DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Trabajo Practico N°1

Especificacion y WP

---

17 de septiembre de 2023 | Algoritmos y Estructura de Datos

TeamJere

Integrante	LU	Correo electrónico
Cocú, Dante	1119/22	`dcocu19@gmail.com`
Novoa Shule, Joaquin	1043/22	`nojoaco2003@gmail.com`
Said, Tomas	170/23	`saidtomasur@gmail.com`
Wittmund Montero, Lourdes	1103/22	`lourdesmonterochiara@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

Anexo - Especificación

1. Especificación

1.1. *hayBallotage*: verifica si hay ballotage en la eleccion presidencial.

`proc` `hayBallotage` (`in` escrutinio : $seq\langle\mathbb{Z}\rangle$) : Bool
`requiere` $\{escrutinioValido(escrutinio)\}$
`asegura` $\{(|escrutinio| = 2 \longrightarrow res = False) \wedge (res = False \leftrightarrow (\exists l : seq\langle\mathbb{Z}\rangle)\ (mismosEltos(l, escrutinio) \wedge estaOrdenadoDecr(subseq(l, 0, |l| - 2)) \wedge l[|l| - 1] = escrutinio[|escrutinio| - 1] \wedge (porcentajeVotos(l[0], l) > 0{,}45 \vee porcentajeVotos(l[0], l) > 0{,}40 \wedge porcentajeVotos(l[0], l) - porcentajeVotos(l[1], l) < 0{,}10)))\}$

`pred` `escrutinioValido` (`in` lista : $seq\langle\mathbb{Z}\rangle$) {
$(\forall i : \mathbb{Z})(0 \leq i < |lista| \longrightarrow lista[i] \geq 0) \wedge (|lista| \geq 3) \wedge (sonTodosDistintos(lista))$
}
/* el pedir la longitud mayor o igual a 3 se debe a que consideramos que para que sea una eleccion se debe de poder elegir, y si solo hay un partido no se puede elegir. La lista mas chica posible de 2 partidos tendria los votos de cada partido y como ultimo elemento los voto en blanco *

`pred` `sonTodosDistintos` (`in` lista : $seq\langle\mathbb{Z}\rangle$) {
$(\forall i, j : \mathbb{Z})(0 \leq i, j < |lista| \wedge i \neq j \longrightarrow lista[i] \neq lista[j])$
}

`pred` `mismosEltos` (`in` l1 : $seq\langle\mathbb{Z}\rangle$, `in` l2 :$seq\langle\mathbb{Z}\rangle$) {
$|l1| = |l2| \wedge (\forall i : \mathbb{Z})\ (0 \leq i < |l2| \longrightarrow pertenece(l2[i], l1)) \wedge (cantapariciones(l2[i], l2) = cantidadapariciones(l2[i], l1))$
$\wedge\ (cantapariciones(l1[i], l2) = cantidadapariciones(l1[i], l1))$
}

`pred` `pertenece` (n: $\mathbb{Z}$, l: $seq\langle\mathbb{Z}\rangle$) {
$(\exists i : \mathbb{Z})(0 \leq i \leq |l| \longrightarrow l[i] = n)$
}

`pred` `estaOrdenadoDecr` (`in` lista: $seq\langle\mathbb{Z}\rangle$) {
$(\forall i : \mathbb{Z})(0 \leq i < |lista| - 1 \longrightarrow lista[i] \geq lista[i + 1])$
}

`aux` `sumarEltos` (lista : $seq\langle\mathbb{Z}\rangle$) : $\mathbb{Z}\ = \sum_{i=0}^{|lista|-1} lista[i]$;

`aux` `porcentajeVotos` (votosCandidato: $\mathbb{Z}$, escrutinio : $seq\langle\mathbb{Z}\rangle$) : $\mathbb{Z}\ = \frac{votosCandidato}{sumarEltos(escrutinio)}$;

1.2. *hayFraude:* verifica que los votos validos de los tres tipos de cargos electivos sumen lo mismo.

`proc` `hayFraude` (`in` escrutinio_presidencial :$seq\langle\mathbb{Z}\rangle$, `in` escrutinio_senadores:$seq\langle\mathbb{Z}\rangle$, `in` escrutinio_diputados: $seq\langle\mathbb{Z}\rangle$) : Bool
`requiere` $\{(escrutinioValido(escrutinio\_presidencial)) \wedge escrutinioValido(escrutinio\_senadores) \wedge escrutinioValido(escrutinio\_diputados)\}$
`asegura` $\{res = True \leftrightarrow sumarEltos(escrutinio\_senadores) \neq sumarEltos(escrutinio\_presidencial) \wedge sumarEltos(escrutinio\_senadores) \neq sumarEltos(escrutinio\_diputados) \wedge sumarEltos(escrutinio\_presidencial) \neq sumarEltos(escrutinio\_diputados)\}$

1.3. *obtenerSenadoresEnProvincia*: obtiene los id de los partidos (primero y segundo) para la elección de senadores en una provincia. El id es el índice de las listas escrutinios.

