x, y)) y el elemento y se relaciona con el elemento z (y, z), entonces x se relaciona con z (x, z). Desde el punto de vista de la relación binaria, si en la matriz binaria el elemento $A_{ij} = 1$ y $A_{jk} = 1$ entonces $A_{ik} = 1$, es decir, $A_{ij} = 1 \wedge A_{jk} = 1 \rightarrow A_{ik} = 1$. No obstante, lo anterior puede simplificarse según: $\alpha \rightarrow \beta \Leftrightarrow \neg\alpha \vee \beta$, por lo que tendríamos $\neg(A_{ij} = 1 \wedge A_{jk} = 1) \vee A_{ik} = 1$. Del mismo modo, si $\neg(\alpha \wedge \beta) \Leftrightarrow \neg\alpha \vee \neg\beta$ tendríamos a $\neg A_{ij} = 1 \vee \neg A_{jk} = 1$. FInalmente, tendríamos que la transitividad se da si $A_{ij} \neq 1 \vee A_{jk} \neq 1 \vee A_{ik} = 1$

2. Diseño y prueba conceptual
Función que determina si la relación es transitiva:

Entradas: Una matriz binaria cuadrada que representa una relación binaria de un conjunto A hacia el mismo conjunto A $(A \in \{0, 1\}^{n \times n})$

Salidas: Un valor de verdad (True si la matriz (relación) es transitiva, False si no lo es) $v \in \mathbb{B}$

Relación:

$transitiva : \{0, 1\}^{n \times n} \rightarrow \mathbb{B}$

$$transitiva(A) \rightarrow \left\{ \begin{array}{lll} True & si & \forall_{i=0}^{n-1}(\forall_{j=0}^{n-1}(\forall_{k=0}^{n-1}A_{ij} \neq 1 \vee A_{jk} \neq 1 \vee A_{ik} = 1)) \\ False & & \text{En otro caso} \end{array} \right|$$

3, Codificación

El código en el que se encuentran estas funciones esta disponible en el archivo llamado «Ejercicios_BINARIOS_SimonRamos.py». De la línea de código 128 a la 144, y en la función principal estas se llaman de la línea de código 267 a la 269

Ejercicio 65
ORDEN: Determina si la primer relación leída es relación de orden o no.
1. Análisis y especificación
1.1 Objetos conocidos:

- Se conoce una relación binaria entre conjuntos, la cual se representa mediante una matriz de unos y ceros. Para poder probar si una relación binaria (matriz) es de orden o no, la relación debe ir sobre el mismo conjunto, es decir, de A en A. Esto implica que la matriz binaria debe ser cuadrada, en caso contrario, no se podrá probar si es de orden. Hay tres tipos de orden: Orden parical, total o estricto

1.2 Objetos desconocidos:

- Se desconoce si la matriz dada es de orden o no

1.3 Relación entre los objetos:

- Hay tres tipos de orden: Parcial, total o estricto. El orden parcial se da si la matriz es reflexiva, transitiva y antisimetrica, las dos primeras funciones fueron definidas pero la última no. Una relación es antisimétrica si cumple que si el elemento x se relaciona con el elemento y (x, y) y el elemento y se relaciona con el x (y, x), entonces $x = y$. En las matrices binarias esto solo se cumple en las diagonales de la matriz. Así que para comprobar la antisimetría se debe tener que si x se relaciona con y (x, y) entonces y no se puede relacionar con x (y, x). Si la primera relación existe (tiene un 1) la segunda no debe existir (debe tener un 0), y visceversa, excepto si se trata de la diagonal de la matriz: $A_{ij} = 1 \rightarrow A_{ji} = 0$ excepto si $i = j$. Lo anterior se puede tranformar según $\alpha \rightarrow \beta \Leftrightarrow \neg\alpha \vee \beta$, así que $A_{ij} = 0 \vee A_{ji} = 0$ siempre que $i \neq j$

- La relación es de orden total si es parcial (es reflexiva, transitiva y antisimétrica) y además se cumple que x esta relacionado con y **ó** y está relacionado con x $((x, y) \vee (y, x))$. Lo anterior se cumple si ninguna de las dos es cero en la matriz al mismo tiempo, es decir si $A_{ij} = 0 \rightarrow A_{ji} = 1$. Lo podemos tranformar según $\alpha \rightarrow \beta \Leftrightarrow \neg\alpha \vee \beta$, por lo que $A_{ij} = 1 \vee A_{ji} = 1$

