DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Trabajo práctico 1: Especificacíon y WP

Elecciones Nacionales

17 de octubre de 2023 — Algoritmos y Estructuras de Datos

sudo_rm-rf_/*

Integrante	LU	Correo electrónico
Rocca, Santiago	152/23	`santiagrocca17@gmail.com`
Fisz, Maximiliano	586/19	`maximilianofisz@gmail.com`
Gomez, Abril	574/20	`goskema@gmail.com`
López, Gonzalo	1017/22	`gonzalo.esloga.uba@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

1.1. General

1.1.1. Predicados Universales

`pred noHayRepetidos` (in escrutinio : $seq\langle\mathbb{Z}\rangle$) {
$(\forall x : \mathbb{Z})(0 \leq x < |escrutinio| \longrightarrow_L ((\forall y : \mathbb{Z})(0 \leq y < |escrutinio| \wedge \neg(x = y) \longrightarrow_L \neg(escrutinio[x] = escrutinio[y])))$
}
`pred cantVotosValidos` (in escrutinio : $seq\langle\mathbb{Z}\rangle$) {
$((\forall x : \mathbb{Z})(0 \leq x < |escrutinio|) \longrightarrow_L (escrutinio[x] \geq 0))$
}
`pred escrutinioValdio` (in escrutinio: $seq\langle\mathbb{Z}\rangle$) {
$|escrutinio| \geq 2$
}
`pred EleccionValida` (in escrutinio: $seq\langle\mathbb{Z}\rangle$) {
$nohayRepetidos(escrutinio) \wedge cantVotosValidos(escrutinio) \wedge escrutinioValido(escrutinio)$
}
`pred umbralElectoral` (in escrutinioDip : $seq\langle\mathbb{Z}\rangle$) {
$((\exists x : \mathbb{Z})(0 \leq x < |escrutinioDip|) \wedge_L (escrutinioDip[x] > (|escrutinioDip| * 3)/100))$
}
`pred minimoDePartidos` (in escrutinio: $seq\langle\mathbb{Z}\rangle$) {
$|escrutinio| \geq 3$
}

1.1.2. Auxiliares

`aux sumaDeVotos` (in escrutinio : $seq\langle\mathbb{Z}\rangle$) : $\mathbb{Z}$ = $\sum\limits_{i=0}^{|escrutinio|-1} escrutinio[i]$;
`aux porcentajeDeVotos` (in escrutinio: $seq\langle\mathbb{Z}\rangle$, in votosPartido: $\mathbb{Z}$) : $\mathbb{R}$ = $sumaDeVotos(escrutinio)^{-1} * votosPartido * 10^2$);
`aux bancasDe` (in indicePartido: $\mathbb{Z}$, in bancas, in dHondt $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) : $\mathbb{Z}$ =
$\sum\limits_{p=0}^{bancas-1} if\ cocienteGanador(indicePartido, p, dHondt)\ then\ 1\ else\ 0$;
`aux bancasGanadas` (in dH: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$,in max: $\mathbb{Z}$, in r: $\mathbb{Z}$) : $\mathbb{Z}$ =
$\sum_{j=0}^{dH[r]-1} if\ max > MaxAnteriores(dH, dH[r][j])\ then\ 1\ else\ 0$;
`aux MaxAnteriores` (in dH: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$,in max: $\mathbb{Z}$, in r: $\mathbb{Z}$) : $\mathbb{Z}$ = $\sum_{j=0}^{|dH|-1} \sum_{i=0}^{dH[j]-1} if\ dH[j][i] > r\ then\ 1\ else\ 0$;

1.2. hayBallotage

1.2.1. Main

`proc hayBallotage` (in escrutinio : $seq\langle\mathbb{Z}\rangle$) : Bool
`requiere` $\{eleccionValida(escrutinio))\}$
`asegura` $\{res = True \longleftrightarrow \neg((partidoMayorA45\%(escrutinio)) \vee (partidoMayorA40\%ConDiferencia(escrutinio)))\}$

1.2.2. Predicados Especificos

`pred partidoMayorA40%ConDiferencia` (in escrutinio : $seq\langle\mathbb{Z}\rangle$) {
$(\exists n : \mathbb{Z})(0 \leq n < |escrutinio| - 1 \wedge_L (porcentajeDeVotos(escrutinio, escrutinio[n]) > 40) \wedge_L$
$\neg(\forall x : \mathbb{Z})(0 \leq x < |escrutinio| - 1 \wedge (\neg(n = x) \longrightarrow_L ((escrutinio[n] - escrutinio[x]) > 10))$
}