`proc` `obtenerSenadoresEnProvincia` (`in` escrutinio: $seq\langle\mathbb{Z}\rangle$) : $\mathbb{Z}\times\ \mathbb{Z}$
`requiere` $\{escrutinioValido(escrutinio)\}$
`asegura` $\{esMax(res_0, escrutinio, |escrutinio| - 1 \wedge esMax2(res_1, escrutinio, |escrutinio| - 1)\}$

`pred` `esMax` (`in` ind: $\mathbb{Z}$, `in` l: $seq\langle\mathbb{Z}\rangle$, `in` end: $\mathbb{Z}$) {
$0 \leq ind < end \wedge_L (\forall k : \mathbb{Z})\ (0 \leq k < end \longrightarrow_L l[ind] \geq l[k])$
}
`pred` `esMax2` (`in` ind: $\mathbb{Z}$, `in` l: $seq\langle\mathbb{Z}\rangle$, `in` end: $\mathbb{Z}$) {
$0 \leq ind < end \wedge_L (\forall k : \mathbb{Z})\ (0 \leq k < end \wedge \neg esMax(k, l, end) \longrightarrow_L l[ind] \geq l[k])$

}

1.4. *calcularDHondtEnProvincia*: calcula los cocientes según el método d'Hondt para diputados en una provincia (importante: no es necesario ordenar los partidos por cantidad de votos).

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

`requiere` $\{(escrutinioValido(escrutinio)) \wedge ((\exists i : \mathbb{Z})(0 \leq i < |escrutinio| - 1 \longrightarrow arribaUmbral(escrutinio[i], escrutinio))\}$

`asegura` $\{((((\forall i : \mathbb{Z})(0 \leq i \leq |res| \longrightarrow |res[0]| = |res[i]|)) \wedge ((\forall i : \mathbb{Z})(0 \leq i < |res| \longrightarrow |res[0]| = |res[i]|))$

$(|res| = cantarribaUmbral(escrutinio) \wedge |res[0]| = cantBancas) \wedge$

$((\forall i : \mathbb{Z})(0 \leq i < |res| \longrightarrow columnasDivididasPorCocientes(res[i]))) \wedge (matrizNoPermutada(res)) \wedge$

$((\forall i : \mathbb{Z})(0 \leq i < |res| \longrightarrow pertenece(res[i][0], escrutinio))) \wedge$

$((\forall i : \mathbb{Z})(0 \leq i < |\mathbb{Z}| \longrightarrow arribaUmbral(res[i][0], escrutinio))) \wedge (sinVotoBlanco(res, escrutino))\}$

`pred arribaUmbral` (in l: $seq\langle\mathbb{Z}\rangle$, in elto : $\mathbb{Z}$) {
$porcentajeVotos(elto, l) > 0{,}03$
}

`aux cantarribaUmbral` (in lista : $seq\langle\mathbb{Z}\rangle$) : $\mathbb{Z} = \sum_{i=0}^{|lista|-1}(\textbf{if } arribaUmbral(l, i) \textbf{ then } 1 \textbf{ else } 0 \textbf{ fi })$;

`pred columnasDivididasPorCocientes` (in l : $seq\langle\mathbb{Z}\rangle$) {
$(\forall I : \mathbb{Z})(0 \leq i < |l| \longrightarrow l[i] = l[0]/i + 1)$
}

`pred sinVotoBlanco` (in matriz : $seq\langle seq\langle\mathbb{Z}\rangle$, in l : $seq\langle\mathbb{Z}\rangle\rangle$) {
$(\forall i : \mathbb{Z})(0 \leq i < |matriz| \longrightarrow matriz[i][0] \neq l[|l| - 1])$
}

`pred matrizNoPermutada` (in matriz : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
$(\forall i : \mathbb{Z})(0 \leq i < |matriz| \longrightarrow sinPermutacion(matriz[0], matriz[i], i))$
}
`pred sinPermutacion` (in l1 :$seq\langle\mathbb{Z}\rangle$,in l2 :$seq\langle\mathbb{Z}\rangle$,j : $\mathbb{Z}$) {
$(\forall i : \mathbb{Z})(0 \leq< |l1| \longrightarrow l1[i] = \frac{l2[i]}{j+1})$
}

1.5. *obtenerDiputadosEnProvincia*: obtenerDiputadosEnProvincia: calcula la cantidad de bancas de diputados obtenidas por cada partido en una provincia.