- Una relación de orden estricto se da si es antireflexiva (x no se relaciona consigo mismo), asimétrica (si x se relaciona con y entonces y no se relaciona con x) y transitiva

2. Diseño y prueba conceptual

Función que determina si la relación es antisimetrica:

Entradas: Una matriz binaria cuadrada que representa una relación binaria de un conjunto A hacia el mismo conjunto A $(A \in \{0, 1\}^{n \times n})$

Salidas: Un valor de verdad (True si la matriz (relación) es antisimetrica, False si no lo es) $v \in \mathbb{B}$

Relación:

$antisim : \{0, 1\}^{n \times n} \rightarrow \mathbb{B}$

$$antisim(A) \rightarrow \left\{ \begin{array}{lll} True & si & \forall_{i=0}^{n-1} (\forall_{j=0}^{n-1} A_{ij} = 0 \vee A_{ji} = 0) \text{ si } i \neq j \\ False & & \text{En otro caso} \end{array} \right|$$

Función que comprueba la condición extra de total:

Entradas: Una matriz binaria cuadrada que representa una relación binaria de un conjunto A hacia el mismo conjunto A $(A \in \{0, 1\}^{n \times n})$

Salidas: Un valor de verdad (True si la matriz (relación) cumple el requisito extra para ser total, False si no lo es) $v \in \mathbb{B}$

Relación:

$extra : \{0,1\}^{n\times n} \to \mathbb{B}$

$$extra(A) \to \begin{cases} True & si \quad \forall_{i=0}^{n-1}(\forall_{j=0}^{n-1} A_{ij} = 1 \vee A_{ji} = 1) \\ False & \text{En otro caso} \end{cases}$$

Función de orden parcial:

Entradas: Una matriz binaria cuadrada que representa una relación binaria de un conjunto A hacia el mismo conjunto A ($A \in \{0,1\}^{n\times n}$)

Salidas: Un valor de verdad (True si la matriz (relación) es de orden parcial, False si no lo es) $v \in \mathbb{B}$

Relación: El orden parcial se da si la matriz es reflexiva, transitiva y antisimetrica, ya que tenemos funciones que calculan estas propiedades simplementa comprobamos si cumple las tres. Esto se hace mediante un and, ya que si alguna llega a ser False (no tiene esta propiedad) lo relación no sería de orden parcial

$parcial : \{0,1\}^{n\times n} \to \mathbb{B}$

$parcial(A) \to reflexiva(A) \wedge transitiva(A) \wedge antisim(A)$

Función de orden total:

Entradas: Una matriz binaria cuadrada que representa una relación binaria de un conjunto A hacia el mismo conjunto A ($A \in \{0,1\}^{n\times n}$)

Salidas: Un valor de verdad (True si la matriz (relación) es de orden total, False si no lo es) $v \in \mathbb{B}$

Relación: La relación es de orden total si es parcial (es reflexiva, transitiva y antisimétrica) y además se cumple la condición *extra* que se definió anteriormente. Esta comprobación se hace mdiante un and ya que si no cumple ninguna de las dos condiciones la relación no es de orden total

$total : \{0,1\}^{n\times n} \to \mathbb{B}$

$total(A) \to parcial(A) \wedge extra(A)$

Función de orden estricto:

Entradas: Una matriz binaria cuadrada que representa una relación binaria de un conjunto A hacia el mismo conjunto A ($A \in \{0,1\}^{n\times n}$)

Salidas: Un valor de verdad (True si la matriz (relación) es de orden total, False si no lo es) $v \in \mathbb{B}$

Relación: Una relación de orden estricto se da si es antireflexiva, asimétrica y transitiva. Ya que tenemos dos funciones que definen si la relación es reflexiva o simétrica, las propiedades contrarias (antireflexiva y asimétrica) pueden verse como la negación de estas funciones definidas previamente. Ya que, por ejemplo, si una relación es simétrica no es asimétrica, y si es asimétrica entonces no es simétrica.

