DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Trabajo práctico 1: Especificacíon y WP

Elecciones Nacionales

2 de noviembre de 2023 — Algoritmos y Estructuras de Datos

sudo_rm-rf_/*

Integrante	LU	Correo electrónico
Rocca, Santiago	152/23	santiagorocca17@gmail.com
Fisz, Maximiliano	586/19	maxifisz@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 escrutinioValido` (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 : $seq\langle\mathbb{Z}\rangle$) {
$(\exists x : \mathbb{Z})(0 \leq x < |escrutinioDip| \wedge_L porcentajeDeVotos(escrutinioDip, escrutinioDip[x]) > 3)$
}
`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 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 Específicos

`pred partidoMayorA45%` (in escrutinio : $seq\langle\mathbb{Z}\rangle$) {
$(\exists n : \mathbb{Z})(0 \leq n < |escrutinio| - 1 \wedge_L porcentajeDeVotos(escrutinio, escrutinio[n]) > 45)$
}
`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

1

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

asegura $\{res = True \longleftrightarrow \neg((sumaDeVotos(escrutinio\_Presidente) = sumaDeVotos(escrutinio\_Senadores)) \land (sumaDeVotos(escrutinio\_Presidente) = sumaDeVotos(escrutinio\_Diputados)))\}$

1.4. obtenerSenadoresEnProvincia

1.4.1. Main

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

requiere $\{nohayRepetidos(escrutinio) \land cantVotosValidos(escrutinio) \land minimoDePartidos(escrutinio)\}$

asegura $\{0 \leq res_1, res_0 < |escrutinio| \land_L (\forall j : \mathbb{Z})(0 \leq j < |escrutinio| \land j \neq res_0 \land j \neq res_1 \longrightarrow_L escrutinio[j] < escrutinio[res_1] < escrutinio[res_0])\}$

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

asegura $\{(\forall\, r : \mathbb{Z})(0 \leq r < |escrutinio| - 1 \longrightarrow_L (((porcentajeDeVotos(escrutinio, escrutinio[r]) > 3) \land (res[r] = bancasGanadas(dHondt, cant\_bancas, r))) \lor ((porcentajeDeVotos(escrutinio, escrutinio[r]) \leq 3) \land res[r] = 0)) \land |res| = |dHondt|\}$

1.6.2. Predicados Específicos

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: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {

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

}

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 Específicos

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

$cant\_bancas = |partido|$

}

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

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

}

2. Implementaciones y demostraciones de correctitud

2.1. Implementaciones

2.1.1. hayBallotage

```
res := true
primero := 0
segundo := 1
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:=2
if (escrutinio[primero]<escrutinio[segundo])
    segundo = 0;
    primero = 1;
else
    skip
endif
while(i<|escrutinio|) do
    if (escrutinio[i]>escrutinio[primero])
        segundo:= primero
        primero:= i
    else
        if (escrutinio[i]>escrutinio[segundo])
            segundo := i
        else
            skip
        endif
    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

```
primero:=0
i:=0
while(i<|escrutinio|) do
    if (escrutinio[i]>escrutinio[primero])
        primero:= i
    else:
        skip
    endif
endwhile
segundo:=0
if (primero==0)
    segundo = 1;
else
    skip
endif
i:=0
while (i<|escrutinio|)
    if (escrutinio[i]>escrutinio[segundo] && i!=primero)
        segundo := i
    else
        skip
    endif
    i:=i+1
endwhile
i:=i+1
res=(primero,segundo)
```

2.1.4. validarListasDiputadosEnProvincia

```
res := true
i := 0
while (listas.size() > i) do
    if (listas[i].size() != cant_bancas)
        res:= false1
    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
- $P_c : res = False \wedge i = 0 \wedge sumaPres = 0 \wedge sumaDip = 0 \wedge sumaSen = 0$
- $Q_c : \sum\limits_{i=0}^{|e1|-1} e1[i] = sumaPres \wedge \sum\limits_{i=0}^{|e2|-1} e2[i] = sumaSen \wedge \sum\limits_{i=0}^{|e3|-1} e3[i] = sumaDip \wedge$ $(res = False \longleftrightarrow sumaPres = SumaDip \wedge sumaPres = sumaSen)$
- $B : |e1| > i$
- $F_v : |e1| - i$
- $Post : res = \neg(sumaDeVotos(e1) = sumaDeVotos(e3) \wedge sumaDeVotos(e1) = sumaDeVotos(e2))$
- $I : 0 \leq i \leq |e1| \wedge_L (\sum\limits_{j=0}^{i-1} e1[j] = sumaPres \wedge \sum\limits_{j=0}^{i-1} e2[j] = sumaSen \wedge \sum\limits_{j=0}^{i-1} e3[j] = sumaDip) \wedge$ $(sumaPres = sumaDip \wedge sumaPres = sumaSen \longleftrightarrow res = False)$

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.
Además, 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 también (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 análogas.