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

`requiere` $\{(|dHondt| > 1) \wedge (escrutiniValido(escrutinio)) \wedge (esdHondtAsociada(dHondt, escrutinio))\}$

/* se respeta las condiciones anterirores donde la Dhondt No incluia los partidos que no pasaban el umbral y los votos en blanco*

`asegura` $\{(esDiputadosEnProvincia(cantBancas, Dhondt, res)\}$

`aux cantapariciones` (in elto : $\mathbb{Z}$, in lista : $seq\langle\mathbb{Z}\rangle$) : $\mathbb{Z} = \sum_{i=0}^{|lista|-1}(\textbf{if } l[i] = elto \textbf{ then } 1 \textbf{ else } 0 \textbf{ fi })$;

`pred pertenecenMatriz` (in matriz: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, in elto : $\mathbb{Z}$) {
$(\exists i : \mathbb{Z})(0 \leq i < |matriz| \longrightarrow pertenece(matriz[i], elto))$
}

`pred esDiputadosEnProvincia` (in n:$\mathbb{Z}$, in dHondt :$seq\langle seq\langle\mathbb{Z}\rangle\rangle$, in lista : $seq\langle\mathbb{Z}\rangle$) {
$(|lista| = |dHondt|) \wedge (sumarEltos(lista) = n) \wedge ((\exists l1, l2 : seq\langle\mathbb{Z}\rangle)(esListaDeMaximos(dHondt, l1) \wedge |l2| = cantBancas \wedge$
$estaTraducido(dHondt, l1, l2)) \wedge (\forall i : \mathbb{Z})(0 \leq i < |lista|)(lista[i] = cantapariciones(i, l2))$
}
`pred todosPertenecenMatriz` (in matriz: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, in lista: $seq\langle\mathbb{Z}\rangle$) {
$(\forall i : \mathbb{Z})(0 \leq i < |lista| \longrightarrow pertenecenMatriz(matriz, lista[i]))$
}

`pred esListaDeMaximosHasta` (in matriz : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, in lista : $seq\langle\mathbb{Z}\rangle$, in i : $\mathbb{Z}$) {
$todosPertenecenMatriz(matriz, lista) \wedge estaOrdenadoDecr(lista) \wedge (|lista| = i) \wedge$
$(\forall xinmatriz)(/mayorAtodos(lista, x) \wedge \neg(pertenece(x, lista))$

}

pred `esTraduccion` (in matriz : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$,in elto : $\mathbb{Z}$, in fila : $\mathbb{Z}$) {
$pertenece(elto, matriz[fila])$
}

/* al hablar de traducido hacemos referencia a la ubicacion(numero de fila) de un numero en la matriz, por eejemplo, la lista [A11,A22,A33,A23] su traduccion sera [1,2,3,2] *

pred `estaTraducido` (in matriz : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$,in lista :$seq\langle\mathbb{Z}\rangle$, in traducida : $seq\langle\mathbb{Z}\rangle$) {
$(\forall i : \mathbb{Z})(0 \leq i < |traducida| \longrightarrow esTraduccion(matriz, lista[i], traducida[i])$
}

pred `esdHondtAsociada` (in matriz : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, in l : $seq\langle\mathbb{Z}\rangle$) {
$(sinVotoBlanco(matriz, l)) \wedge (\forall i : \mathbb{Z})(0 \leq i < |matriz| \longrightarrow columnasDivididasPorCocientes(matriz[i])) \wedge$
$matrizNoPermutada(matriz) \wedge (|matriz| = cantArribaUmbral(l)) \wedge (matriz[0] = l)$
}

1.6. *validarListasDiputadosEnProvincia*:verifica que la listas de diputados de cada partido en una provincia contenga exactamente la misma cantidad de candidatos que bancas en disputa en esa provincia, y que además se cumpla la alternancia de géneros.

