UBA – Facultad de Ciencias Exactas y Naturales – Departamento de Computación

Algoritmos y Estructuras de Datos I

Primer cuatrimestre de 2022

22 de Mayo del 2022

Taller de matrices y tableros

Este taller contiene una serie de ejercicios sobre matrices y tableros para resolver y en el archivo `template-alumnos.zip` encontrarán una serie de casos de test, como habíamos visto en laboratorios pasados. Como parte del ejercicio de este labo, deben convertir los casos de test propuestos en el archivo cases.cpp a su versión GTEST como vimos en el laboratorio pasado. Se incluye en `template-alumnos.zip` el folder lib con GTEST.

Para este laboratorio, además deberán crear el `CMakeList.txt` en la carpeta que es abierta como proyecto de CLion (pueden basarse en el proyecto del laboratorio pasado como ejemplo). No olvidar modificar los archivos fuente (`SOURCE_FILES` y el nombre del proyecto)). Para compilar con GTEST utilicen como base el del labo correspondiente.

No olvidarse de descomprimir la librería GTEST dentro del folder `lib`.

Ejercicio 1: Dados dos vectores, calcular la matriz que resulta de hacer el producto vectorial entre ambos.

`proc productoVectorial` (in u: $seq\langle\mathbb{Z}\rangle$, in v: $seq\langle\mathbb{Z}\rangle$, out res: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
  Pre $\{True\}$
  Post $\{esMatrizDeAltoYAncho(res, |u|, |v|) \land_L cadaCoordenadaEsElProducto(res, u, v)\}$
}

`pred esMatrizDeAltoYAncho` (mat: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, alto: $\mathbb{Z}$, ancho: $\mathbb{Z}$) {
  $|mat| = alto \land todasLasFilasTienenAncho(mat, ancho)$
}

`pred todasLasFilasTienenAncho` (matriz: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, ancho: $\mathbb{Z}$) {
  $(\forall i : \mathbb{Z})(0 \leq i < |matriz| \longrightarrow_L |matriz[i]| = ancho)$
}

`pred cadaCoordenadaEsElProducto` (producto: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, vectorFila: $seq\langle\mathbb{Z}\rangle$, vectorColumna: $seq\langle\mathbb{Z}\rangle$) {
  $(\forall i : \mathbb{Z})(\forall j : \mathbb{Z})(0 \leq i < |vectorFila| \land_L 0 \leq j < |vectorColumna| \longrightarrow_L producto[i][j] = vectorFila[i] * vectorColumna[j])$
}

Ejercicio 2: Dada una matriz cuadrada, modificarla para obtener su traspuesta.

`proc trasponer` (inout m: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
  Pre $\{m = m_0 \land esCuadrada(m)\}$
  Post $\{esMatrizDeAltoYAncho(m, |m_0|, |m_0|) \land_L cadaCoordenadaEsLaTraspuesta(m, m_0)\}$
}

`pred esCuadrada` (m: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
  $(\forall i : \mathbb{Z})(0 \leq i < |m| \rightarrow_L |m[i]| = |m|)$
}

`pred cadaCoordenadaEsLaTraspuesta` (traspuesta: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, original: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
  $(\forall i : \mathbb{Z})(\forall j : \mathbb{Z})(0 \leq i < |traspuesta| \land 0 \leq j < |traspuesta| \rightarrow_L traspuesta[i][j] = original[j][i]))$
}

Ejercicio 3: Multiplicar matrices.

`proc multiplicar` (in m1: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, in m2: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, out res: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
  Pre $\{|m1| > 0 \land |m2| > 0 \land_L |m2[0]| > 0 \land |m1[0]| = |m2| \land_L todasLasFilasTienenAncho(m1, |m1[0]|) \land todasLasFilasTienenAncho(m2, |m2[0]|)\}$
  Post $\{esMatrizDeAltoYAncho(res, |m1|, |m2[0]|) \land_L (\forall i : \mathbb{Z})(\forall j : \mathbb{Z})(0 \leq i < |m1| \land_L 0 \leq j < |m2[0]| \longrightarrow_L res[i][j] = \sum_{k=0}^{|m2|-1} m1[i][k] * m2[k][j])\}$
}

Ejercicio 4: Dada una matriz, devolver otra matriz reemplazando cada casillero por el promedio de la región compuesta por sus vecinos más el valor del centro.

