DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Trabajo práctico 1

Especificación y WP

---

6 de octubre de 2024 — Algoritmos y Estructura de Datos

SinGrupo4

Integrante	LU	Correo electrónico
Algañaraz, Franco	1092/22	`francoarga10@gmail.com`
Illescas, Marcos	390/14	`marcosillescas90@gmail.com`
Bahamonde, Matias	694/21	`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. Especificación

1.0. Predicados recurrentes

`pred habitantesPositivos` (s: $seq\langle Ciudad\rangle$) {
$(\forall j : \mathbb{Z})(0 \leq j < |s| \longrightarrow_L s[j].habitantes \geq 0)$
}
`pred noCiudadesRepetidas` (s: $seq\langle Ciudad\rangle$) {
$(\forall i : \mathbb{Z})(\forall j : \mathbb{Z})(0 \leq i < j < |s| \longrightarrow_L s[i].nombre \neq s[j].nombre)$
}
`pred matrizCuadrada` (s: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) {
$(\forall i : \mathbb{Z})(0 \leq i < |s| \longrightarrow_L |s| = |s[i]|)$
}
`pred matrizPositivaSimetricaConCerosDiagonal` (s: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) {
$(\forall i : \mathbb{Z})(\forall j : \mathbb{Z})(0 \leq i < j < |s| \longrightarrow_L s[i][i] = 0 \wedge s[i][j] = s[j][i] \wedge s[i][j] \geq 0)$
}
`pred esCamino` (camino: $seq\langle \mathbb{Z}\rangle$, desde: $\mathbb{Z}$, hasta: $\mathbb{Z}$, A: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) {
$|camino| \geq 2 \wedge_L camino[0] = desde \wedge camino[|camino| - 1] = hasta \wedge$
$(\forall i : \mathbb{Z})(0 \leq i < |camino| \longrightarrow_L 0 \leq camino[i] < |A|) \wedge$
$(\forall j : \mathbb{Z})(0 \leq j < |camino| - 1 \longrightarrow_L A[camino[j]][camino[j + 1]] > 0)$
}

1.1. grandesCiudades

`proc grandesCiudades` (in ciudades: $seq\langle Ciudad\rangle$) : $seq\langle Ciudad\rangle$
`requiere` $\{habitantesPositivos(ciudades) \wedge noCiudadesRepetidas(ciudades)\}$
`asegura` $\{|res| = cantidadCiudadesMayor50000(ciudades) \ \wedge noCiudadesRepetidas(res) \wedge$
$(\forall i : \mathbb{Z})(0 \leq i < |res| \longrightarrow_L (\exists j : \mathbb{Z})(0 \leq j < |ciudades| \wedge_L (res[i] = ciudades[j] \wedge ciudades[j].habitantes > 50000)))\}$

`aux cantidadCiudadesMayor50000` (in s: $seq\langle Ciudad\rangle$) : $\mathbb{Z}$ =
;
$\sum_{i=0}^{|s|-1}$ if $s[i].habitantes > 50000$ then 1 else 0 fi;

1.2. sumaDeHabitantes

`proc sumaDeHabitantes` (in menoresDeCiudades: $seq\langle Ciudad\rangle$, in mayoresDeCiudades: $seq\langle Ciudad\rangle$) : $seq\langle Ciudad\rangle$
`requiere` $\{|menoresDeCiudades| = |mayoresDeCiudades| \ \wedge$
$mismosNombres(menoresDeCiudades, mayoresDeCiudades) \ \wedge$
$habitantesPositivos(menoresDeCiudades) \ \wedge \ habitantesPositivos(mayoresDeCiudades) \ \wedge$
$noCiudadesRepetidas(menoresDeCiudades) \ \wedge \ noCiudadesRepetidas(mayoresDeCiudades)\}$
`asegura` $\{|res| = |menoresDeCiudades| \ \wedge$
$(\forall i : \mathbb{Z})(0 \leq i < |res| \longrightarrow_L (\exists j : \mathbb{Z})(\exists k : \mathbb{Z})(0 \leq j < k < |menoresDeCiudad| \ \wedge_L$
$siCoincidenNombresSumoHabitantes(res, menoresDeCiudades, mayoresDeCiudades, i, j, k)))\}$

