DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Trabajo Práctico 1: Especificación y WP

"En Búsqueda del Camino"

4 de septiembre de 2024 — Ciencias de la Computación/Ciencia de Datos

Grupo puntoJava

Integrante	LU	Correo electrónico
Gremes, Juan Ignacio	21/24	juanigremes@gmail.com
Apellido, Nombre2	002/01	email2@dominio.com
Apellido, Nombre3	003/01	email3@dominio.com
Apellido, Nombre4	004/01	email4@dominio.com

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellón I/Planta Baja)
Intendente Güiraldes 2610 - C1428EGA
Ciudad Autónoma de Buenos Aires - Rep. Argentina
Tel/Fax: (++54 +11) 4576-3300
http://www.exactas.uba.ar

1. Especificación

1.1. grandesCiudades:

A partir de una lista de ciudades, devuelve aquellas que tienen más de 50.000 habitantes.

proc grandesCiudades (in ciudades : $seq\langle \text{String} \times \mathbb{Z} \rangle$) : $seq\langle \text{String} \times \mathbb{Z} \rangle$
requiere $\{True\}$
asegura $\{(\forall i : \mathbb{Z})\ ((o \leq i < |ciudades|) \longrightarrow_L ((res[i] \in ciudades) \wedge_L (res[i]_1 > 50{,}000)))\}$
asegura $\{(\forall j : \mathbb{Z})\ (((0 \leq j < |ciudades|) \wedge (ciudades[j] > 50{,}000)) \longrightarrow_L (ciudades[j] \in res))\}$

1.2. sumaDeHabitantes:

Por cuestiones de planificación urbana, las ciudades registran sus habitantes mayores de edad por un lado y menores de edad por el otro.

Dadas dos listas de ciudades del mismo largo con los mismos nombres, una con sus habitantes mayores y otra con sus habitantes menores, este procedimiento debe devolver una lista de ciudades con la cantidad total de sus habitantes.

proc sumaDeHabitantes (in menoresDeCiudades : $seq\langle \text{String} \times \mathbb{Z} \rangle$, in mayoresDeCiudades : $seq\langle \text{String} \times \mathbb{Z} \rangle$) : $seq\langle \text{String} \times \mathbb{Z} \rangle$
requiere $\{|menoresDeCiudades| = |mayoresDeCiudades|\}$
requiere $\{(\forall i : \mathbb{Z})\ ((0 \leq i < |menoresDeCiudades|) \longrightarrow_L (\#Apariciones(menoresDeCiudades[i]_0, menoresDeCiudades)$ $\#Apariciones(menoresDeCiudades[i]_0, mayoresDeCiudades)))\}$
asegura $\{|res| = |menoresDeCiudades|\}$
asegura $\{(\forall j : \mathbb{Z})\ ((0 \leq j < |res|) \longrightarrow_L (\#Apariciones(res[j]_0, res) = \#Apariciones(res[j]_0, menoresDeCiudades)))\}$
asegura $\{(\forall x, y, z : \mathbb{Z})\ (((0 \leq x < |res|) \wedge (0 \leq y < |res|) \wedge (0 \leq z < |res|) \wedge_L (menoresDeCiudades[x]_0 =$ $mayoresDeCiudades[y]_0 = res[z]_0)) \longrightarrow_L (res[z]_1 = menoresDeCiudades[x]_1 + mayoresDeCiudades[y]_1))\}$

aux #Apariciones (ciudad : String, ciudades : $seq\langle \text{String} \times \mathbb{Z} \rangle$) : $\mathbb{Z} = \sum_{k=0}^{|ciudades|-1} IfThenElse(ciudad = ciudades[k]_0, 1, 0)$;

1.3. hayCamino:

Un mapa de ciudades está conformada por ciudades y caminos que unen a algunas de ellas. A partir de este mapa, podemos definir las distancias entre ciudades como una matriz donde cada celda i, j representa la distancia entre la ciudad i y la ciudad j. Una distancia de 0 equivale a no haber camino entre i y j. Notar que la distancia de una ciudad hacia sí misma es cero y la distancia entre A y B es la misma que entre B y A.