1.3. hayFraude

1.3.1. Main

`proc hayFraude` (in escrutinio_Presidente: $seq\langle\mathbb{Z}\rangle$, in escrutinio_Senadores: $seq\langle\mathbb{Z}\rangle$, in escrutinio_Diputados: $seq\langle\mathbb{Z}\rangle$) : Bool
`requiere` $\{eleccionValida(escrutinio\_Presidente) \wedge eleccionValida(escrutinio\_Senadores) \wedge eleccionValida(escrutinio\_diputados) \wedge minimoDePartidos(escrutinio\_Senadores) \wedge (|escrutinio\_Presidente| = |escrutinio\_Senadores| = |escrutinio\_Diputados|)\}$
`asegura` $\{res = \neg(((sumaDeVotos(escrutinio\_Presidente) = sumaDeVotos(escrutinio\_Senadores)) \wedge (sumaDeVotos(escrutinio\_Presidente) = sumaDeVotos(escrutinio\_Diputados)))\}$

1.4. obtenerSenadoresEnProvincia

1.4.1. Main

`proc obtenerSenadoresEnProvincia` (in escrutinio : $seq\langle\mathbb{Z}\rangle$) : $\mathbb{Z}\times\mathbb{Z}$

`requiere` $\{eleccionValida(escrutinio) \land minimoDePartidos(escrutinio)\}$

`asegura` $\{(\exists! x:\mathbb{Z})(0 \leq x < |escrutinio| - 1 \land_L ((\exists! y:\mathbb{Z})(0 \leq y < |escrutinio| - 1 \land_L ((\forall i:\mathbb{Z})(0 \leq i < |escrutinio| - 1 \land \neg(i = x) \land \neg(i = y) \longrightarrow_L (escrutinio[i] < escrutinio[y] < escrutinio[x] \land res_0 = x \land res_1 = y)))))\}$

1.4.2. Predicados Especificos

1.5. calcularDHondtEnProvincia

1.5.1. Main

`proc calcularDHondtEnProvincia` (in cant_bancas: $\mathbb{Z}$, in escrutinio: $seq\langle\mathbb{Z}\rangle$) : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$

`requiere` $\{eleccionValida(escrutinio) \land umbralElectoral(escrutinio) \land cant\_bancas > 0)\}$

`asegura` $\{((\forall\, i:\mathbb{Z})(0 \leq i < cant\_bancas) \land_L (\forall\, j:\mathbb{Z})(0 \leq j < |escrutinio|)) \longrightarrow_L (res[j][i] = \frac{escrutinio[j]}{i+1} \land \frac{escrutinio[j]}{i+1} \geq 0)\}$

1.6. obtenerDiputadosEnProvincia

1.6.1. Main

`proc obtenerDiputadosEnProvincia` (in cant_bancas: $\mathbb{Z}$, in escrutinio: $seq\langle\mathbb{Z}\rangle$, in dHondt: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) : $seq\langle\mathbb{Z}\rangle$

`requiere` $\{eleccionValida(escrutinio) \land umbralElectoral(escrutinio) \land coeficientesDistintos(dHondt) \land esMatriz(dHondt) \land matrizDelEscrutinio(dHondt,\ cant\_bancas,\ escrutinio)\}$

`asegura` $\{(\forall\, r:\mathbb{Z})(0 \leq r < |escrutinio| - 1 \longrightarrow_L res[r] = bancasGanadas(dHondt, cant\_bancas, r)) \land |res| = |dHondt|\}$

1.6.2. Predicados Especificos

`pred esMatriz` (in dH: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {

$True \longleftrightarrow (\forall i:\mathbb{Z})(0 \leq i < |dH| - 1 \longrightarrow |dH[i]| = |dH[i+1]|)$

}

`pred matrizDelEscrutinio` (in dH: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, in cant_bancas: $\mathbb{Z}$, in escrutinio: $seq\langle\mathbb{Z}\rangle$) {

$True \longleftrightarrow ((\forall\, i:\mathbb{Z})(0 \leq i < cant\_bancas) \land_L (\forall\, j:\mathbb{Z})(0 \leq j < |escrutinio|)) \longrightarrow_L dH[j][i] = \frac{escrutinio[j]}{i+1})$

}

`pred coeficientesDistintos` (in DHondt: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {

$(\forall j:\mathbb{Z})(0 \leq j < |DHondt| \longrightarrow_L ((\forall i:\mathbb{Z})(0 \leq i < |DHondt[j]| \longrightarrow_L \neg((\exists z:\mathbb{Z})(0 \leq z < |DHondt| \land (z \neq j) \land_L ((\exists t:\mathbb{Z})(0 \leq t < |DHondt[z]| \land (t \neq i) \land_L DHondt[z][t] = DHondt[j][i]))))))$

}

`pred todosPositivos` (in DHondt : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {

$(\forall j:\mathbb{Z})(0 \leq j < |DHondt| \longrightarrow_L ((\forall i:\mathbb{Z})(0 \leq i < |DHondt[j]| \longrightarrow_L DHondt[j][i] > 0)))$

}

1.7. validarListasDiputadosEnProvincia

1.7.1. Main

`proc (in cant_bancas:` $\mathbb{Z}$`, in listas:` $seq\langle seq\langle dni:\mathbb{Z}\times genero:\mathbb{Z}\rangle\rangle$ (Bool) :

`requiere` $\{(cant\_bancas > 0) \land (\forall\, x:\mathbb{Z})(0 \leq x < |listas| \longrightarrow_L listas[x]_0 > 0 \land 1 \leq listas[x]_1 \leq 2)\}$

`asegura` $\{(\forall\ partido:\mathbb{Z})(0 \leq partido < |listas|) \longrightarrow_L (cantCandidatosCorrecta(cant\_bancas,\ listas[partido]) \land altGenero(listas[partido])\}$

1.7.2. Predicados Especificos

`pred cantCandidatosCorrecta` (cant_bancas: $\mathbb{Z}$, partido: $seq\langle dni:\mathbb{Z}\times genero:\mathbb{Z}\rangle$) {

$cant\_bancas = |partido|$

}

`pred altGenero` (partido: $seq\langle dni:\mathbb{Z}\times genero:\mathbb{Z}\rangle$) {

$(((\forall n:\mathbb{Z})(0 \leq n < |partido|)) \longrightarrow_L (((n \mod 2 = 0) \longrightarrow_L (partido[n]_1 = 1)) \land_L ((n \mod 2 = 1) \longrightarrow_L (partido[n]_1 = 2)) \lor_L ((n \mod 2 = 0) \longrightarrow_L (partido[n]_1 = 2 \land_L (n \mod 2 = 1) \longrightarrow_L partido[n]_1 = 1))$

}

2. Implementaciones y demostraciones de correctitud

2.1. Implementaciones

2.1.1. hayBallotage

```
res := true
trans := 0
primero := 0
segundo := 0
i := 0
suma := 0
while (escrutinio.size() > i) do
    suma:= suma + escrutinio[i]
    i := i + 1
endwhile
i := 0
while (escrutinio.size() > i) do
    escrutinio[i] := (escrutinio[i] * 100)/suma
    i := i + 1
endwhile
i := 0
while (escrutinio.size() > i) do
    if (segundo < escrutinio[i])
        segundo := escrutinio[i]
    else:
        skip
    endif
    if (primero < segundo)
        trans := primero
        primero := segundo
        segundo := trans
    else:
        skip
    endif
    i := i + 1
endwhile
if (primero > 45)
    res := false
else
    if ((primero > 40) && (primero − segundo >= 10))
        res := false
    else
        skip
    endif
endif
```

2.1.2. hayFraude

```
res := false
i := 0
SumaSen := 0
sumaDip := 0
sumaPres := 0
while (escrutinio_Presidente.size() > i) do
    sumaPres := sumaPres + escrutinio_Presidente[i]
    sumaDip := sumaDip + escrutinio_Diputados[i]
    sumaSen := sumaSen + escrutinio_Senadoresl[i]
    i := i + 1
endwhile
if (sumaPres = sumaDip && sumaPres = sumaSen) then
    skip
else:
    res := true
endif
```

2.1.3. obtenerSenadoresEnProvincia

```
trans := 0
if (escrutinio[0] > escrutinio[1]):
        primero := 0
        segundo := 1
else:
        primero := 1
        segundo := 0
endif
i := 2
while (escrutinio.size() > i) do
    if (escrutinio[segundo] < escrutinio[i])
        segundo := i
    else:
        skip
    endif
    if (escrutinio[primero] < escrutinio[segundo])
        trans := primero
        primero := segundo
        segundo :=  trans
    else:
        skip
    endif
    i := i + 1
endwhile
res := (primero,segundo)
```

2.1.4. validarListasDiputadosEnProvincia

```
res := true
i := 0
while (listas.size() > i) do
    if (listas[i].size() != cant_bancas)
        res:= false
    else:
        skip
    endif
    i := i + 1
endwhile
i := 0
while (listas.size() > i) do
    j := 1
    genero := listas[i][0][1]
    while (listas[i].size() > j) do
        if (listas[i][j][1] == genero)
            res:=false
        else:
            genero := listas[i][j][1]
            j := j + 1
        endif
    endwhile
    i := i + 1
endwhile
```