`proc promediar` (in m: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, out res: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
  Pre $\{|m| \geq 2 \land_L |m[0]| \geq 2 \land_L todasLasFilasTienenAncho(m, |m[0]|)\}$
  Post $\{esMatrizDeAltoYAncho(res, |m|, |m[0]|) \land_L cadaCoordenadaEsElPromedioDeLaRegion(res, m)\}$
}

`pred cadaCoordenadaEsElPromedioDeLaRegion` (res: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, m: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {

$(\forall i:\mathbb{Z})(\forall j:\mathbb{Z})\ 0 \le i < |res| \land 0 \le j < |res[i]| \longrightarrow_L res[i][j] = promedioVecinos(m,i,j)$
}
`aux promedioVecinos` (m : $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, i: $\mathbb{Z}$, j: $\mathbb{Z}$) : $\mathbb{Z}$ = $sumaVecinos(m,i,j)$ div $cantidadVecinos(m,i,j)$;

`aux sumaVecinos` (m: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, i: $\mathbb{Z}$, j: $\mathbb{Z}$) : $\mathbb{Z}$ = $\displaystyle\sum_{a=i-1}^{i+1}\sum_{b=j-1}^{j+1}$ if $vecinosEnRango(m,a,b)$ then $m[a][b]$ else 0 fi ;

`aux cantidadVecinos` (m: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, i: $\mathbb{Z}$, j: $\mathbb{Z}$) : $\mathbb{Z}$ = $\displaystyle\sum_{a=i-1}^{i+1}\sum_{b=j-1}^{j+1}$ if $vecinosEnRango(m,a,b)$ then 1 else 0 fi ;

`pred vecinosEnRango` (m: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, i: $\mathbb{Z}$, j: $\mathbb{Z}$) {
$0 \le i < |m| \land 0 \le j < |m[a]|$
}

Ejercicio 5: Contar cuántos picos tiene una matriz, donde un pico es un elemento que es mayor que todos sus vecinos.

proc `contarPicos` (in m: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, out res: $\mathbb{Z}$) {
`Pre` $\{|m| \ge 2 \land_L |m[0]| \ge 2 \land todasLasFilasTienenAncho(m, |m[0]|)\}$
`Post` $\{res = \displaystyle\sum_{i=0}^{|m|}\sum_{j=0}^{|m[i]|}$ if $esPico(m,i,j)$ then 1 else 0 fi}
}
`pred esPico` (m : $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, i: $\mathbb{Z}$, j: $\mathbb{Z}$) {
$(\forall a:\mathbb{Z})(\forall b:\mathbb{Z})(esVecino(m,i,j,a,b) \longrightarrow_L m[i][j] > m[a][b])$
}
`pred esVecino` (m : $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, i: $\mathbb{Z}$, j: $\mathbb{Z}$, a: $\mathbb{Z}$, b: $\mathbb{Z}$) {
$(a \ne i \lor b \ne j) \land i-1 \le a \le i+1 \land j-1 \le b \le j+1 \land enRango(m,a,b)$
}

Ejercicio 6: Dada una matriz cuadrada, decidir si es triangular (inferior o superior).

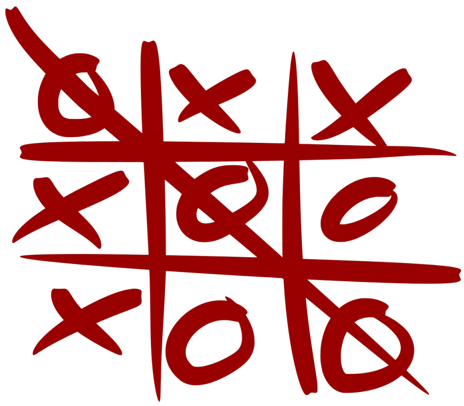
proc `esTriangular` (in m: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, out res: Bool) {
`Pre` $\{esCuadrada(m)\}$
`Post` $\{res = true \leftrightarrow esTriangularSuperior(m) \lor esTriangularInferior(m)\}$
}
`pred esTriangularInferior` (m: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) {
$(\forall i:\mathbb{Z})(\forall j:\mathbb{Z})(enRango(m,i,j) \land i < j \longrightarrow_L m[i][j] == 0)$
}
`pred esTriangularSuperior` (m: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) {
$(\forall i:\mathbb{Z})(\forall j:\mathbb{Z})(enRango(m,i,j) \land j < i \longrightarrow_L m[i][j] == 0)$
}