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

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

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 idénticos! 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 expresión 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

Calculamos el WP del código con respecto a su post. Para simplificar la operación
llamaremos a "Escrutinio" como "S", "primero" como " 1' " y "segundo" como " 2' ".
Poseemos 6 operaciones elementales dentro del monotonía, luego por regla de cadena de Wp
(Wp(s1;s2;...;s_n,Q) = Wp(s1,Wp(s2,Wp(...,wp(s_n,Q))))).
Empezaremos desde la última linea de código hasta la primera, tomando como poscondición el Wp de la linea posterior a la linea a evaluar.

$Wp(res := (1', 2'), Pos) \equiv def(1', 2') \wedge_L Pos^{res}_{(1',2')}$
$\equiv 0 \leq 2', 1' < |S| \wedge_L (\forall j : \mathbb{Z})(0 \leq j < |S| \wedge j \neq 1' \wedge j \neq 2' \longrightarrow_L S[j] < S[2'] < S[1'] \equiv A$

Al obtener A, necesitamos calcular el Wp de un ciclo por lo que tenemos que usar el Teorema del invariante.

- $I \equiv 0 \leq i \leq |S| \wedge_L 0 \leq 2', 1' < |S| \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i \wedge j \neq 1' \wedge j \neq 2' \longrightarrow_L S[j] < S[2'] < S[1']$
- $Q_{c_2} \equiv A$
- $P_{c_2} \equiv (2' = 0 \vee 2' = 1) \wedge 0 \leq 1' < |S| \wedge 1' \neq 2' \wedge i = 0 \wedge |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1'])$
- $B \equiv i < |S|$
- $f_v = |S| - i$

1. $\underline{P_c \longrightarrow I}$
   $(2' = 0 \vee 2' = 1) \wedge 0 \leq 1' < |S| \wedge 1' \neq 2' \wedge i = 0 \wedge |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1']) \longrightarrow (0 \leq i \leq |S| \wedge_L 0 \leq 2', 1' < |S| \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i \wedge j \neq 1 \wedge j \neq 2' \longrightarrow_L S[j] < S[2'] < S[1'])$

   - <u>Caso 2'=0</u>
     $(2' = 0 \wedge 0 \leq 1' < |S| \wedge 1' \neq 2' \wedge i = 0 \wedge |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1']) \longrightarrow (0 \leq i \leq |S| \wedge_L 0 \leq 2', 1' < |S| \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i \wedge j \neq 1 \wedge j \neq 2' \longrightarrow_L S[j] < S[2'] < S[1'])$

     Asumo como verdadero el P_c y reemplazo en el invariante.

     $0 \leq 0 < |S| \wedge_L 0 \leq 0 < |S| \wedge True \wedge (\forall j : \mathbb{Z})(0 \leq j < 0 \wedge j \neq 1 \wedge j \neq 0 \longrightarrow_L S[j] < S[0] < S[1']) \equiv$

     $True \wedge_L True \wedge True \wedge (\forall j : \mathbb{Z})(False \wedge j \neq 1 \wedge j \neq 0 \longrightarrow_L S[j] < S[0] < S[1']) \equiv$

     $(\forall j : \mathbb{Z})(False \longrightarrow_L S[j] < S[0] < S[1']) \equiv True$

     Vale para este caso.

   - <u>Caso 2'=1</u>
     $(2' = 1 \wedge 0 \leq 1' < |S| \wedge 1' \neq 2' \wedge i = 0 \wedge |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1']) \longrightarrow (0 \leq i \leq |S| \wedge_L 0 \leq 2', 1' < |S| \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i \wedge j \neq 1 \wedge j \neq 2' \longrightarrow_L S[j] < S[2'] < S[1'])$

     Asumo como verdadero el P_c y reemplazo en el invariante.