2.2. Demostraciones de correctitud

2.2.1. hayFraude

- e1 : escrutinioPresidente, e2 : escrutinioSenadores, e3 : escrutinioDiputados
- $P_c : res = False \wedge i = 0 \wedge sumaPres = 0 \wedge sumaDip = 0 \wedge sumaSen = 0$
- $Q_c : \sum_{i=0}^{|e1|-1} e1[i] = sumaPres \wedge \sum_{i=0}^{|e2|-1} e2[i] = sumaSen \wedge \sum_{i=0}^{|e3|-1} e3[i] = sumaDip \wedge$ $(res = False \longleftrightarrow sumaPres = SumaDip \wedge sumaPres = sumaSen)$
- $B : |e1| > i$
- $F_v : |e1| - i$
- $Post : res = \neg(sumaDeVotos(e1) = sumaDeVotos(e3) \wedge sumaDeVotos(e1) = sumaDeVotos(e2))$
- $I : 0 \leq i \leq |e1| \wedge (\sum_{j=0}^{i-1} e1[j] = sumaPres \wedge \sum_{j=0}^{i-1} e2[j] = sumaSen \wedge \sum_{j=0}^{i-1} e3[j] = sumaDip) \wedge$ $(sumaPres = sumaDip \wedge sumaPres = sumaSen \longleftrightarrow res = False)$

1. Pre $\longrightarrow_L$ wp(sumaPres := 0; sumaDip := 0; sumaSen := 0; i := 0; res := False, Pc)

$$wp(sumaDip := 0,\ wp(sumaDip := 0,\ wp(sumaSen := 0,\ wp(i := 0,\ wp(res := False,\ P_c))))$$

$$wp(res := False, P_c) \equiv def(False) \wedge_L Pc^{res}_{False} \equiv$$

$$\equiv True \wedge_L (False = False \wedge i = 0 \wedge sumaPres = 0 \wedge sumaDip = 0 \wedge sumaSen = 0)$$

$$\equiv i = 0 \wedge sumaPres = 0 \wedge sumaDip = 0 \wedge sumaSen = 0 \equiv p0$$

$$wp(i := 0, p0) \equiv def(0) \wedge_L p0^i_0 \equiv$$

$$\equiv True \wedge_L 0 = 0 \wedge sumaPres = 0 \wedge sumaDip = 0 \wedge sumaSen = 0$$

$$\equiv sumaPres = 0 \wedge sumaDip = 0 \wedge sumaSen = 0 \equiv p1$$

$$wp(sumaSen := 0,\ p1) \equiv def(0) \wedge_L p1^{sumaSen}_0$$

$$\equiv True \wedge_L sumaPres = 0 \wedge sumaDip = 0 \wedge 0 = 0$$

$\equiv sumaPres = 0 \wedge sumaDip = 0 \equiv p2$

$wp(sumaDip := 0,\ p2) \equiv def(0) \wedge_L p2_0^{sumaDip}$

$\equiv True \wedge_L sumaPres = 0 \wedge 0 = 0$

$\equiv sumaPres = 0 \equiv p3$

$wp(sumaPres := 0, p3) \equiv def(0) \wedge_L p3_0^{sumaPres}$

$\equiv True \wedge_L 0 = 0$

$\equiv True$

Pre $\longrightarrow_L$ wp(sumaPres := 0; sumaDip := 0; sumaSen := 0; i :=0; res := False, Pc)
es verdadero porque Pre siempre implica a True

2. $Q_c \longrightarrow_L$ wp(If(...), $Post$)

$wp(\text{If}(\ldots), Post) \equiv$

$B : sumaPres = sumaDip \wedge sumaPres = sumaSen$

$\equiv def(B) \wedge_L ((B \wedge wp(Skip,\ Post) \vee (\neg B \wedge wp(res := True,\ Post))))$

$\equiv True \wedge_L ((B \wedge wp(Skip,\ Post) \vee (\neg B \wedge wp(res := True,\ Post))))$

$wp(Skip,\ Post) \equiv Post$

$wp(res := True,\ Post) \equiv$

$\equiv def(True) \wedge_v Post_{True}^{res}$

$\equiv True \wedge_v Post_{True}^{res}$

$\equiv True = \neg(sumaDeVotos(e1) = sumaDeVotos(e3)\ \wedge sumaDeVotos(e1) = sumaDeVotos(e2))$

$\equiv True = \neg(\sum_{i=0}^{|e1|-1} e1[i] = \sum_{i=0}^{|e3|-1} e3[i]\ \wedge \sum_{i=0}^{|e1|-1} e1[i] = \sum_{i=0}^{|e2|-1} e2[i]) \equiv C$

$\equiv True \wedge_L ((B \wedge Post) \vee (\neg B \wedge wp(res := True,\ Post))) \equiv ((B \wedge Post) \vee (\neg B \wedge C)) \equiv$