Dadas dos ciudades y una matriz de distancias, se pide determinar si existe un camino entre ambas ciudades.

proc hayCamino (in distancias : $seq\langle seq\langle \mathbb{Z} \rangle \rangle$, in desde : $\mathbb{Z}$, in hasta : $\mathbb{Z}$) : Bool
requiere $\{True\}$
asegura $\{(\exists c : seq\langle \mathbb{Z} \rangle)\ ((c[0] = desde) \wedge_L (c[|c|-1] = hasta) \wedge ((\forall n : \mathbb{Z})\ ((0 \leq n < |c|-1) \longrightarrow_L (hayCaminoDirecto(c[n], c[n+$ $1], distancias)))))\}$

pred hayCaminoDirecto (c1 : $\mathbb{Z}$, c2 : $\mathbb{Z}$, distancias : $seq\langle seq\langle \mathbb{Z} \rangle \rangle$) {
$distancias[c1][c2] \neq 0$
}

1.4. cantidadCaminosNSaltos:

Dentro del contexto de redes informáticas, nos interesa contar la cantidad de "saltos" que realizan los paquetes de datos, donde un salto se define como pasar por un nodo. Así como definimos la matriz de distancias, podemos definir la matriz de conexión entre nodos, donde cada celda i, j tiene un 1 si hay un único camino a un salto de distancia entre el nodo i y el nodo j, y un 0 en caso contrario. En este caso, se trata de una matriz de conexión de orden 1, ya que indica cuáles pares de nodos poseen 1 camino entre ellos a 1 salto de distancia.

Dada la matriz de conexión de orden 1, este procedimineto debe obtener aquella de orden n que indica cuántos caminos de n saltos hay entre los distintos nodos. Notar que la multiplicación de una matriz de conexión de orden 1 consigo misma nos da la matriz de conexión de orden 2, y así sucesivamente.

1.5. caminoMínimo

Dada una matriz de distancias, una ciudad de origen y una ciudad de destino, este procedimiento debe devolver la lista de ciudades que conforman el camino más corto entre ambas. En caso de no existir un camino, se debe devolver una lista vacía.

2. Demostraciones de Correctitud

2.1. esto ya estaba

Lo principal: las fórmulas. Se puede poner en una linea, como $x_i = x_{i-1} + x_{i-2}$, o ponerse más grande:

$$\sum_{i=0}^{n} i \tag{1}$$

Y se pueden citar ecuaciones con `\eqref{nombreDeEq}`: (1)
Ejemplo de itemizado:

- Item 1
- Item 2
- Item 3

Ejemplo de enumerado con menor distancia entre items:

1. Item 1
2. Item 2
3. Item 3

Podemos escribir mucho texto. Mucho texto. Mucho texto. Mucho texto. Mucho texto. Mucho texto. Mucho texto. Mucho texto. Mucho texto. Mucho texto.
Otro párrafo. Otro párrafo. Otro párrafo. Otro párrafo. Otro párrafo. Otro párrafo. Otro párrafo. Otro párrafo. Otro párrafo. Otro párrafo. Otro párrafo. Otro párrafo. Otro párrafo.

Le agregamos una separación entre párrafos. Le agregamos una separación entre párrafos. Le agregamos una separación entre párrafos. Le agregamos una separación entre párrafos. Le agregamos una separación entre párrafos.

La tabla 1 es un ejemplo de cómo se hace una tabla.

Col1	Col2	Col2	Col3
1	6	87837	787
2	7	78	5415
3	545	778	7507
4	545	18744	7560
5	88	788	6344

Tabla 1: Ejemplo de tabla

La figura 2 es un ejemplo de cómo se agrega una imagen.

Figura 1: Ejemplo de figura

(a) Logo de LaTeX

(b) Logo de TeX

Figura 2: Ejemplo para poner dos figuras juntas. Y citarlas por separado a (a) y (b).

```
1 res := 0;
2 i := 0;
3 while (i < s.size()) do
4     res := res + s[i];
5     i := i + 1
6 endwhile
```

Código 1: Ejemplo de código (usando los estilos de la cátedra, ver las macros para más detalles)

Si se pone un label al `lstlisting`, se puede referenciar: Código 1.

2.2. Macros de la cátedra para especificar

`proc nombre` (in paramIn : $\mathbb{N}$, inout paramInout : $seq\langle\mathbb{Z}\rangle$) : tipoRes
`requiere` {$expresionBooleana1$}
`asegura` {$expresionBooleana2$}
`aux auxiliar1` (parametros) : tipoRes = $expresion$;
`pred pred1` (parametros) {
$expresion$
}

`aux auxiliarSuelto` (parametros) : tipoRes = $expresion$;
`pred predSuelto` (parametros) {
$(\forall variable : tipo)\ (algo \longrightarrow_L expresion)$
}
`pred predSuelto` (parametros) {
$(\exists variable : tipo)\ (algo \wedge_L expresion)$
}