     $0 \leq 0 < |S| \wedge_L 0 \leq 1 < |S| \wedge True \wedge (\forall j : \mathbb{Z})(0 \leq j < 0 \wedge j \neq 1 \wedge j \neq 1 \longrightarrow_L S[j] < S[1] < S[1']) \equiv$

     $True \wedge_L True \wedge True \wedge (\forall j : \mathbb{Z})(False \wedge j \neq 1 \wedge j \neq 1 \longrightarrow_L S[j] < S[1] < S[1']) \equiv$

     $(\forall j : \mathbb{Z})(False \longrightarrow_L S[j] < S[1] < S[1']) \equiv True$
     Vale para este caso

Como vale para ambos casos, entonces vale que $P_c \longrightarrow I$

2. $\underline{I \wedge \neg B \longrightarrow Q_c}$
$I \wedge i \geq |S| \equiv 0 \leq i \leq |S| \wedge_L 0 \leq 2`, 1` < |S| \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i \wedge j \neq 1` \wedge j \neq 2` \longrightarrow_L S[j] < S[2`] < S[1`]) \wedge i \geq |S| \equiv$
$i = |S| \wedge_L 0 \leq 2`, 1` < |S| \wedge_L (\forall j : \mathbb{Z})(0 \leq j < |S| \wedge j \neq 1` \wedge j \neq 2` \longrightarrow_L S[j] < S[2`] < S[1`] \equiv Q_c$
Luego obtenemos que:
$Q_c \longrightarrow Q_c \equiv True$
Luego vale que $I \wedge \neg B \longrightarrow Q_c$

3. $\underline{\{I \wedge B\}P\{I\}}$

Para esta prueba queremos ver que $I \wedge B \longrightarrow Wp(P, I)$ siendo P el codigo dentro del primer ciclo. Primero calculamos el wp:

$Wp(P, I) \equiv Wp(if(S[2`] < S[i] \wedge i \neq 1`)then(2` := i)else(Skip), Wp(i := i + 1, I))$

Lo calculamos por partes para simplificar:

$Wp(i := i + 1, I) \equiv def(i + 1) \wedge_L I^i_{i+1} \equiv$

$0 \leq i + 1 \leq |S| \wedge_L 0 \leq 2`, 1` < |S| \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i + 1 \wedge j \neq 1` \wedge j \neq 2` \longrightarrow_L S[j] < S[2`] < S[1`] \equiv$

$-1 \leq i < |S| \wedge_L 0 \leq 2`, 1` < |S| \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i + 1 \wedge j \neq 1` \wedge j \neq 2` \longrightarrow_L S[j] < S[2`] < S[1`] \equiv \color{red}{A}$

Luego resta calcular el $Wp(if(S[2`] < S[i] \wedge i \neq 1`)then(2` := i)else(Skip), Wp(i := i + 1, I), \color{red}{A})$

$Wp(if(S[2`] < S[i] \wedge i \neq 1`)then(2` := i)else(Skip), Wp(i := i + 1, I), \color{red}{A}) \equiv$
$def(S[2`] < S[i]) \wedge def(i \neq 1`) \wedge_L ((S[2`] < S[i] \wedge i \neq 1` \wedge Wp(2` := i, \color{red}{A})) \vee (S[2`] \geq S[i] \vee i = 1` \wedge Wp(Skip, \color{red}{A}))) \equiv$

$0 \leq 2`, i < |S| \wedge_L ((S[2`] < S[i] \wedge i \neq 1` \wedge def(i) \wedge_L \color{red}{A}^{2`}_i) \vee (S[2`] \geq S[i] \vee i = 1` \wedge \color{red}{A})) \equiv$

$0 \leq 2`, i < |S| \wedge_L ((S[2`] < S[i] \wedge i \neq 1` \wedge (-1 \leq i < |S| \wedge_L 0 \leq i, 1` < |S| \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i + 1 \wedge j \neq 1` \wedge j \neq i \longrightarrow_L S[j] < S[i] < S[1`]))) \vee (S[2`] \geq S[i] \vee i = 1` \wedge \color{red}{A})) \equiv$

$0 \leq 2`, i < |S| \wedge_L (((S[2`] < S[i] \wedge i \neq 1`) \wedge (0 \leq i, 1` < |S| \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i + 1 \wedge j \neq 1` \wedge j \neq i \longrightarrow_L S[j] < S[i] < S[1`]))) \vee ((S[2`] \geq S[i] \vee i = 1`) \wedge 0 \leq 2`, 1` < |S| \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i + 1 \wedge j \neq 1` \wedge j \neq 2` \longrightarrow_L S[j] < S[2`] < S[1`])) \equiv$