$\equiv ((sumaPres = sumaDip \wedge sumaPres = sumaSen \wedge Post) \vee$
$(\neg(sumaPres = sumaDip \wedge sumaPres = sumaSen) \wedge C))$

$wp(\text{If}(\ldots), Post) \equiv ((sumaPres = sumaDip \wedge sumaPres = sumaSen \wedge Post) \vee$
$(\neg(sumaPres = sumaDip \wedge sumaPres = sumaSen) \wedge C))$

Recordatorio de Q_C : $\sum_{i=0}^{|e1|-1} e1[i] = sumaPres \wedge \sum_{i=0}^{|e2|-1} e2[i] = sumaSen \wedge \sum_{i=0}^{|e3|-1} e3[i] = sumaDip \wedge$
$(res = False \longleftrightarrow sumaPres = SumaDip \wedge sumaPres = sumaSen)$

Veamos si $Q_c \longrightarrow_L$ wp(If(...), $Post$) por partes. Asumimos Q_c verdadero. Vemos que nos basta con probar una de las ramas del wp, que se separa en dos casos: cuando se cumple B o $\neg B$. Sabemos que $(res = False \longleftrightarrow sumaPres = SumaDip \wedge sumaPres = sumaSen)$ entonces sumaPres = SumaDip, sumaPres = sumaSen y res = False.
Si probamos $Q_C \longrightarrow_L Post$, queda probada la implicación porque es verdadera la rama B del wp.
Sabemos que Post $\equiv res = \neg(\sum_{i=0}^{|e1|-1} e1[i] = \sum_{i=0}^{|e3|-1} e3[i]\ \wedge \sum_{i=0}^{|e1|-1} e1[i] = \sum_{i=0}^{|e2|-1} e2[i])$

Las sumatorias, sabemos por Q_c, que son iguales a la variable con el resultado de la suma, por ejemplo: $\sum_{i=0}^{|e1|-1} e1[i] =$ sumaPres y que sumaPres = sumaDip y sumaPres = sumaSen. Ademas, sabemos que res = False. Entonces:

$$Post \equiv res = \neg(\sum_{i=0}^{|e1|-1} e1[i] = \sum_{i=0}^{|e3|-1} e3[i]\ \wedge \sum_{i=0}^{|e1|-1} e1[i] = \sum_{i=0}^{|e2|-1} e2[i])$$
$$Post \equiv False = \neg(sumaPres = sumaDip \wedge sumaPres = sumaSen)$$
$$Post \equiv False = \neg(True \wedge True)$$
$$Post \equiv False = False$$
$$Post \equiv True$$

Entonces $Q_c \longrightarrow_L$ wp(If(...), $Post$)

3. $P_c \longrightarrow_L$ wp(While(...), Qc) mediante el teorema del invariante.

e1 : escrutinioPresidente, e2 : escrutinioSenadores, e3 : escrutinioDiputados

a) $P_c \longrightarrow I$

$P_c : res = False \wedge i = 0 \wedge sumaPres = 0 \wedge sumaDip = 0 \wedge sumaSen = 0$

$I : 0 \leq i \leq |e1| \wedge (\sum_{j=0}^{i-1} e1[j] = sumaPres \wedge \sum_{j=0}^{i-1} e2[j] = sumaSen \wedge \sum_{j=0}^{i-1} e3[j] = sumaDip) \wedge$
$(sumaPres = sumaDip \wedge sumaPres = sumaSen \longleftrightarrow res = False)$

- Como i = 0, se cumple que $0 \leq$ i $\leq |e1|$
- Como i = 0, las sumatorias suman 0 (suma desde un limite a otro menor) y tenemos:

  $$0 = \text{sumaPres} \wedge 0 = \text{sumaSen} \wedge 0 = \text{sumaDip}$$
- Como las tres variables, sumaPres, sumaDip y sumaSen son iguales a 0, sumaPres = sumaDip y sumaPres = sumaSen. Res tambien es False, por lo que es verdadera la ultima condicion del invariante.