$estricto : \{0,1\}^{n\times n} \to \mathbb{B}$

$estricto(A) \to \neg reflexiva(A) \wedge \neg simetrica(A) \wedge transitiva(A)$

3, Codificación

El código en el que se encuentran estas funciones esta disponible en el archivo llamado «Ejercicios_BINARIOS_SimonRamos.py». De la línea de código 147 a la 180, y en la función principal estas se llaman de la línea de código 270 a la 276

Ejercicio 66

EQUIVALENCIA: Determina si la primer relación leída es una relación de equivalencia.

1. Análisis y especificación

1.1 Objetos conocidos:

- Se conoce una relación binaria entre conjuntos, la cual se representa mediante una matriz de unos y ceros. Para poder probar si una relación binaria (matriz) es de equivalencia o no, la relación debe ir sobre el mismo conjunto, es decir, de A en A. Esto implica que la matriz binaria debe ser cuadrada, en caso contrario, no se podrá probar si es una relación de equivalencia.

1.2 Objetos desconocidos:

- Se desconoce si la matriz dada es de equivalencia o no

1.3 Relación entre los objetos:

- Para que una relación sea de equivalencia debe cumplir con tres propiedades: Reflexividad, simetría y transitividad. Las anteriores funciones ya fueron especificadas anteriormente, por lo que solo es necesario comporbar si las tres son verdaderas para la relación dada. La reflexividad implica que todo elemento de la relación esta relacionado consigo mismo, es decir, si lo vemos como parejas ordenadas una relación reflexiva contiene a la pareja (x, x). Esto se ve en la matriz como que en su diagonal solo hay unos. La simetría implica que si existe la pareja ordenada (x, y) en la relación, entonces existe la pareja (y, x). En términos de la matriz A, si $A_{ij} = 1$ entonces $A_{ji} = 1$. Por último, la transitividad implica que si un elemento x se relaciona con un elemento y (visto como parejas ordenadas (x, y)) y el elemento y se relaciona con el elemento z (y, z), entonces x se relaciona con z (x, z).

2. Diseño y prueba conceptual
Función que determina si la relación es reflexiva:

Entradas: Una matriz binaria cuadrada que representa una relación binaria de un conjunto A hacia el mismo conjunto A ($A \in \{0,1\}^{n \times n}$)

Salidas: Un valor de verdad (True si la matriz (relación) es reflexiva, False si no lo es) $v \in \mathbb{B}$

Relación:

$reflexiva : \{0,1\}^{n \times n} \to \mathbb{B}$

$$reflexvia(A) \to \left\{ \begin{array}{lll} True & si & \forall_{i=0}^{n-1} A_{ii} = 1 \\ False & & \text{En otro caso} \end{array} \right|$$

Función que determina si la relación es simétrica:

Entradas: Una matriz binaria cuadrada que representa una relación binaria de un conjunto A hacia el mismo conjunto A ($A \in \{0,1\}^{n \times n}$)

Salidas: Un valor de verdad (True si la matriz (relación) es simétrica, False si no lo es) $v \in \mathbb{B}$

Relación:

$simetrica : \{0,1\}^{n \times n} \to \mathbb{B}$

$$simetrica(A) \to \left\{ \begin{array}{lll} True & si & \forall_{i=0}^{n-1} (\forall_{j=0}^{n-1} \exists A_{ij} = 1 \Rightarrow A_{ji} = 1) \\ False & & \text{En otro caso} \end{array} \right|$$

==

$simetrica2 : \{0,1\}^{n \times n} \to \mathbb{B}$

$$simetrica2(A) \to \left\{ \begin{array}{lll} True & si & \forall_{i=0}^{n-1} (\forall_{j=0}^{n-1} A_{ij} = A_{ji}) \\ False & & \text{En otro caso} \end{array} \right|$$

Función que determina si la relación es transitiva:

Entradas: Una matriz binaria cuadrada que representa una relación binaria de un conjunto A hacia el mismo conjunto A ($A \in \{0,1\}^{n \times n}$)

Salidas: Un valor de verdad (True si la matriz (relación) es transitiva, False si no lo es) $v \in \mathbb{B}$

Relación:

$transitiva : \{0,1\}^{n \times n} \to \mathbb{B}$

$$transitiva(A) \to \left\{ \begin{array}{lll} True & si & \forall_{i=0}^{n-1} (\forall_{j=0}^{n-1} (\forall_{k=0}^{n-1} A_{ij} \neq 1 \lor A_{jk} \neq 1 \lor A_{ik} = 1)) \\ False & & \text{En otro caso} \end{array} \right|$$