proc `validarListasDiputadosEnProvincia` (in cantBancas :$\mathbb{Z}$, in listas :$seq\langle seq\langle dni : \mathbb{Z} \times genero : \mathbb{Z}\rangle\rangle$) : Bool
`requiere` $\{DniyGeneroValido(listas) \wedge cantBancasValio(cantBancas)\}$
`asegura` $\{res = True \longleftrightarrow (\forall i, j : \mathbb{Z})(0 \leq i < |listas| \longrightarrow cantBancas = |listas[i]|) \wedge$
$res = True \longleftrightarrow (\forall i, j : \mathbb{Z})(0 \leq i < |listas| \wedge 0 \leq j < cantBancas - 1 \longrightarrow lista[i][j][1] \neq lista[i][j + 1][1])\}$

pred `DniyGeneroValido` (in listas :$seq\langle seq\langle dni : \mathbb{Z} \times genero : \mathbb{Z}\rangle\rangle$) {
$(\forall i : \mathbb{Z})(0 \leq i < |listas| \longrightarrow (listas[i][0] > 0) \wedge (listas[i][1] = 1 \vee listas[i][1] = 2))$
}

2. Implementaciones y demostraciones de correctitud

2.1. Proponer algoritmos para todos los problemas, excepto para calcularDHondtEnProvincia y obtenerDiputado- sEnProvincia.

```
res := True;
i := 0;
sumador := 0;
max := −1;
max2 := −2;
j:=0;
while (i < escrutinio.size()) do
    sumador := sumador + escrutinio[i];
    if(max<= escrutinio[i] && i != escrutinio.size()−1) then
        max := s[i]
    else
        skip
    endif
    i := i + 1
endwhile
if ( (max∗100)/sumador > 45) then
    res:= False
else
    skip
endif
while (j<escrutinio.size()−1) do
    if (escrutinio[j]>=max2 and escrutinio[j]<max) then
        max2 := escrutinio[j]
    else
```

```
        skip
    endif;
endwhile
if ( (max*100)/sumador − (max2*100)/sumador > 10 && (max*100)/sumador > 40 ) then
    res:= False
else
    skip
endif
return res
```

Código 1: Algoritmo hayBallotage

```
sumadorp := 0;
i := 0;
while (i < escrutinio_presidencial.size()) do
    sumadorp := sumadorp + escrutinio_presidencial[i];
    i := i + 1;
endwhile
sumadors := 0;
j := 0;
while (j < escrutinio_senadores.size()) do
    sumadors := sumadors + escrutinio_senadores[j];
    j := j + 1;
endwhile
sumadord := 0;
k := 0;
while (k < escrutinio_diputados.size()) do
    sumadord := sumadord + escrutinio_diputados[k];
    k := k + 1;
endwhile
if (sumadors != sumadorp || sumadors != sumadord || sumadorp != sumadord) then
   res := True;
else
    res := False;
endif
```

Código 2: Algoritmo hayFraude

```
i:=0
indiceMax1:=escrutinio[0]
while (i < escrutinio.size()−1) do
    if (escrutinio[i] > escrutinio[indiceMax1]) then
        indiceMax1:=i
    else
        skip
    endif
    i := i+1
endwhile
j:=1
indiceMax2:=0
while (j < escrutinio.size()−1) do
    if (escrutinio[j] > escrutinio[indiceMax1] & escrutinio[j] < escrutinio[indiceMax2]) then
        max2:=escrutinio[j]
        indiceMax2:=j
    else
        skip
    endif
    j := j+1
endwhile
res:=(indiceMax1,indiceMax2)
```

Código 3: Algoritmo obtenerSenadoresEnProvincia

```
res := True;
i := 0;

while (i < listas.size()) do
    if(cantBancas!= listas[i].size() then
        res = False
    i = i+1
    endif
endwhile
k := 0;
while (k < listas.size()) do
    j := 0;
    while (j < cantBancas - 1) do
        if (listas[k][j][1] == listas[k][j+1][1]) then
            res = False
        else
            skip
        endif
        j := j + 1
    endwhile
    k :=  k + 1
endwhile
```

Código 4: Algoritmo validarListasDiputadosEnProvincia

2.2. Weakest Precondition de hayFraude

(definimos e_p como escrutinio_presidencial, e_s como escrutinio_senadores y e_d como escrutinio_diputados por comodidad)

En primer lugar, definimos P como el requiere del programa y tenemos estas dos lineas y el primer ciclo:
$P \equiv escrutinioValido(e\_p) \wedge escrutinioValido(e\_s) \wedge escrutinioValido(e\_d)$

```
sumadorp := 0;
i := 0;
while (i < escrutinio_presidencial.size()) do
    sumadorp := sumadorp + escrutinio_presidencial[i];
    i := i + 1
endwhile
```

