DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Trabajo práctico 1

Especificación y WP

8 de septiembre de 2024 — Algoritmos y Estructura de Datos

SinGrupo4

Integrante	LU	Correo electrónico
Algañaraz, Franco	001/01	francoarga10@gmail.com
Illescas, Marcos	390/14	marcosillescas90@gmail.com
Bahamonde, Matias	003/01	matubaham@gmail.com
Marión, Ian Pablo	004/01	ianfrodin@gmail.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. Especifiación

1.1. grandesCiudades

proc grandesCiudades (in ciudades :$seq\langle struct < nombre : string, habitantes : \mathbb{Z} >\rangle$) : $seq\langle struct < nombre : string, habitantes : \mathbb{Z} >\rangle$

requiere {true}

asegura {$|res| = (\sum_{i=0}^{|ciudades|-1}$ if $ciudades[i].habitantes > 50000$ then 1 else 0 fi) $\wedge_L$ $(\forall i : Z)(0 \leq i < |res| \rightarrow (\exists j : Z)(0 \leq j < |ciudades| \wedge_L res[i] = ciudades[j] \wedge_L ciudades[j].habitantes > 50000))$}

1.2. sumaDeHabitantes

proc sumaDeHabitantes (in menoresDeCiudades : $seq\langle struct < nombre : string, habitantes : \mathbb{Z} >\rangle$, in mayoresDeCiudades : $seq\langle struct < nombre : string, habitantes : \mathbb{Z} >\rangle$) : $seq\langle struct < nombre : string, habitantes : \mathbb{Z} >\rangle$

requiere {$|menoresDeCiudades| = |mayoresDeCiudades| \wedge_L (\forall i : Z)(0 \leq i < |menoresDeCiudades| \rightarrow (\exists j : \mathbb{Z})$ ($menoresDeCiudades[i].nombre = mayoresDeCiudades[j].nombre$))}

asegura {$|res| = |menoresDeCiudades| \wedge_L (\forall i : Z)(0 \leq i < |res| \rightarrow (\exists j, k : \mathbb{Z})(0 \leq j < |menoresDeCiudad| \wedge 0 \leq k < |mayoresDeCiudad| \wedge_L res[i].nombre = menoresDeCiudad[j].nombre = mayoresDeCiudad[k].nombre \longrightarrow_L res[i].habitantes = menoresDeCiudad[j].habitantes + mayoresDeCiudad[k].habitantes))$}

1.3. hayCamino

proc hayCamino (in distancias : $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, in desde : $\mathbb{Z}$, in hasta : $\mathbb{Z}$) : Bool

requiere {$0 \leq desde < |distancias| \wedge 0 \leq hasta < |distancias|$}

asegura {$res =$ true $\iff existeCamino(distancia, desde, hasta)$}

pred existeCamino (in distancias : $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, in desde : $\mathbb{Z}$, in hasta : $\mathbb{Z}$) {

$(\exists camino : seq\langle \mathbb{Z}\rangle)$ $(|camino| \geq 2 \wedge camino[0] = desde \wedge camino[|camino|-1] = hasta \wedge (\forall i : \mathbb{Z})(0 \leq i < |camino|-1 \rightarrow distancias[camino[i]][camino[i+1]] > 0))$

}

1.4. cantidadCaminosNSaltos

proc cantidadCaminosNSaltos (in conexion : $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, in n : $\mathbb{Z}$) : $seq\langle \mathbb{Z}\rangle$

requiere {$n \geq 1 \wedge conexion = C_0$}

asegura {$conexion = {C_0}^n$}

1.5. caminoMínimo

proc caminoMinimo (in origen : $\mathbb{N}$, in destino : $\mathbb{Z}$, in distancias : $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) : $seq\langle \mathbb{Z}\rangle$

requiere {$0 \leq origen < |distancias| \wedge 0 \leq destino < |distancias|$}

asegura {$(res = [\,] \iff existeCamino(distancias, origen, destino) \vee origen = destino) \wedge (res \neq [\,] \implies (\forall camino : seq\langle \mathbb{Z}\rangle)(camino[0] = origen \wedge camino[|camino| - 1] = destino \wedge sumaDistancias(res, distancias) \leq sumaDistancias(camino, distancias)))$}

aux sumaDistancia (in camino : $seq\langle \mathbb{Z}\rangle$, in distancias : $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) : $\mathbb{Z} = \sum_{i=0}^{|camino|-2} distancias[camino[i]][camino[i+1]]$;

2. Demostraciones de correctitud

2.1. Demostrar que la implementación es correcta con respecto a la especificación.

2.2. Demostrar que el valor devuelto es mayor a 50.000.