`pred mismosNombres` (s: $seq\langle Ciudad\rangle$, t: $seq\langle Ciudad\rangle$) {
$(\forall i : \mathbb{Z})(0 \leq i < |s| \longrightarrow_L (\exists j : \mathbb{Z})(0 \leq j < |t| \wedge_L s[i].nombre = t[j].nombre))$
}
`pred siCoincidenNombresSumoHabitantes` (res: $seq\langle Ciudad\rangle$, men: $seq\langle Ciudad\rangle$, may: $seq\langle Ciudad\rangle$, i: $\mathbb{Z}$, j: $\mathbb{Z}$, k: $\mathbb{Z}$) {
$(res[i].nombre = men[j].nombre \ \wedge \ res[i].nombre = may[k].nombre) \ \wedge$
$res[i].habitantes = men[j].habitantes + may[k].habitantes$
}

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` $\{matrizCuadrada(distancias) \wedge matrizPositivaSimetricaConCerosDiagonal(distancias) \wedge$
$0 \leq desde < |distancias| \wedge 0 \leq hasta < |distancias|\}$
`asegura` $\{res = \text{true} \longleftrightarrow (\exists c : seq\langle \mathbb{Z}\rangle)(esCamino(c, desde, hasta, distancias))\}$

1.4. cantidadCaminosNSaltos

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

`requiere` $\{n \geq 1 \wedge matrizCuadrada(conexion) \wedge$
$matrizPositivaSimetricaConCerosDiagonal(conexion) \wedge unosYceros(conexion) \wedge conexion = C_0\}$

`asegura` $\{(\exists s : seq\langle seq\langle seq\langle \mathbb{Z}\rangle\rangle\rangle)(|s| = n \wedge_L s[0] = C_0 \wedge$
$(\forall i : \mathbb{Z})(0 \leq i < n \longrightarrow_L matrizCuadrada(s[i]) \wedge matrizPositivaSimetrica(s[i])) \wedge$
$(\forall j : \mathbb{Z})(1 \leq j < n \longrightarrow_L esMultiplicacionMatricialDe(s[j], s[j-1], C_0)) \wedge conexion = s[n-1])\}$

`pred unosYceros` (s: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) {
$(\forall i : \mathbb{Z})(\forall j : \mathbb{Z})(0 \leq i < j < |s| \longrightarrow_L s[i][j] = 0 \vee s[i][j] = 1)$
}

`pred matrizPositivaSimetrica` (s: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) {
$(\forall i : \mathbb{Z})(\forall j : \mathbb{Z})(0 \leq i < j < |s| \longrightarrow_L s[i][j] = s[j][i] \wedge s[i][j] \geq 0)$
}

`pred esMultiplicacionMatricialDe` (A: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, B: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, C: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) {
$(\forall i : \mathbb{Z})(\forall j : \mathbb{Z})(0 \leq i \leq j < |A| \longrightarrow_L A[i][j] = \sum_{k=0}^{|A|-1} B[i][k] * C[k][j])$
}

1.5. caminoMínimo

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

`requiere` $\{matrizCuadrada(distancias) \wedge matrizPositivaSimetricaConCerosDiagonal(distancias) \wedge$
$0 \leq origen < |distancias| \wedge 0 \leq destino < |distancias|\}$

`asegura` $\{(|res| = 0 \wedge \neg(\exists s : seq\langle \mathbb{Z}\rangle)(esCamino(s, origen, hasta, distancias))) \vee$
$(esCamino(res, origen, hasta, distancias) \wedge (\forall c : seq\langle \mathbb{Z}\rangle)(esCamino(c, origen, destino, distancias) \longrightarrow$
$distanciaTotal(res, distancias) \leq distanciaTotal(c, distancias)))\}$

`aux distanciaTotal` (in s: $seq\langle \mathbb{Z}\rangle$, in A: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) : $\mathbb{Z} = \sum_{i=0}^{|c|-2} A[c[i]][c[i+1]]$;