Definimos su precondición y poscondición:
$P_{c_1} \equiv P \wedge sumadorp = 0 \wedge i = 0$
$Q_{c_1} \equiv P \wedge i = |e\_p| \wedge sumadorp = \sum_{n=0}^{|e\_p|-1} e\_p[n]$

Planteamos ¿$P \longrightarrow wp(S1; S2, P_{c_1})$?
$wp(S1; S2, P_{c_1}) \equiv wp(S1, wp(S2, P_{c_1}))$

$wp(S2, P_{c_1}) \equiv$
$wp(i := 0, P \wedge sumadorp = 0 \wedge i = 0) \equiv$
$def(0) \wedge_L P \wedge sumadorp = 0 \wedge 0 = 0 \equiv$
$P \wedge sumadorp = 0 \equiv E_2$

$wp(S1, E_2) \equiv$
$wp(sumadorp := 0, P \wedge sumadorp = 0) \equiv$
$def(0) \wedge_L P \wedge 0 = 0) \equiv$
P

Entonces, me queda que $P \longrightarrow P$ que es verdadero

Paso a probar $P_{c_1} \longrightarrow Q_{c_1}$ definiendo un invariante y una función variante. Además, defino B_1 como la guarda del while:
$I \equiv P \wedge 0 \leq i \leq |e\_p| \wedge_L sumadorp = \sum_{n=0}^{i-1} e\_p[n]$
$fv = |e\_p| - i$
$B_{c_1} \equiv i < |e\_p|$

1. $P_{c_1} \longrightarrow I$**:**
$P_{c_1} \equiv P \wedge sumadorp = 0 \wedge i = 0$
$I \equiv P \wedge 0 \leq i \leq |e\_p| \wedge_L sumadorp = \sum_{n=0}^{i-1} e\_p[n]$
Si asumimos P_{c_1} verdadero, I me queda:
$True \wedge 0 \leq 0 \leq |e\_p| \wedge_L 0 = \sum_{n=0}^{0-1} e\_p[n] \equiv$
$True \wedge True \wedge_L True \equiv$
$True$

2. $\{I \wedge B_{c_1}\}S\{I\} \equiv I \wedge B_{c_1} \longrightarrow wp(S1; S2, I)$**:**
$I \equiv P \wedge 0 \leq i \leq |e\_p| \wedge_L sumadorp = \sum_{n=0}^{i-1} e\_p[n]$
$B_{c_1} \equiv i < |e\_p|$

$wp(S1; S2, I) \equiv wp(S1, wp(S2, I))$

$wp(S2, I) \equiv$
$wp(i := i + 1, P \wedge 0 \leq i \leq |e\_p| \wedge_L sumadorp = \sum_{n=0}^{i-1} e\_p[n]) \equiv$
$def(i + 1) \wedge_L P \wedge 0 \leq i + 1 \leq |e\_p| \wedge_L sumadorp = \sum_{n=0}^{i} e\_p[n] \equiv$
$P \wedge 0 \leq i + 1 \leq |e\_p| \wedge_L sumadorp = \sum_{n=0}^{i} e\_p[n] \equiv E_2$
$wp(S1, E_2) \equiv$
$wp(sumadorp := sumadorp + e\_p[i], P \wedge 0 \leq i + 1 \leq |e\_p| \wedge_L sumadorp = \sum_{n=0}^{i} e\_p[n]) \equiv$
$def(sumadorp) \wedge_L def(e\_p[i]) \wedge_L P \wedge 0 \leq i + 1 \leq |e\_p| \wedge_L sumadorp + e\_p[i] = \sum_{n=0}^{i} e\_p[n] \equiv$
$True \wedge_L 0 \leq i < |e\_p| \wedge_L P \wedge 0 \leq i + 1 \leq |e\_p| \wedge_L sumadorp = \sum_{n=0}^{i} e\_p[n] - e\_p[i] \equiv$
$0 \leq i < |e\_p| \wedge_L P \wedge_L sumadorp = \sum_{n=0}^{i-1} e\_p[n] \equiv$
$I \wedge B_{c_1}$

3. $I \wedge \neg B_{c_1} \longrightarrow Q_{c_1}$**:**
$I \equiv P \wedge 0 \leq i \leq |e\_p| \wedge_L sumadorp = \sum_{n=0}^{i-1} e\_p[n]$
$B_{c_1} \equiv i < |e\_p|$
$Q_{c_1} \equiv P \wedge i = |e\_p| \wedge sumadorp = \sum_{n=0}^{|e\_p|-1} e\_p[n]$