Función que determina si es de equivalencia:

Entradas: Una matriz binaria cuadrada que representa una relación binaria de un conjunto A hacia el mismo conjunto A ($A \in \{0,1\}^{n \times n}$)

Salidas: Un valor de verdad (True si la matriz (relación) es de equivalencia, False si no lo es) $v \in \mathbb{B}$

Relación:

$equivalencia : \{0,1\}^{n \times n} \rightarrow \mathbb{B}$

$equivalencia(A) \rightarrow reflexiva(A) \wedge simetrica(A) \wedge transitiva(A)$

3, Codificación

El código en el que se encuentran estas funciones esta disponible en el archivo llamado «Ejercicios_BINARIOS_SimonRamos.py». De la línea de código 181 a la 185, y en la función principal estas se llaman de la línea de código 277 a la 279

Ejercicio 67
FUNCIÓN: Determina si la relación es una función o no.
1.1 Objetos conocidos:

- Se conoce una relación binaria entre conjuntos, la cual se representa mediante una matriz de unos y ceros. En este caso, la relación se puede dar entre dos conjuntos distintos, por lo que no necesariamente la matriz será cuadrada

1.2 Objetos desconocidos:

- Se desconoce si la matriz dada representa una función o no

1.3 Relación entre los objetos:

- Una relación es función si cada elemento del conjunto de salida (dominio)A se relaciona con un único elemento de un conjunto de llegada B (codominio). No obstante, no necesariamente a todos los elementos de A le corresponde un elemento en B, y puede ser que un elemento de B se relaciona con más de un elemento de A. Si lo vemos en la representación matricial, que una relación sea función implicaría en cada fila de la matriz puede haber como máximo solo un 1.

- Para calcular si una matriz binaria representa a una función podríamos sumar cada fila de la matriz, y ya que debe haber como máximo un 1 en esta fila, la suma de sus componentes debe ser ≤ 1

2. Diseño y prueba conceptual
Función que hace la suma de la fila

Entradas: 2 variables: Una matriz $A \in \mathbb{R}^{n \times m}$ y el índice de la fila que se desa sumar $(k : k \in \mathbb{N} \wedge \in [0, n-1])$ $(k \in \mathbb{N}^{[0,n-1]})$

Salidas: 1 variable: La suma de los elementos de la fila ($suma \in \mathbb{R}$)

Relación:

$sumafil : \{0,1\}^{n\times m} \times \mathbb{N}^{[0,n-1]} \rightarrow \mathbb{R}$

$sumafil(A,k) \rightarrow \sum_{j=0}^{m-1} A_{k,j}$

Función que evalúa si la suma es menor o igual a 1:

Entradas: Una matriz $A \in \mathbb{R}^{n\times m}$

Salidas: 1 valor de verdad (True si la relación es función (la suma de todos los componentes de cada fila es ≤ 1) o False en otro caso) $v \in \mathbb{B}$

Relación:

$funcion : \{0,1\}^{n\times m} \rightarrow \mathbb{B}$

$$funcion(A) \rightarrow \left\{ \begin{array}{lll} True & si & \forall_{i=0}^{n-1} sumafil(A,i) \leq 1 \\ False & & \text{En otro caso} \end{array} \right|$$

3, Codificación

El código en el que se encuentran estas funciones esta disponible en el archivo llamado «Ejercicios_BINARIOS_SimonRamos.py». De la línea de código 188 a la 201, y en la función principal estas se llaman de la línea de código 280 a la 284

Ejercicio 68:
INYECTIVIDAD: Determina si la relació es una función inyectiva.
1.1 Objetos conocidos:

- Se conoce una relación binaria entre conjuntos, la cual se representa mediante una matriz de unos y ceros. En este caso, la relación se puede dar entre dos conjuntos distintos, por lo que no necesariamente la matriz será cuadrada

1.2 Objetos desconocidos:

- Se desconoce si la matriz dada representa una función inyectiva o no

1.3 Relación entre los objetos:

- Para comporbar inyectividad, en primer lugar se debe comporbar si se trata de una función. Una relación binaria es función si cada elemento del conjunto de salida (dominio)A se relaciona con un único elemento de un conjunto de llegada B (codominio). No obstante, no necesariamente a todos los elementos de A le corresponde un elemento en B, y puede ser que un elemento de B se relaciona con más de un elemento de A. Si lo vemos en la representación matricial, que una relación sea función implicaría en cada fila de la matriz puede haber como máximo solo un 1.