2. Demostraciones de correctitud

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

`proc poblacionTotal` (in ciudades: $seq\langle Ciudad\rangle$) : $\mathbb{Z}$

`requiere` $\{(\exists j : \mathbb{Z})(0 \leq j < |ciudades| \wedge_L ciudades[j].habitantes > 50000) \wedge$
$(\forall k : \mathbb{Z})(0 \leq k < |ciudades| \longrightarrow_L ciudades[i].habitantes \geq 0) \wedge$
$(\forall m : \mathbb{Z})(\forall n : \mathbb{Z})(0 \leq m < n < |ciudades| \longrightarrow_L ciudades[m].nombre \neq ciudades[n].nombre)\}$

`asegura` $\{res = \sum_{i=0}^{|ciudades|-1} ciudades[i].habitantes\}$

$S_1 : res := 0$
$S_2 : i := 0$
while (i < ciudades.length) do
$S_3 : res := res + ciudades[i].habitantes$
$S_4 : i := i + 1$
endwhile

$P \equiv A \wedge B \wedge C$
$A \equiv (\exists l : \mathbb{Z})(0 \leq l < |ciudades| \wedge_L ciudades[l].habitantes > 50000)$
$B \equiv (\forall k : \mathbb{Z})(0 \leq k < |ciudades| \longrightarrow_L ciudades[k].habitantes \geq 0)$
$C \equiv (\forall m : \mathbb{Z})(\forall n : \mathbb{Z})(0 \leq m < n < |ciudades| \longrightarrow_L ciudades[m].nombre \neq ciudades[n].nombre)$

$Pc \equiv res = 0 \ \wedge i = 0 \ \wedge A \wedge B \wedge C$

$B \equiv i < |ciudades|$

$I \equiv 0 \leq i \leq |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes$

$f_v \equiv |ciudades| - i$

Como la última instrucción del programa es el ciclo, podemos asumir que

$Q \equiv Qc \equiv res = \sum_{j=0}^{|ciudades|-1} ciudades[j].habitantes$

Por corolario de la propiedad de monotonía de las WP, para demostrar la correctitud del programa basta ver que son válidas las siguientes triplas de Hoare:

$$\{Pc\}S_c\{Qc \equiv Q\} \; y \; \{P\}S_1; S_2\{Pc\}$$

2.1.1. $Pc \implies wp(S_3; S_4, Qc)$ (Teorema del Invariante y Terminación)

2.1.1.1 $Pc \implies I$

$Pc \equiv res = 0 \ \wedge i = 0 \ \wedge A \wedge B \wedge C$

Reemplazando en I y asumiendo el antecedente cómo verdadero:

$$0 \leq 0 \leq |ciudades| \wedge_L 0 = \sum_{j=0}^{-1} ciudades[j].habitantes \equiv I$$
$$0 \leq |ciudades| \wedge_L 0 = 0 \equiv I$$
$$True \ \wedge_L \ True \equiv I$$
$$True \equiv I$$
$$Pc \implies I$$

2.1.1.2 $I \wedge B \implies wp(S_3; S_4, I)$

$wp(S_3; S_4, I) \equiv wp(res = res + ciudades[i].habitantes, \ wp(i = i + 1, I)) \equiv$

$wp(res = res + ciudades[i].habitantes, \ def(i + 1) \wedge_L I_{i+1}^{i}) \equiv$

$wp(res = res + ciudades[i].habitantes, \ True \wedge_L 0 \leq i + 1 \leq |ciudades| \wedge_L res = \sum_{j=0}^{i} ciudades[j].habitantes) \equiv$

$$wp(res = res + ciudades[i].habitantes, \ \underbrace{-1 \leq i \leq |ciudades| - 1 \wedge_L res = \sum_{j=0}^{i} ciudades[j].habitantes)}_{I'} \equiv$$

$def(res + ciudades[i].habitantes) \wedge_L (I')_{res+ciudades.[i].habitantes}^{res} \equiv$

$0 \leq i < |ciudades| \wedge_L$ -1 $\leq i \leq |ciudades| - 1 \wedge_L res + ciudades[i].habitantes = \sum_{j=0}^{i} ciudades[j].habitantes \equiv$