$I \wedge \neg B_{c_1} \equiv P \wedge 0 \leq i \leq |e\_p| \wedge_L sumadorp = \sum_{n=0}^{i-1} e\_p[n] \wedge i \geq |e\_p|$

Como $i \leq |e\_p| \wedge i \geq |e\_p|$, entonces $i = |e\_p|$. La expresión queda:
$I \wedge \neg B_{c_1} \equiv P \wedge i = |e\_p| \wedge sumadorp = \sum_{n=0}^{|e\_p|-1} e\_p[n]$ **que es equivalente a Q_{c_1}, por lo tanto es verdadero que** $I \wedge \neg B_{c_1} \longrightarrow Q_{c_1}$

4. $\{I \wedge B_{c_1} \wedge v_0 = fv\}S\{fv < v_0\} \equiv I \wedge B_{c_1} \wedge v_0 = fv \longrightarrow wp(S_1; S_2, fv < v_0)$**:**
$I \equiv P \wedge 0 \leq i \leq |e\_p| \wedge_L sumadorp = \sum_{n=0}^{i-1} e\_p[n]$
$B_{c_1} \equiv i < |e\_p|$
$fv = |e\_p| - i$

$I \wedge B_{c_1} \wedge fv = v_0 \equiv P \wedge 0 \leq i \leq |e\_p| \wedge_L sumadorp = \sum_{n=0}^{i-1} e\_p[n] \wedge i < |e\_p| \wedge |e\_p| - i = v_0 \equiv$
$P \wedge 0 \leq i < |e\_p| \wedge_L sumadorp = \sum_{n=0}^{i-1} e\_p[n] \wedge |e\_p| - i = v_0$

$wp(S_1; S_2, fv < v_0) \equiv wp(S_1, wp(S_2, fv < v_0))$

$wp(S_2, fv < v_0) \equiv wp(i := i + 1, fv < v_0) \equiv$
$def(i + 1) \wedge_L |e\_p| - (i + 1) < v_0 \equiv$
$True \wedge_L |e\_p| - i - 1 < v_0 \equiv$
$|e\_p| - i - 1 < v_0 \equiv E_2$

$wp(S_1, E_2) \equiv wp(sumadorp := sumadorp + e\_p[i], |e\_p| - i - 1 < v_0) \equiv$
$def(sumadorp) \wedge_L def(e\_p[i]) \wedge_L |e\_p| - i - 1 < v_0 \equiv$
$True \wedge_L 0 \leq i < |e\_p| \wedge_L |e\_p| - i - 1 < v_0 \equiv$
$0 \leq i < |e\_p| \wedge_L |e\_p| - i - 1 < v_0$

Tenemos de $I \wedge B_{c_1} \wedge v_0 = fv$ que $0 \leq i < |e\_p|$ y que $|e\_p| - i = v_0$. Asumimos esto como verdadero y queda:
$True \wedge_L v_0 - 1 < v_0 \equiv$
$True \wedge_L True \equiv$
$True$

5. $I \wedge fv \leq 0 \longrightarrow \neg B_{c_1}$**:**
$I \equiv P \wedge 0 \leq i \leq |e\_p| \wedge_L sumadorp = \sum_{n=0}^{i-1} e\_p[n]$
$fv \leq 0 \equiv |e\_p| - i \leq 0$
$B_{c_1} \equiv i < |e\_p|$

$I \wedge fv \leq 0 \equiv$
$I \wedge |e\_p| - i \leq 0 \equiv$
$I \wedge i \geq |e\_p| \equiv$
$I \wedge \neg B_{c_1}$
$I \wedge \neg B_{c_1} \longrightarrow \neg B_{c_1}$ **es tautología.**

Pasamos a las siguientes dos líneas y el ciclo posterior:

```
sumadors:=0;
j:=0;
while (j < escrutinio_senadores.size()) do
    sumadors := sumadors + escrutinio_senadores[j];
    j := j + 1
endwhile
```

$P_{c_2} \equiv Q_{c_1} \wedge sumadors = 0 \wedge j = 0$
$Q_{c_2} \equiv Q_{c_1} \wedge sumadors = \sum_{m=0}^{|e\_s|} e\_s[m] \wedge j = |e\_s|$

Planteamos: ¿$Q_{c_1} \longrightarrow wp(sumadors := 0; j := 0, P_{c_2})$**?**
$wp(sumadors := 0; j := 0, P_{c_2}) \equiv wp(sumadors := 0, wp(j := 0, P_{c_2}))$