$0 \leq 2`, i, 1` < |S| \wedge_L (((S[2`] < S[i] \wedge i \neq 1`) \wedge (\forall j : \mathbb{Z})(0 \leq j < i + 1 \wedge j \neq 1` \wedge j \neq i \longrightarrow_L S[j] < S[i] < S[1`]))) \vee ((S[2`] \geq S[i] \vee i = 1`) \wedge (\forall j : \mathbb{Z})(0 \leq j < i + 1 \wedge j \neq 1` \wedge j \neq 2` \longrightarrow_L S[j] < S[2`] < S[1`])) \equiv$

$0 \leq 2`, i, 1` < |S| \wedge_L (((S[2`] < S[i] \wedge i \neq 1`) \wedge (\forall j : \mathbb{Z})(0 \leq j < i + 1 \wedge j \neq 1` \wedge j \neq i \longrightarrow_L S[j] < S[i] < S[1`]))) \vee ((S[2`] \geq S[i] \wedge (\forall j : \mathbb{Z})(0 \leq j < i + 1 \wedge j \neq 1` \wedge j \neq 2` \longrightarrow_L S[j] < S[2`] < S[1`])) \vee (i = 1` \wedge (\forall j : \mathbb{Z})(0 \leq j < i + 1 \wedge j \neq 1` \wedge j \neq 2` \longrightarrow_L S[j] < S[2`] < S[1`])))$

$\color{brown}{indices} \wedge_L (\color{blue}{Caso1} \vee (\color{green}{Caso2{,}1} \vee (i = 1` \wedge (\forall j : \mathbb{Z})(0 \leq j < i + 1 \wedge j \neq 1` \wedge j \neq 2` \longrightarrow_L S[j] < S[2`] < S[1`])))) \equiv$

$\color{brown}{indices} \wedge_L (\color{blue}{Caso1} \vee (\color{green}{Caso2{,}1} \vee \color{red}{D})$

$\color{red}{D} \equiv i = 1` \wedge (\forall j : \mathbb{Z})(0 \leq j < i + 1 \wedge j \neq 1` \wedge j \neq 2` \longrightarrow_L S[j] < S[2`] < S[1`]) \equiv$

$i = 1` \wedge (\forall j : \mathbb{Z})(0 \leq j < i \wedge j \neq 1` \wedge j \neq 2` \longrightarrow_L S[j] < S[2`] < S[1`]) \wedge (i \neq 1` \wedge i \neq 2` \longrightarrow_L S[i] < S[2`] < S[1`]) \equiv$

$(\forall j : \mathbb{Z})(0 \leq j < i \wedge j \neq 1` \wedge j \neq 2` \longrightarrow_L S[j] < S[2`] < S[1`]) \wedge (1` \neq 1` \wedge i \neq 2` \longrightarrow_L S[1`] < S[2`] < S[1`]) \equiv$

$(\forall j : \mathbb{Z})(0 \leq j < i \wedge j \neq 1` \wedge j \neq 2` \longrightarrow_L S[j] < S[2`] < S[1`]) \wedge (False \longrightarrow_L False) \equiv$

$(\forall j:\mathbb{Z})(0\leq j<i\wedge j\neq 1` \wedge j\neq 2` \longrightarrow_L S[j]<S[2`]<S[1`])\wedge True \equiv$

$(\forall j:\mathbb{Z})(0\leq j<i\wedge j\neq 1` \wedge j\neq 2` \longrightarrow_L S[j]<S[2`]<S[1`])$

Luego juntado todo quedaria:

$indices \wedge_L (Caso1 \vee (Caso2{,}1 \vee (\forall j:\mathbb{Z})(0\leq j<i\wedge j\neq 1` \wedge j\neq 2` \longrightarrow_L S[j]<S[2`]<S[1`]))$

Calculamos $I\wedge B$

$I\wedge B\equiv 0\leq i<|S|\wedge_L 0\leq 2`,1`<|S|\wedge_L (\forall j:\mathbb{Z})(0\leq j<i\wedge j\neq 1` \wedge j\neq 2` \longrightarrow_L S[j]<S[2`]<S[1`]\equiv$

$0\leq 2`,1`,i<|S|\wedge_L (\forall j:\mathbb{Z})(0\leq j<i\wedge j\neq 1` \wedge j\neq 2` \longrightarrow_L S[j]<S[2`]<S[1`]\equiv C$

Para verificar la implicación la separamos por partes:

- $C\longrightarrow Indices$
  $0\leq 2`,1`,i<|S|\wedge_L (\forall j:\mathbb{Z})(0\leq j<i\wedge j\neq 1` \wedge j\neq 2` \longrightarrow_L S[j]<S[2`]<S[1`]\longrightarrow 0\leq 2`,i,1`<|S|\wedge_L (Caso1\vee(Caso2{,}1\vee D)\equiv$

  $0\leq 2`,1`,i<|S|\longrightarrow 0\leq 2`,i,1`<|S|\equiv True$
- $C\longrightarrow (Caso1\vee(Caso2{,}1\vee D)$
  Para realizar esta implicación basta con ver que o vale $C\longrightarrow Caso1$ o $C\longrightarrow Caso2,1$ o $C\longrightarrow D$

  Podemos ver a simple vista que las mas fácil de probar es $C\longrightarrow D$

  $0\leq 2`,1`,i<|S|\wedge_L (\forall j:\mathbb{Z})(0\leq j<i\wedge j\neq 1` \wedge j\neq 2` \longrightarrow_L S[j]<S[2`]<S[1`])\longrightarrow (\forall j:\mathbb{Z})(0\leq j<i\wedge j\neq 1` \wedge j\neq 2` \longrightarrow_L S[j]<S[2`]<S[1`])\equiv$

  $(\forall j:\mathbb{Z})(0\leq j<i\wedge j\neq 1` \wedge j\neq 2` \longrightarrow_L S[j]<S[2`]<S[1`])\longrightarrow (\forall j:\mathbb{Z})(0\leq j<i\wedge j\neq 1` \wedge j\neq 2` \longrightarrow_L S[j]<S[2`]<S[1`])\equiv True$
  Como vale este caso, luego vale que $C\longrightarrow Caso1\vee(Caso2{,}1\vee D)$

Concluyendo como vale que $C\longrightarrow Caso1\vee(Caso2{,}1\vee D)$ y $C\longrightarrow indices$ entonces vale:
$C\longrightarrow indices\wedge_L Caso1\vee(Caso2{,}1\vee D)$

Por lo tanto vale la prueba $\{I\wedge B\}P\{I\}$

4. $I\wedge f_v\leq 0\longrightarrow\neg B$
   Del invariante sabemos que $2\leq i\leq |S|$ y de F_v que:

   $$f_v\leq 0\longleftrightarrow |S|-i\leq 0\longleftrightarrow |S|\leq i$$

   Entonces, tenemos $2\leq i\leq |S|\wedge |S|\leq i$ que implica que $|S|=i$ que a su vez implica a $i\geq |S|$, que es $\neg B$.

5. $\{I\wedge B\wedge v_0=|S|-i\}P\{|S|-i<v_0\}$

   Veamos si $I\wedge B\wedge v_0=|S|-i\longrightarrow wp(if()...;i:=i+1,|S|-i<V_0)$
   $wp(i:=i+1,|S|-i<V_0)\equiv$
   $\equiv |S|-(i+1)<V_0$
   $\equiv |S|-i-1<V_0$

   $wp(if()...,|S|-i-1<V_0)\equiv$

   $\equiv (B\wedge wp(2':=i,|S|-i-1<V_0)\vee(\neg B\wedge wp(Skip,|S|-i-1<V_0)$

   $wp(2':=i,|S|-i-1<V_0)\equiv$

$\equiv |S| - i - 1 < V_0$

Nos basta con probar:
$B \wedge wp(2' := i, |S| - i - 1 < V_0) \equiv i < |S| \wedge |S| - i - 1 < V_0$ (La rama positiva del If)

Como asumimos B verdadero, la primer mitad de la expresion ya esta probada, nos falta la segunda mitad:
$|S| - i - 1 < V_0$

Como asumimos $v_0 = |S| - i$ verdadero lo vamos a utilizar para probar la segunda expresión. Como v_0 es igual a $|S| - i$, sabemos que es mayor a cualquier cosa menor que $|S| - i$, como mínimo, mayor $|S| - i - 1$, entonces:

$$v_0 = |S| - i \equiv v_0 > |S| - i - 1$$

Esto es exactamente lo que queriamos probar.

Una vez terminado este Wp del ciclo, continuamos con Wp del codigo anterior al segundo ciclo con respecto al P_c del segundo ciclo.