$0 \leq i < |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes$

$I \wedge B \equiv 0 \leq i \leq |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge i < |ciudades| \equiv$

$$0 \leq i < |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \equiv wp(S_3; S_4, I)$$

Cómo asumimos el antecedente como verdadero:

$$True \equiv wp(S_3; S_4, I)$$
$$I \wedge B \implies wp(S_3; S_4, I)$$

2.1.1.3 $I \wedge \neg B \implies Qc$

$I \wedge \neg B \equiv 0 \leq i \leq |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge i \geq |ciudades| \equiv$

$$i = |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \implies$$
$$res = \sum_{j=0}^{|ciudades|-1} ciudades[j].habitantes \equiv Qc$$

Cómo asumimos el antecedente como verdadero:

$$True \equiv Qc$$
$$I \wedge \neg B \implies Qc$$

2.1.1.4 $I \wedge B \wedge v_0 = f_v \implies wp(S_3; S_4, f_v < v_0)$

$wp(S_3; S_4, f_v < v_0) \equiv wp(res = res + ciudades[i].habitantes, wp(i = i + 1, |ciudades| - i < v_0)) \equiv$

$wp(res = res + ciudades[i].habitantes, def(i + 1) \wedge_L (|ciudades| - i < v_0)^i_{i+1}) \equiv$

$wp(res = res + ciudades[i].habitantes, True \wedge_L |ciudades| - i - 1 < v_0) \equiv$

$def(res + ciudades[i].habitantes) \wedge_L (|ciudades| - i - 1 < v_0)^{res}_{res+ciudades[i].habitantes} \equiv$

$0 \leq i < |ciudades| \wedge_L |ciudades| - i - 1 < v_0$

$I \wedge B \wedge v_0 = f_v \equiv 0 \leq i \leq |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge i < |ciudades| \wedge v_0 = |ciudades| - i \equiv$

$$0 \leq i < |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge v_0 = |ciudades| - i$$

Reemplazando en $wp(S_3; S_4, f_v < v_0)$ y asumiendo el antecedente cómo verdadero:

$$0 \leq i < |ciudades| \wedge_L |ciudades| - i - 1 < |ciudades| - i \equiv wp(S_3; S_4, f_v < v_0)$$
$$True \wedge_L True \equiv wp(S_3; S_4, f_v < v_0)$$
$$True \equiv wp(S_3; S_4, f_v < v_0)$$
$$I \wedge B \wedge v_0 = f_v \implies wp(S_3; S_4, f_v < v_0)$$

2.1.1.5 $I \wedge f_v \leq 0 \implies \neg B$

$I \wedge f_v \leq 0 \equiv 0 \leq i \leq |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge |ciudades| - i \leq 0 \equiv$

$$0 \leq i \leq |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge |ciudades| \leq i \equiv$$
$$i = |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \implies$$
$$i \geq |ciudades| \equiv \neg B$$

Cómo asumimos el antecedente como verdadero:

$$True \equiv \neg B$$
$$I \wedge f_v \leq 0 \implies \neg B$$

2.1.2. $P \implies wp(S_1; S_2, Pc)$

$wp(S_1; S_2, Pc) \equiv wp(res = 0, wp(i = 0, Pc)) \equiv wp(res = 0, def(0) \wedge_L Pc^i_0) \equiv$

$wp(res = 0, True \wedge_L res = 0 \wedge 0 = 0 \wedge A \wedge B \wedge C) \equiv$

$def(0) \wedge_L (res = 0 \wedge True \wedge A \wedge B \wedge C)^{res}_0 \equiv$

$True \wedge_L 0 = 0 \wedge A \wedge B \wedge C \equiv True \wedge A \wedge B \wedge C \equiv A \wedge B \wedge C$

$P \equiv A \wedge B \wedge C \equiv wp(S_1; S_2, Pc)$

Cómo asumimos el antecedente como verdadero:

$$True \wedge True \wedge True \equiv wp(S_1; S_2, Pc)$$
$$True \equiv wp(S_1; S_2, Pc)$$
$$P \implies wp(S_1; S_2, Pc)$$

El programa es correcto con respecto a su especificación.