$wp(j := 0, P_{c_2}) \equiv$
$wp(j := 0, Q_{c_1} \wedge sumadors = 0 \wedge j = 0) \equiv$
$def(0) \wedge_L Q_{c_1} \wedge sumadors = 0 \wedge 0 = 0 \equiv$
$Q_{c_1} \wedge sumadors = 0$

$wp(sumadors := 0, Q_{c_1} \wedge sumadors = 0) \equiv$
$def(0) \wedge_L Q_{c_1} \wedge 0 = 0) \equiv$
Q_{c_1}

Nos queda $Q_{c_1} \longrightarrow Q_{c_1}$ **que es verdadero.**

Paso a probar $P_{c_2} \longrightarrow Q_{c_2}$ **definiendo un invariante y una función variante. Además, defino** B_{c_2} **como la guarda del while:**
$I \equiv Q_{c_1} \wedge 0 \leq j \leq |e\_s| \wedge_L sumadors = \sum_{m=0}^{j-1} e\_s[m]$
$fv = |e\_s| - j$
$B_{c_1} \equiv j < |e\_s|$

1. $P_{c_2} \longrightarrow I$**:**
$P_{c_2} \equiv Q_{c_1} \wedge sumadors = 0 \wedge j = 0$
$I \equiv Q_{c_1} \wedge 0 \leq j \leq |e\_s| \wedge_L sumadors = \sum_{m=0}^{j-1} e\_s[m]$

Si asumimos P_{c_2} **verdadero,** I **nos queda:**
$True \wedge 0 \leq 0 \leq |e\_s| \wedge_L 0 = \sum_{m=0}^{0-1} e\_s[m] \equiv$
$True \wedge True \wedge_L True \equiv$
$True$

2. $\{I \wedge B_{c_2}\}S\{I\} \equiv I \wedge B_{c_2} \longrightarrow wp(S1; S2, I)$**:**
$I \equiv Q_{c_1} \wedge 0 \leq j \leq |e\_s| \wedge_L sumadors = \sum_{m=0}^{j-1} e\_s[m]$
$B_{c_2} \equiv j < |e\_s|$

$wp(S1; S2, I) \equiv wp(S1, wp(S2, I))$

$wp(S2, I) \equiv$
$wp(j := j + 1, Q_{c_1} \wedge 0 \leq j \leq |e\_s| \wedge_L sumadors = \sum_{m=0}^{j-1} e\_s[m]) \equiv$
$def(j + 1) \wedge_L Q_{c_1} \wedge 0 \leq j + 1 \leq |e\_s| \wedge_L sumadors = \sum_{m=0}^{j} e\_s[m] \equiv$
$Q_{c_1} \wedge 0 \leq j + 1 \leq |e\_s| \wedge_L sumadors = \sum_{m=0}^{j} e\_s[m] \equiv E_2$

$wp(S1, E_2) \equiv$
$wp(sumadors := sumadors + e\_s[j], Q_{c_1} \land 0 \leq j+1 \leq |e\_s| \land_L sumadors = \sum_{m=0}^{j} e\_s[m]) \equiv$
$def(sumadors) \land_L def(e\_s[j]) \land_L Q_{c_1} \land 0 \leq j+1 \leq |e\_s| \land_L sumadors + e\_s[j] = \sum_{m=0}^{j} e\_s[m] \equiv$
$True \land_L 0 \leq j < |e\_s| \land_L Q_{c_1} \land 0 \leq j+1 \leq |e\_s| \land_L sumadors = \sum_{m=0}^{j} e\_s[m] - e\_s[j] \equiv$
$0 \leq j < |e\_s| \land_L Q_{c_1} \land_L sumadors = \sum_{m=0}^{j-1} e\_s[m] \equiv$
$I \land B_{c_2}$

3. $I \land \neg B_{c_2} \longrightarrow Q_{c_2}$**:**
$I \equiv Q_{c_1} \land 0 \leq j \leq |e\_s| \land_L sumadors = \sum_{m=0}^{j-1} e\_s[m]$
$B_{c_2} \equiv j < |e\_s|$
$Q_{c_2} \equiv Q_{c_1} \land sumadors = \sum_{m=0}^{|e\_s|} e\_s[m] \land j = |e\_s|$

$I \land \neg B_{c_2} \equiv Q_{c_1} \land 0 \leq j \leq |e\_s| \land_L sumadors = \sum_{m=0}^{j-1} e\_s[m] \land j \geq |e\_s|$