Entonces $P_c \longrightarrow I$

b) $I \wedge \neg B \longrightarrow Q_c$

$I : 0 \leq i \leq |e1| \wedge (\sum_{j=0}^{i-1} e1[j] = sumaPres \wedge \sum_{j=0}^{i-1} e2[j] = sumaSen \wedge \sum_{j=0}^{i-1} e3[j] = sumaDip) \wedge$
$(sumaPres = sumaDip \wedge sumaPres = sumaSen \longleftrightarrow res = False)$

$\neg B : \neg(|e1| > i) \equiv |e1| \leq i$

$Q_c : \sum_{i=0}^{|e1|-1} e1[i] = sumaPres \wedge \sum_{i=0}^{|e2|-1} e2[i] = sumaSen \wedge \sum_{i=0}^{|e3|-1} e3[i] = sumaDip \wedge$
$(res = False \longleftrightarrow sumaPres = SumaDip \wedge sumaPres = sumaSen)$

- Como i $\leq |e1|$ y tambien $|e1| \leq i$ entonces i = $|e1|$. Al reemplazar i por $|e1|$ en las sumatorias de I, se cumplen las igualdades de las sumatorias de Q_c.
- Del I, tenemos inmediatamente la implicacion de la doble igualdad de Q_c.

Entonces $I \wedge \neg B \longrightarrow Q_c$

c) $I \wedge fv \leq 0 \longrightarrow \neg B$

$I : 0 \leq i \leq |e1| \wedge (\sum_{j=0}^{i-1} e1[j] = sumaPres \wedge \sum_{j=0}^{i-1} e2[j] = sumaSen \wedge \sum_{j=0}^{i-1} e3[j] = sumaDip) \wedge$
$(sumaPres = sumaDip \wedge sumaPres = sumaSen \longleftrightarrow res = False)$

$fv \leq 0 : |e1| - i \leq 0$
$\neg B : |e1| \leq i$

- De fv $\leq 0 : |e1| - i \leq 0 \equiv |e1| \leq i$, Que es exactamente $\neg B$

Entonces $I \wedge fv \leq 0 \longrightarrow \neg B$

d) $I \wedge B \longrightarrow wp(sumaPres := sumaPres + e1[i](\ldots);\ i\ :=\ i+1,\ I)$

$I : 0 \leq i \leq |e1| \wedge (\sum_{j=0}^{i-1} e1[j] = sumaPres \wedge \sum_{j=0}^{i-1} e2[j] = sumaSen \wedge \sum_{j=0}^{i-1} e3[j] = sumaDip) \wedge$
$(sumaPres = sumaDip \wedge sumaPres = sumaSen \longleftrightarrow res = False)$
$B : |e1| > i$
$wp(sumaPres := sumaPres + e1[i](\ldots);\ i\ :=\ i+1,\ I) \equiv$

wp(sumaPres := sumaPres + e1[i], wp(sumaDip := sumaDip + e3[i],
wp(sumaSen := sumaSen + e2[i], wp(i := i + 1, I))))

wp(i := i + 1, I) $\equiv def(i+1)\ \wedge_L I^i_{i+1} \equiv$
$\equiv True\ \wedge_L I^i_{i+1}$
$\equiv 0 \leq i+1 \leq |e1| \wedge (\sum_{j=0}^{i} e1[j] = sumaPres \wedge \sum_{j=0}^{i} e2[j] = sumaSen \wedge \sum_{j=0}^{i} e3[j] = sumaDip) \wedge$
$(sumaPres = sumaDip \wedge sumaPres = sumaSen \longleftrightarrow res = False) \equiv wp1$

wp(sumaSen := sumaSen + e2[i], wp1) $\equiv def(sumaSen + e2[i]) \wedge_L wp1^{sumaSen}_{sumaSen+e2[i]}$
$\equiv True\ \wedge_L wp1^{sumaSen}_{sumaSen+e2[i]}$
$\equiv 0 \leq i+1 \leq |e1| \wedge (\sum_{j=0}^{i} e1[j] = sumaPres \wedge \sum_{j=0}^{i} e2[j] = sumaSen + e2[i] \wedge \sum_{j=0}^{i} e3[j] = sumaDip) \wedge$
$(sumaPres = sumaDip \wedge sumaPres = sumaSen + e2[i] \longleftrightarrow res = False) \equiv wp2$

wp(sumaDip := sumaDip + e3[i], wp2) $\equiv def(sumaDip + e3[i]) \wedge_L wp2^{sumaDip}_{sumaDip+e3[i]}$

$\equiv True \wedge_L wp2^{sumaDip}_{sumaDip+e3[i]}$

$\equiv 0 \leq i+1 \leq |e1| \wedge (\sum_{j=0}^{i} e1[j] = sumaPres \wedge \sum_{j=0}^{i} e2[j] = sumaSen + e2[i] \wedge \sum_{j=0}^{i} e3[j] = sumaDip + e3[i]) \wedge$
$(sumaPres = sumaDip + e3[i] \wedge sumaPres = sumaSen + e2[i] \longleftrightarrow res = False) \equiv wp3$

wp(sumaPres := sumaPres + e1[i], wp3) $\equiv def(sumaDip + e3[i]) \wedge_L wp3^{sumaPres}_{sumaPres+e1[i]}$