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

Informalmente, a partir de la precondición del programa se puede asumir que existe algún elemento de la secuencia de entrada con más de 50.000 habitantes. Como el ciclo, mediante el paso de las iteraciones, suma los habitantes de cada ciudad de la entrada, no hay ciudades con un numero negativo de habitantes y el ciclo finaliza, se puede concluir que la suma total de habitantes dará como resultado al menos 50.000, cumpliendo así lo pedido.

Formalmente, para demostrar que el valor devuelto es mayor a 50.000, consideramos la siguiente postcondición Q y vemos si el programa sigue siendo correcto con respecto a su especificación.

Tomando P, B, Pc, f_v igual que en el punto anterior, y

$Q \equiv Qc \equiv res = \sum_{j=0}^{|ciudades|-1} ciudades[j].habitantes \wedge res > 50000$

$A \equiv (\exists l : \mathbb{Z})(0 \leq l < |ciudades| \wedge_L ciudades[l].habitantes > 50000)$

$B \equiv (\forall k : \mathbb{Z})(0 \leq k < |ciudades| \longrightarrow_L ciudades[k].habitantes \geq 0)$

$I \equiv 0 \leq i \leq |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge A \wedge B$

Para demostrar la correctitud del programa basta ver que son válidas las siguientes triplas de Hoare:

$$\{Pc\}S_c\{Qc \equiv Q\}\ y\ \{P\}S_1; S_2\{Pc\}$$

2.2.1. $Pc \implies wp(S_3; S_4, Qc)$ (Teorema del Invariante y Terminación)

2.2.1.1 $Pc \implies I$

$Pc \equiv res = 0\ \wedge i = 0\ \wedge A \wedge B \wedge C$

Reemplazando en I y asumiendo el antecedente cómo verdadero:

$$0 \leq 0 \leq |ciudades| \wedge_L 0 = \sum_{j=0}^{-1} ciudades[j].habitantes \wedge A \wedge B \equiv I$$
$$0 \leq |ciudades| \wedge_L 0 = 0 \wedge A \wedge B \equiv I$$
$$True\ \wedge_L\ True \wedge True \wedge True \equiv I$$
$$True \equiv I$$
$$Pc \implies I$$

2.2.1.2 $I \wedge B \implies wp(S_3; S_4, I)$

$wp(S_3; S_4, I) \equiv wp(res = res + ciudades[i].habitantes,\ wp(i = i + 1, I)) \equiv$

$wp(res = res + ciudades[i].habitantes,\ def(i + 1) \wedge_L I^i_{i+1}) \equiv$

$wp(res = res + ciudades[i].habitantes,\ True \wedge_L 0 \leq i + 1 \leq |ciudades| \wedge_L res = \sum_{j=0}^{i} ciudades[j].habitantes \wedge A \wedge B) \equiv$

$$wp(res = res + ciudades[i].habitantes,\ \underbrace{-1 \leq i \leq |ciudades| - 1 \wedge_L res = \sum_{j=0}^{i} ciudades[j].habitantes \wedge A \wedge B}_{I'}) \equiv$$

$def(res + ciudades[i].habitantes) \wedge_L (I')^{res}_{res+ciudades.[i].habitantes} \equiv$

$0 \leq i < |ciudades|\ \wedge_L -1 \leq i \leq |ciudades| - 1 \wedge_L res + ciudades[i].habitantes = \sum_{j=0}^{i} ciudades[j].habitantes \wedge A \wedge B \equiv$

$0 \leq i < |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge A \wedge B$

$I \wedge B \equiv 0 \leq i \leq |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge A \wedge B \wedge i < |ciudades| \equiv$

$$0 \leq i < |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge A \wedge B \equiv wp(S_3; S_4, I)$$

Cómo asumimos el antecedente como verdadero:

$$True \equiv wp(S_3; S_4, I)$$
$$I \wedge B \implies wp(S_3; S_4, I)$$

2.2.1.3 $I \wedge \neg B \implies Qc$

$I \wedge \neg B \equiv 0 \leq i \leq |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge A \wedge B \wedge i \geq |ciudades| \equiv$

$$i = |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge A \wedge B \implies$$
$$res = \sum_{j=0}^{|ciudades|-1} ciudades[j].habitantes \wedge A \wedge B$$