$Wp(2 := 0, Wp(if(1` = 0)then(2` := 1)else(skip), Wp(i := 0, P_{c_2})))$

Vamos calculándolo por partes:

$Wp(i := 0, P_c) \equiv def(0) \wedge P^i_{c_0} \equiv$

$(2` = 0 \vee 2` = 1) \wedge 0 \leq 1` < |S| \wedge 1` \neq 2` \wedge 0 = 0 \wedge |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1`]) \equiv$

$(2` = 0 \vee 2` = 1) \wedge 0 \leq 1` < |S| \wedge 1` \neq 2` \wedge |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1`]) \equiv E$
$Wp(if\ (1` = 0)\ then\ (2` := 1)\ else\ (Skip), E) \equiv def(1` = 0) \wedge_L (1` = 0 \wedge (Wp(2 := 1, E)) \vee (1` \neq 0 \wedge Wp(Skip, E)) \equiv$

$(1` = 0 \wedge def(1) \wedge_L E^{2`}_1) \vee (1` \neq 0 \wedge E) \equiv$

$(1` = 0 \wedge (1 = 0 \vee 1 = 1) \wedge 0 \leq 1` < |S| \wedge 1` \neq 1 \wedge |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1`])) \vee (1` \neq 0 \wedge (2` = 0 \vee 2` = 1) \wedge 0 \leq 1` < |S| \wedge 1` \neq 2` \wedge |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1`])) \equiv$

$(1` = 0 \wedge True \wedge 0 \leq 1` < |S| \wedge 0 \neq 1 \wedge |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1`])) \vee (1` \neq 0 \wedge (2` = 0 \vee 2` = 1) \wedge 0 \leq 1` < |S| \wedge 1` \neq 2` \wedge |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1`])) \equiv$

$(1` = 0 \wedge 0 \leq 1` < |S| \wedge |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1`])) \vee (1` \neq 0 \wedge (2` = 0 \vee 2` = 1) \wedge 0 \leq 1` < |S| \wedge 1` \neq 2` \wedge |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1`])) \equiv F$

$Wp(2 := 0, F) \equiv def(0) \wedge_L F^{2`}_0 \equiv$

$((1` = 0 \wedge 0 \leq 1` < |S| \wedge |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1`])) \vee (1` \neq 0 \wedge (0 = 0 \vee 0 = 1) \wedge 0 \leq 1` < |S| \wedge 1` \neq 0 \wedge |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1`]))) \equiv$

$(1` = 0 \wedge \leq 1` < |S| \wedge |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1`])) \vee (1` \neq 0 \wedge True \wedge 0 \leq 1` < |S| \wedge |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1`])) \equiv$

$(1` = 0 \wedge \leq 1` < |S| \wedge |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1`])) \vee (1` \neq 0 \wedge 0 \leq 1` < |S| \wedge |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1`])) \equiv$

$0 \leq 1` < |S| \wedge |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1`])$
Una vez terminado el código anterior al segundo ciclo, calculamos el wp respecto al primer ciclo

- $I \equiv 0 \leq i \leq |S| \wedge 0 \leq 1` < |S| \wedge_L |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < i \longrightarrow S[j] \leq S[1`])$
- $Q_c \equiv 0 \leq 1` < |S| \wedge |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1`])$
- $P_c \equiv i = 0 \wedge 1` = 0 \wedge |S| \geq 3$
- $B \equiv i < |S|$
- $f_v = |S| - i$

1. $\underline{P_c \longrightarrow I}$

   $i = 0 \wedge 1` = 0 \wedge |S| \geq 3 \longrightarrow 0 \leq i \leq |S| \wedge 0 \leq 1` < |S| \wedge_L |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < i \longrightarrow S[j] \leq S[1`])$

   Asumimos como verdadero el P_c luego, reemplazamos en el invariante:

   $0 \leq 0 < |S| \wedge 0 \leq 0 \leq |S| \wedge_L |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < 0 \longrightarrow S[j] \leq S[1`]) \equiv$

   $True \wedge True \wedge True \wedge (\forall j : \mathbb{Z})(False \longrightarrow S[j] \leq S[1`]) \equiv True$

   Luego vale que $P_c \longrightarrow I$

2. $\underline{I \wedge \neg B \longrightarrow Q_c}$

   $I \wedge \neg B \equiv 0 \leq i \leq |S| \wedge 0 \leq 1` < |S| \wedge_L |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < i \longrightarrow S[j] \leq S[1`]) \wedge i \geq |S| \equiv$

   $i = |S| \wedge 0 \leq 1` < |S| \wedge_L |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < |S| \longrightarrow S[j] \leq S[1`]) \equiv Q_c$