$\equiv True \wedge_L wp3^{sumaPres}_{sumaPres+e1[i]}$

$\equiv 0 \leq i+1 \leq |e1| \wedge (\sum_{j=0}^{i} e1[j] = sumaPres + e1[i] \wedge \sum_{j=0}^{i} e2[j] = sumaSen + e2[i] \wedge \sum_{j=0}^{i} e3[j] = sumaDip + e3[i]) \wedge$
$(sumaPres + e1[i] = sumaDip + e3[i] \wedge sumaPres + e1[i] = sumaSen + e2[i] \longleftrightarrow res = False) \equiv wp4$

Veamos si I $\wedge B \longrightarrow$ wp4

- De $0 \leq i \leq |e1|$ y $|e1| > i$ tenemos que $0 \leq i < |e1|$ que implica a $0 \leq i+1 \leq |e1|$ para todos los valores de i.
- Del I, tenemos res = False, por lo que es verdadero tambien (sumaPres + e1[i] = sumaDip + e3[i] $\wedge$ sumaPres + e1[i] = sumaSen + e2[i] $\longleftrightarrow$ res = False)
- Para probar que las 3 sumatorias son iguales a su respectiva variable más el termino actual del escrutinio correspondiente en i, nos basta con probarla de forma general o probar 1, ya que las otras dos son analogas.

$$\sum_{j=0}^{i} e1[j] = sumaPres + e1[i]$$

$$\sum_{j=0}^{i} e1[j] = sumaPres + e1[i] \equiv e1[0] + e1[1] + ... + e1[i-1] + e1[i] = sumaPres + e1[i]$$

Restamos el termino e1[i] de ambos lados y nos queda:

$$e1[0] + e1[1] + ... + e1[i-1] = sumaPres$$

Del I, sabemos que $\sum_{j=0}^{i-1} e1[j] = sumaPres$, que podemos descomponer y ver que vale:

$$e1[0] + e1[1] + ... + e1[i-1] = sumaPres$$

¡Son identicos! Así que el I me prueba las tres sumatorias del wp.

Entonces $I \wedge B \longrightarrow wp(sumaPres := sumaPres + e1[i](\ldots);\ i\ :=\ i+1,\ I)$

e) $I \wedge B \wedge v_0 = |e1| - i \longrightarrow wp(sumaPres := sumaPres + e1[i](\ldots);\ i\ :=\ i+1,\ |s| - i < v_0)$

$I : 0 \leq i \leq |e1| \wedge (\sum_{j=0}^{i-1} e1[j] = sumaPres \wedge \sum_{j=0}^{i-1} e2[j] = sumaSen \wedge \sum_{j=0}^{i-1} e3[j] = sumaDip) \wedge$
$(sumaPres = sumaDip \wedge sumaPres = sumaSen \longleftrightarrow res = False)$
$B : |e1| > i$
$v_0 = |e1| - i$
$wp(sumaPres := sumaPres + e1[i](\ldots);\ i\ :=\ i+1,\ |e1| - i < v_0) \equiv$

wp(sumaPres := sumaPres + e1[i], wp(sumaDip := sumaDip + e3[i],
wp(sumaSen := sumaSen + e2[i], wp(i := i + 1, $|e1| - i < v_0$))))

wp(i := i + 1, $|e1| - i < v_0$) $\equiv def(i+1) \wedge_L (|e1| - i < v_0)^i_{i+1} \equiv$

$\equiv True \wedge (|e1| - i < v_0)^i_{i+1}$

$\equiv |e1| - (i+1) < v_0$

$\equiv |e1| - i - 1 < v_0 \equiv wp1$

wp(sumaSen := sumaSen + e2[i], $wp1$) $\equiv def(sumaSen + e2[i]) \wedge_L wp1^{sumaSen}_{sumaSen+e2[i]} \equiv$

$\equiv True \wedge_L wp1^{sumaSen}_{sumaSen+e2[i]}$

$\equiv |e1| - i - 1 < v_0 \equiv wp2$

wp(sumaDip := sumaDip + e3[i], $wp2$) $\equiv def(sumaDip + e3[i]) \wedge_L wp2^{sumaDip}_{sumaDip+e3[i]} \equiv$

$\equiv True \wedge_L wp2^{sumaDip}_{sumaDip+e3[i]}$

$\equiv |e1| - i - 1 < v_0 \equiv wp3$

wp(sumaPres := sumaPres + e1[i], $wp3$) $\equiv def(sumaPres + e1[i]) \wedge_L wp3^{sumaPres}_{sumaPres+e1[i]} \equiv$

$\equiv True \wedge_L wp3^{sumaPres}_{sumaPres+e1[i]}$

$\equiv |e1| - i - 1 < v_0 \equiv wp4$

Veamos si I $\wedge B \wedge v_0 = |e1| - i \longrightarrow wp4$

- Queremos probar que $|e1| - i - 1 < v_0$. Sabemos que $v_0 = |e1| - i$. Esta igualdad nos dice que $v_0 = |e1| - i$ y por lo tanto mayor a todas las expresiones menores a $|e1| - i$, como por ejemplo $|e1| - i - 1$, que justamente es la expresion de $wp4$.