- SIn embargo, al querer comprobar inyectividad el último caso no se puede dar. En una función inyectiva cada elemento de B está relacionado solo con un elemento de A. Para calcular si una matriz binaria representa a una función podríamos sumar cada fila de la matriz, y ya que debe haber como máximo un 1 en esta fila, la suma de sus componentes debe ser ≤ 1. Del mismo modo, para comprobar si es inyectiva, la suma de los elementos de cada columna debe ser ≤ 1

2. Diseño y prueba conceptual
Función que hace la suma de la fila

Entradas: 2 variables: Una matriz $A \in \mathbb{R}^{n\times m}$ y el índice de la fila que se desa sumar $(k : k \in \mathbb{N} \wedge \in [0, n-1])$ $(k \in \mathbb{N}^{[0,n-1]})$

Salidas: 1 variable: La suma de los elementos de la fila $(suma \in \mathbb{R})$

Relación:

$sumafil : \{0,1\}^{n\times m} \times \mathbb{N}^{[0,n-1]} \to \mathbb{R}$

$sumafil(A, k) \to \sum_{j=0}^{m-1} A_{k,j}$

Función que hace la suma de la columna

Entradas: 2 variables: Una matriz $A \in \mathbb{R}^{n\times m}$ y el índice de la columna que se desa sumar $(k : k \in \mathbb{N} \wedge \in [0, m-1])$ $(k \in \mathbb{N}^{[0,m-1]})$

Salidas: 1 variable: La suma de los elementos de la columna $(suma \in \mathbb{R})$

Relación:

$sumacol : \{0,1\}^{n\times m} \times \mathbb{N}^{[0,m-1]} \to \mathbb{R}$

$sumacol(A, k) \to \sum_{i=0}^{n-1} A_{i,k}$

Función que determina si inyectiva:

Entradas: Una matriz $A \in \{0,1\}^{n\times m}$

Salidas: 1 valor de verdad (True si la relación es función inyectiva (la suma de todos los componentes de cada fila y cada columna es ≤ 1) o False en otro caso) $v \in \mathbb{B}$

Relación:

$inyectiva : \{0,1\}^{n\times m} \to \mathbb{B}$

$$inyectiva(A) \to \begin{cases} True \quad si & \forall_{i=0}^{n-1} sumafil(A,i) \leq 1 \wedge \forall_{j=0}^{m-1} sumacol(A,j) \leq 1 \\ False & \text{En otro caso} \end{cases}$$

3, Codificación

El código en el que se encuentran estas funciones esta disponible en el archivo llamado «Ejercicios_BINARIOS_SimonRamos.py». De la línea de código 204 a la 222, y en la función principal estas se llaman de la línea de código 285 a la 289

Ejercicio 69:
SOBREYECTIVIDAD: Determina si la relación es una función sobreyectiva.
1.1 Objetos conocidos:

- Se conoce una relación binaria entre conjuntos, la cual se representa mediante una matriz de unos y ceros. En este caso, la relación se puede dar entre dos conjuntos distintos, por lo que no necesariamente la matriz será cuadrada

1.2 Objetos desconocidos:

- Se desconoce si la matriz dada representa una función sobreyectiva o no

1.3 Relación entre los objetos:

- Para comporbar sobreyectividad, en primer lugar se debe comporbar si se trata de una función. Una relación binaria es función si cada elemento del conjunto de salida (dominio)A se relaciona con un único elemento de un conjunto de llegada B (codominio). No obstante, no necesariamente a todos los elementos de A le corresponde un elemento en B, y puede ser que un elemento de B se relaciona con más de un elemento de A. Si lo vemos en la representación matricial, que una relación sea función implicaría en cada fila de la matriz puede haber como máximo solo un 1.
- SIn embargo, al querer comprobar sobreyectividad todos los elementos en B deben estar relacionados con algún elemento de A. En una función sobreyectiva no importa si un elemento de B está relacionado con más de un elemento de A. Para calcular si una matriz binaria representa a una función podríamos sumar cada fila de la matriz, y ya que debe haber como máximo un 1 en esta fila, la suma de sus componentes debe ser ≤ 1. Del mismo modo, para comprobar si es sobreyectiva, la suma de los elementos de cada columna debe ser > 0