   Luego $Q_c \longrightarrow Q_c \equiv True$

   Luego vale que $I \wedge B \longrightarrow Q_c$

3. $\underline{\{I \wedge B\}P\{I\}}$
   Para esta prueba queremos ver que $I \wedge B \longrightarrow Wp(P, I)$ siendo P el código dentro del primer ciclo. Primero calculamos el wp:

   $Wp(P, I) \equiv Wp(if(S[1`] < S[i])then(1` := i)else(Skip), Wp(i := i + 1, I))$

   $Wp(i := i + 1, I) \equiv def(i + 1) \wedge_L I^i_{i+1} \equiv$

   $0 \leq i + 1 \leq |S| \wedge 0 \leq 1` < |S| \wedge_L |S| \geq 3 \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i + 1 \longrightarrow S[j] \leq S[1`]) \equiv$

   $-1 \leq i < |S| \wedge 0 \leq 1` < |S| \wedge_L |S| \geq 3 \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i + 1 \longrightarrow S[j] \leq S[1`]) \equiv G$

   $Wp(if\ (S[1`] < S[i])\ then\ (1` := i)\ else\ skip, G) \equiv$

   $def(S[1`] < S[i]) \wedge ((S[1`] < S[i] \wedge Wp(1` := i, G)) \vee (S[1`] \geq S[i] \wedge Wp(Skip, G)) \equiv$

   $0 \leq i, 1` < |S| \wedge ((S[1`] < S[i] \wedge def(i) \wedge_L G^{1`}_i) \vee (S[1`] \geq S[i] \wedge G)) \equiv$

   $0 \leq i, 1` < |S| \wedge ((S[1`] < S[i] \wedge -1 \leq i < |S| \wedge 0 \leq i < |S| \wedge_L |S| \geq 3 \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i + 1 \longrightarrow S[j] \leq S[1`])) \vee (S[1`] \geq S[i]) \wedge -1 \leq i < |S| \wedge 0 \leq 1` < |S| \wedge_L |S| \geq 3 \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i + 1 \longrightarrow S[j] \leq S[1`])) \equiv$

   $0 \leq i, 1` < |S| \wedge ((S[1`] < S[i] \wedge 0 \leq i < |S| \wedge |S| \geq 3 \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i + 1 \longrightarrow S[j] \leq S[1`])) \vee (S[1`] \geq S[i] \wedge -1 \leq i < |S| \wedge 0 \leq 1` < |S| \wedge_L |S| \geq 3 \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i + 1 \longrightarrow S[j] \leq S[1`]))) \equiv$

   $(S[1`] < S[i] \wedge 0 \leq i, 1` < |S| \wedge |S| \geq 3 \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i + 1 \longrightarrow S[j] \leq S[1`])) \vee (S[1`] \geq S[i] \wedge 0 \leq i, 1` < |S| \wedge_L |S| \geq 3 \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i + 1 \longrightarrow S[j] \leq S[1`])) \equiv$

   $0 \leq i, 1` < |S| \wedge |S| \geq 3 \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i + 1 \longrightarrow S[j] \leq S[1`]) \equiv H$

   Calculemos que es $I \wedge B$

   $I \wedge B \equiv 0 \leq i \leq |S| \wedge 0 \leq 1` < |S| \wedge_L |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < i \longrightarrow S[j] \leq S[1`]) \wedge i < |S| \equiv$

$0 \leq 1`, i < |S| \wedge_L |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < i \longrightarrow S[j] \leq S[1`]) \equiv H$
Luego vale que $H \longrightarrow H \equiv True$

Luego vale la prueba de $\{I \wedge B\}P\{I\}$

4. $\underline{I \wedge f_v \leq 0 \longrightarrow \neg B}$
$I \wedge f_v \leq 0 \equiv 0 \leq i \leq |S| \wedge 0 \leq 1` < |S| \wedge_L |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < i \longrightarrow S[j] \leq S[1`]) \wedge |S| - i \leq 0 \equiv$

$0 \leq i \leq |S| \wedge 0 \leq 1` < |S| \wedge_L |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < i \longrightarrow S[j] \leq S[1`]) \wedge |S| \leq i \equiv$

$i = |S| \wedge 0 \leq 1` < |S| \wedge_L |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < i \longrightarrow S[j] \leq S[1`]) \equiv J$

$J \longrightarrow i \geq |S| \equiv i = |S| \longrightarrow i \geq |S| \equiv True$