Entonces $I \wedge B \wedge v_0 = |e1| - i \longrightarrow wp(sumaPres := sumaPres + e1[i](\ldots);\ i\ :=\ i+1,\ |s| - i < v_0)$

Como probamos:

- Pre $\longrightarrow$ wp(codigo previo al ciclo, P_c)
- $P_c \longrightarrow$ wp(ciclo(por teorema del invariante), Q_c)
- $Q_c \longrightarrow$ wp(codigo posterior al ciclo, Post)

Al probar estas tres cosas, por corolario de monotonía sabemos que Pre $\longrightarrow$ wp(programa completo, Post) y, por lo tanto, el programa es correcto con respecto a la especificación.

2.2.2. obtenerSenadoresProvincia

- S= escrutinio, $1^{\underline{o}}$ = primero, $2^{\underline{o}}$ = segundo, $t^{\underline{o}}$ =trans
- $P_c = \{t = 0 \wedge i = 2 \wedge (1 = 0 \wedge 2 = 1) \vee (= 1 \wedge 2 = 0)\}$
- $Q_c = (\forall j : \mathbb{Z})(0 \leq j < |s| \longrightarrow_L (s[j] \leq s[2] \longrightarrow s[2] < s[1])$
- $B = |S| > i$
- $F_v = |S| - i$
- $I = \{0 \leq i \leq |s| \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i \longrightarrow_L S[1] > S[j]) \wedge (\forall j : \mathbb{Z})(0 \leq j < i \wedge j \neq 1 \longrightarrow_L S[1] > S[j])$
- $Q_c = \{\}$

1. $P_c \longrightarrow I$
   $P_c : t = 0 \wedge i = 2 \wedge (1 = 0 \wedge 2 = 1) \vee (= 1 \wedge segundo = 0)$
   $I : 0 \leq i \leq |s| \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i \longrightarrow_L S[1] > S[j]) \wedge (\forall j : \mathbb{Z})(0 \leq j < i \wedge j \neq 1 \longrightarrow_L S[1] > S[j])$ $Q_c : 0 \leq j < |S|$
   - Como i = 2, se cumple que 0 $\leq$ i $\leq |s|$
   - Procedemos a observar ambos casos de $1^{\underline{o}}$ y $2^{\underline{o}}$ ($t^{\underline{o}}$ no nos importa debido a que no se lo menciona en el I).
     Caso 1: 1°=0 $\wedge$ 2°=1 $\wedge$ i=0

     $$(\forall j : \mathbb{Z})(0 \leq j < 0 \longrightarrow_L S[0] > S[j]) \wedge (\forall j : \mathbb{Z})(0 \leq j < 0 \wedge j \neq 0 \longrightarrow_L S[1] > S[j]) \equiv$$
     $$(\forall j : \mathbb{Z})(False \longrightarrow_L S[0] > S[j]) \wedge (\forall j : \mathbb{Z})(False \wedge j \neq 0 \longrightarrow_L S[1] > S[j]) \equiv$$
     $$True \wedge True \quad \text{(por tabla de verdad de la implicacion)}$$

     Caso 2: 1°=1 $\wedge$ 2°=0 $\wedge$ i=0

     $$(\forall j : \mathbb{Z})(0 \leq j < 0 \longrightarrow_L S[1] > S[j]) \wedge (\forall j : \mathbb{Z})(0 \leq j < 0 \wedge j \neq 1 \longrightarrow_L S[0] > S[j]) \equiv$$
     $$(\forall j : \mathbb{Z})(False \longrightarrow_L S[0] > S[j]) \wedge (\forall j : \mathbb{Z})(False \wedge j \neq 1 \longrightarrow_L S[1] > S[j]) \equiv$$
     $$True \wedge True \text{ (por tabla de verdad de la implicacion)}$$

     Luego como se cumple en ambos casos vale entonces $P_c \longrightarrow I$.

2. $I \wedge \neg B \longrightarrow Q_c$ $I : 0 \leq i \leq |s| \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i \longrightarrow_L S[1] > S[j]) \wedge (\forall j : \mathbb{Z})(0 \leq j < i \wedge j \neq 1 \longrightarrow_L S[1] > S[j])$
   $B : |S| > i$