2. Diseño y prueba conceptual
Función que hace la suma de la fila

Entradas: 2 variables: Una matriz $A \in \mathbb{R}^{n\times m}$ y el índice de la fila que se desa sumar ($k : k \in \mathbb{N} \wedge \in [0, n-1]$) ($k \in \mathbb{N}^{[0,n-1]}$)

Salidas: 1 variable: La suma de los elementos de la fila ($suma \in \mathbb{R}$)

Relación:

$sumafil : \{0,1\}^{n\times m} \times \mathbb{N}^{[0,n-1]} \rightarrow \mathbb{R}$

$sumafil(A, k) \rightarrow \sum_{j=0}^{m-1} A_{k,j}$

Función que hace la suma de la columna

Entradas: 2 variables: Una matriz $A \in \mathbb{R}^{n\times m}$ y el índice de la columna que se desa sumar ($k : k \in \mathbb{N} \wedge \in [0, m-1]$) ($k \in \mathbb{N}^{[0,m-1]}$)

Salidas: 1 variable: La suma de los elementos de la columna ($suma \in \mathbb{R}$)

Relación:

$sumacol : \{0,1\}^{n\times m} \times \mathbb{N}^{[0,m-1]} \to \mathbb{R}$

$sumacol(A,k) \to \sum_{i=0}^{n-1} A_{i,k}$

Función que determina si es sobreyectiva:

Entradas: Una matriz $A \in \{0,1\}^{n\times m}$

Salidas: 1 valor de verdad (True si la relación es función es sobreyectiva (la suma de todos los componentes de cada fila ≤ 1 y de cada columna es > 0) o False en otro caso) $v \in \mathbb{B}$

Relación:

$sobreyectiva : \{0,1\}^{n\times m} \to \mathbb{B}$

$$sobreyectiva(A) \to \begin{cases} True \quad si \quad \forall_{i=0}^{n-1} sumafil(A,i) \leq 1 \wedge \forall_{j=0}^{m-1} sumacol(A,j) > 0 \\ False \quad \text{En otro caso} \end{cases}$$

3, Codificación

El código en el que se encuentran estas funciones esta disponible en el archivo llamado «Ejercicios_BINARIOS_SimonRamos.py». De la línea de código 225 a la 236, y en la función principal estas se llaman de la línea de código 290 a la 294

Ejercicio 70
SALIR: Permite al usuario salir de la aplicación.
1. Análisis y especificación
1.1 Objetos conocidos:

- Ya que mediante un menú se deben escoger las operaciones especificadas en los puntos 60-69, para salir se debe pedir esta función especifica que terminará con las operaciones antes pedidas.

1.2 Objetos desconocidos:

- Se desconoce si el usuario quiere salir del menú presentado

1.3 Relación entre los objetos:

- Si el usuario desea salir, debe escoger esta función en el menú presentado presionando un número específico

2. Diseño y prueba conceptual

En el programa principal se presentarán como opciones las operaciones descritas en los puntos 60-69. Además tendrá la opción de salir

Entradas: 1 variable: $x \in \mathbb{Z}$la cual indica la operación que se requiere realizar

Salidas: El procedimiento que se ha pedido

Relación:

$$
\begin{aligned}
men\acute{u} : \mathbb{Z} &\rightarrow funciones \\
menu(x) &\rightarrow \begin{cases} UNION & si & x = 1 \\ INTERSECCION & si & x = 2 \\ SIMETRIA & si & x = 3 \\ REFLEXIVIDAD & si & x = 4 \\ TRANSITIVIDAD & si & x = 5 \\ ORDEN & si & x = 6 \\ EQUIVALENCIA & si & x = 7 \\ FUNCION & si & x = 8 \\ INYECTIVIDAD & si & x = 9 \\ SOBREYECTIVIDAD & si & x = 10 \\ SALIR & & \text{En otro caso} \end{cases}
\end{aligned}
$$

3. Codificación

El código en el que se resuelve el ejercicio se encuentra en el archivo llamado «Ejercicios_BINARIOS_SimonRamos.py». El programa principal donde se llama a todas las funciones y se desarrolla el menú con las opciones se encuentra de la línea de código 237 a la 305.