Luego la prueba $I \wedge f_v \leq 0 \longrightarrow \neg B$ vale

5. $\underline{\{I \wedge B \wedge v_0 = |S| - i\}P\{|S| - i < v_0\}}$
Veamos si $I \wedge B \wedge v_0 = |S| - i \longrightarrow Wp(if()...; i := i + 1, |S| - i < V_0)$

Calculamos los Wp

$Wp(i := i + 1, |S| - i < v_0) \equiv def(i + 1) \wedge (|S| - i < v_0)^i_{i+1} \equiv |S| - i - 1 < v_0 \equiv |S| - i < v_0 + 1 \equiv K$

$Wp(if\ (S[1`] < S[i]\ then\ (1` := i)\ else\ (Skip), K) \equiv$

$def(S[1`] < S[i]) \wedge ((S[1`] < S[i] \wedge Wp(1` := i, K)) \vee (S[1`] \geq S[i] \wedge Wp(Skip, K))) \equiv$

$0 \leq 1`, i < |S| \wedge ((S[1`] < S[i] \wedge K^{1`}_i) \vee (S[1`] \geq S[i] \wedge K)) \equiv$

$0 \leq 1`, i < |S| \wedge ((S[1`] < S[i] \wedge |S| - i < v_0 + 1) \vee (S[1`] \geq S[i] \wedge |S| - i < v_0 + 1) \equiv$

$0 \leq 1`, i < |S| \wedge |S| - i < v_0 + 1 \equiv L$
Luego queremos ver que $I \wedge B \wedge v_0 = |S| - i \longrightarrow L$

Calculemos que es $I \wedge B \wedge v_0 = |S| - i$
$I \wedge B \wedge v_0 = |S| - i \equiv$

$0 \leq 1`, i < |S| \wedge_L |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < i \longrightarrow S[j] \leq S[1`]) \wedge v_0 = |S| - i$

Luego:

$0 \leq 1`, i < |S| \wedge_L |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < i \longrightarrow S[j] \leq S[1`]) \wedge v_0 = |S| - i \longrightarrow 0 \leq 1`, i < |S| \wedge |S| - i < v_0 + 1$

- $0 \leq 1`, i < |S| \wedge_L |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < i \longrightarrow S[j] \leq S[1`]) \wedge v_0 = |S| - i \longrightarrow 0 \leq 1`, i < |S| \equiv$

  $0 \leq 1`, i < |S \longrightarrow 0 \leq 1`, i < |S| \equiv True$

- $0 \leq 1`, i < |S| \wedge_L |S| \geq 3 \wedge (\forall j : \mathbb{Z})(0 \leq j < i \longrightarrow S[j] \leq S[1`]) \wedge v_0 = |S| - i \longrightarrow |S| - i < v_0 + 1 \equiv$

  $v_0 = |S| - i \longrightarrow |S| - i < v_0 + 1 \equiv v_0 < v_0 + 1 \equiv True$

Como valen ambas implicaciones por separado, vale la implicacion conjunta

Luego vale la prueba $\{I \wedge B \wedge v_0 = |S| - i\}P\{|S| - i < v_0\}$

Una vez terminado de probar el teorema del invariante, seguimos calculando el Wp del código anterior al primer ciclo respecto a la precondición del primer ciclo

$Wp(i := 0, i = 0 \wedge 1` = 0 \wedge |S| \geq 3) \equiv def(0) \wedge 0 = 0 \wedge 1` = 0 \wedge |S| \geq 3 \equiv 1` = 0 \wedge |S| \geq 3$

$Wp(1` := 0, 1` = 0 \wedge |S| \geq 3) \equiv def(0) \wedge_L 0 = 0 \wedge |S| \geq 3 \equiv |S| \geq 3$

Luego el Wp(Programa,Post) $\equiv |S| \geq 3$

Para comprobar que el algoritmo es correcto respecto a su especificacion debemos validar la siguiente implicación:

$Pre \Longrightarrow |S| \geq 3 \equiv$

$nohayRepetidos(S) \wedge cantVotosValidos(S) \wedge minimoDePartidos(S) \longrightarrow |S| \geq 3 \equiv$

$nohayRepetidos(S) \wedge cantVotosValidos(S) \wedge |S| \geq 3 \longrightarrow |S| \geq 3 \equiv$

$|S| \geq 3 \longrightarrow |S| \geq 3 \equiv True$

Luego el algoritmo es correcto respecto a su especificación