Cómo sabemos que A nos asegura que haya al menos un elemento de ciudades mayor a 50000, B nos asegura que todos los elementos de ciudades son no negativos y asumimos el antecedente como verdadero, se tiene que:

$$res = \sum_{j=0}^{|ciudades|-1} ciudades[j].habitantes \wedge A \wedge B \implies$$
$$res = \sum_{j=0}^{|ciudades|-1} ciudades[j].habitantes \wedge res > 50000 \equiv Qc$$
$$True \wedge True \equiv Qc$$
$$True \equiv Qc$$
$$I \wedge \neg B \implies Qc$$

2.2.1.4 $I \wedge B \wedge v_0 = f_v \implies wp(S_3; S_4, f_v < v_0)$

$wp(S_3; S_4, f_v < v_0) \equiv wp(res = res + ciudades[i].habitantes, wp(i = i + 1, |ciudades| - i < v_0)) \equiv$

$wp(res = res + ciudades[i].habitantes, def(i + 1) \wedge_L (|ciudades| - i < v_0)^{i}_{i+1}) \equiv$

$wp(res = res + ciudades[i].habitantes, True \wedge_L |ciudades| - i - 1 < v_0) \equiv$

$def(res + ciudades[i].habitantes) \wedge_L (|ciudades| - i - 1 < v_0)^{res}_{res+ciudades[i].habitantes} \equiv$

$0 \leq i < |ciudades| \wedge_L |ciudades| - i - 1 < v_0 \equiv$

$I \wedge B \wedge v_0 = f_v \equiv 0 \leq i \leq |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge A \wedge B \wedge i < |ciudades| \wedge v_0 = |ciudades| - i$

$$\equiv 0 \leq i < |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge A \wedge B \wedge v_0 = |ciudades| - i$$

Reemplazando en $wp(S_3; S_4, f_v < v_0)$ y asumiendo el antecedente cómo verdadero:

$$0 \leq i < |ciudades| \wedge_L |ciudades| - i - 1 < |ciudades| - i \equiv wp(S_3; S_4, f_v < v_0)$$
$$True \wedge_L True \equiv wp(S_3; S_4, f_v < v_0)$$
$$True \equiv wp(S_3; S_4, f_v < v_0)$$
$$I \wedge B \wedge v_0 = f_v \implies wp(S_3; S_4, f_v < v_0)$$

2.2.1.5 $I \wedge f_v \leq 0 \implies \neg B$

$I \wedge f_v \leq 0 \equiv 0 \leq i \leq |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge A \wedge B \wedge |ciudades| - i \leq 0 \equiv$

$$0 \leq i \leq |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge A \wedge B \wedge |ciudades| \leq i \equiv$$
$$i = |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge A \wedge B \implies$$
$$i \geq |ciudades| \equiv \neg B$$

Cómo asumimos el antecedente como verdadero:

$$True \equiv \neg B$$
$$I \wedge f_v \leq 0 \implies \neg B$$

2.2.2. $P \implies wp(S_1; S_2, Pc)$

$wp(S_1; S_2, Pc) \equiv wp(res = 0, wp(i = 0, Pc)) \equiv wp(res = 0, def(0) \wedge_L Pc^i_0) \equiv$

$wp(res = 0, True \wedge_L res = 0 \wedge 0 = 0 \wedge A \wedge B \wedge C) \equiv$

$def(0) \wedge_L (res = 0 \wedge True \wedge A \wedge B \wedge C)^{res}_0 \equiv$

$True \wedge_L 0 = 0 \wedge A \wedge B \wedge C \equiv True \wedge A \wedge B \wedge C \equiv A \wedge B \wedge C$

$P \equiv A \wedge B \wedge C \equiv wp(S_1; S_2, Pc)$

Cómo asumimos el antecedente como verdadero:

$$True \wedge True \wedge True \equiv wp(S_1; S_2, Pc)$$
$$True \equiv wp(S_1; S_2, Pc)$$
$$P \implies wp(S_1; S_2, Pc)$$

El programa es correcto con respecto a su especificación y devuelve un resultado mayor a 50.000.