Ejercicio 7: Decidir si, dado un tablero (no necesariamente de 8 x 8) con reinas de ajedrez, existen dos reinas que se amenazan entre sí.

proc `hayAmenaza` (in m: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, out res: Bool) {
`Pre` $\{|m| \ge 2 \land_L |m[0]| \ge 2 \land_L todasLasFilasTienenAncho(m, |m[0]|) \land esBinaria(m)\}$
`Post` $\{res = true \leftrightarrow existeAmenaza(m)\}$
}
`pred esBinaria` (m: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) {
$(\forall i:\mathbb{Z})(\forall j:\mathbb{Z})(0 \le i < |m| \land 0 \le j < |m[i]| \rightarrow_L 0 \le m[i][j] \le 1)$
}
`pred existeAmenaza` (m: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) {
$(\exists i1:\mathbb{Z})(0 \le i1 < |m| \land_L (\exists j1:\mathbb{Z})(0 \le j1 < |m[i1]| \land_L m[i1][j1] = 1 \land amenazaAlguna(m,i1,j1)))$
}
`pred amenazaAlguna` (m: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, i1: $\mathbb{Z}$, j1: $\mathbb{Z}$) {
$(\exists i2:\mathbb{Z})(0 \le i2 < |m| \land_L (\exists j2:\mathbb{Z})(0 \le j2 < |m[i2]| \land_L m[i2][j2] = 1 \land seAmenazan(i1,j1,i2,j2)))$
}
`pred seAmenazan` (i1: $\mathbb{Z}$, j1: $\mathbb{Z}$, i2: $\mathbb{Z}$, j2: $\mathbb{Z}$) {
$(i1 \ne i2 \lor j1 \ne j2) \land (i1 = i2\ \lor\ j1 = j2\ \lor\ abs(i1-i2) = abs(j1-j2))$
}
`aux abs` (t : $\mathbb{Z}$) : $\mathbb{Z}$ = if $t \ge 0$ then t else $-t$ fi ;

Ejercicio 8: Dada una matriz cuadrada de $n \times n$, devolver la diferencia absoluta entre la suma de sus dos diagonales. Una diagonal es la que empieza en la posición (0,0) y termina en (n-1,n-1), y la otra que va entre las posiciones (0,n-1) y (n-1,0).

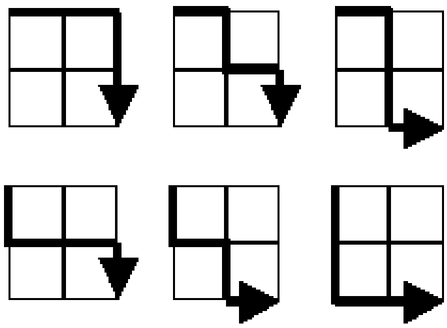
Ejercicio Adicional TaTeTi: Escribir un algoritmo que verifique si una partida de TaTeTi está terminada.



Muy fácil? Ahora generalizarlo para un tateti de N columnas y N filas.

Generar varios TESTs para verificar la implementación.

Ejercicio Adicional "Willy, el robot" Supongamos que tenemos un robot sentado en la esquina arriba izquierda de una grilla de X*Y. El robot se puede mover en dos direcciones: para abajo y para la derecha.



Escribir un algoritmo que determine cuántos caminos posibles puede hacer el robot para llegar de la posición (0,0) a la (X,Y). Queda prohibido usar la fórmula cerrada para calcularlo.

Generar varios TESTs para verificar la implementación.