Como $j \leq |e\_s| \land j \geq |e\_s|$**, entonces** $j = |e\_s|$**. La expresión queda:**
$I \land \neg B_{c_2} \equiv Q_{c_1} \land j = |e\_s| \land sumadors = \sum_{m=0}^{|e\_s|-1} e\_s[m]$ **que es equivalente a** Q_{c_2}**, por lo tanto es verdadero que** $I \land \neg B_{c_2} \longrightarrow Q_{c_2}$

4. $\{I \land B_{c_2} \land v_0 = fv\} S \{fv < v_0\} \equiv I \land B_{c_2} \land v_0 = fv \longrightarrow wp(S_1; S_2, fv < v_0)$**:**
$I \equiv Q_{c_1} \land 0 \leq j \leq |e\_s| \land_L sumadors = \sum_{m=0}^{j-1} e\_s[m]$
$B_{c_2} \equiv j < |e\_s|$
$fv = |e\_s| - j$

$I \land B_{c_2} \land fv = v_0 \equiv Q_{c_1} \land 0 \leq j \leq |e\_s| \land_L sumadors = \sum_{m=0}^{j-1} e\_s[m] \land j < |e\_s| \land |e\_s| - j = v_0 \equiv$
$Q_{c_1} \land 0 \leq j < |e\_s| \land_L sumadors = \sum_{m=0}^{j-1} e\_s[m] \land |e\_s| - j = v_0$

$wp(S_1; S_2, fv < v_0) \equiv wp(S_1, wp(S_2, fv < v_0))$

$wp(S_2, fv < v_0) \equiv wp(j := j+1, fv < v_0) \equiv$
$def(j+1) \land_L |e\_s| - (j+1) < v_0 \equiv$
$True \land_L |e\_s| - j - 1 < v_0 \equiv$
$|e\_s| - j - 1 < v_0 \equiv E_2$

$wp(S_1, E_2) \equiv wp(sumadors := sumadors + e\_s[j], |e\_s| - j - 1 < v_0) \equiv$
$def(sumadors) \land_L def(e\_s[j]) \land_L |e\_s| - j - 1 < v_0 \equiv$
$True \land_L 0 \leq j < |e\_s| \land_L |e\_s| - j - 1 < v_0 \equiv$
$0 \leq j < |e\_s| \land_L |e\_s| - j - 1 < v_0$

Tenemos de $I \land B_{c_2} \land v_0 = fv$ **que** $0 \leq j < |e\_s|$ **y que** $|e\_s| - j = v_0$**. Asumimos esto como verdadero y queda:**
$True \land_L v_0 - 1 < v_0 \equiv$
$True \land_L True \equiv$
$True$

5. $I \land fv \leq 0 \longrightarrow \neg B_{c_2}$**:**
$I \equiv Q_{c_1} \land 0 \leq j \leq |e\_s| \land_L sumadors = \sum_{m=0}^{j-1} e\_s[m]$
$fv \leq 0 \equiv |e\_s| - j \leq 0$
$B_{c_2} \equiv j < |e\_s|$

$I \land fv \leq 0 \equiv$
$I \land |e\_s| - j \leq 0 \equiv$
$I \land j \geq |e\_s| \equiv$
$I \land \neg B_{c_2}$
$I \land \neg B_{c_2} \longrightarrow \neg B_{c_2}$ **es tautología.**

Pasamos a las siguientes dos líneas y al último ciclo del programa:

```
sumadord := 0;
k := 0;
while (k < escrutinio_diputados.size()) do
    sumadord := sumadord + escrutinio_diputados[k];
    k := k + 1
endwhile
```

$P_{c_3} \equiv Q_{c_2} \wedge sumadord = 0 \wedge k = 0$
$Q_{c_3} \equiv Q_{c_2} \wedge sumadord = \sum_{r=0}^{|e\_d|} e\_d[r] \wedge k = |e\_d|$

Planteamos: ¿$Q_{c_2} \longrightarrow wp(sumadord := 0; k := 0, P_{c_3})$?
$wp(sumadord := 0; k := 0, P_{c_3}) \equiv wp(sumadord := 0, wp(k := 0, P_{c_3}))$

$wp(k := 0, P_{c_3}) \equiv$
$wp(k := 0, Q_{c_2} \wedge sumadord = 0 \wedge k = 0) \equiv$
$def(0) \wedge_L Q_{c_2} \wedge sumadord = 0 \wedge 0 = 0 \equiv$
$Q_{c_2} \wedge sumadord = 0$

$wp(sumadord := 0, Q_{c_2} \wedge sumadord = 0) \equiv$
$def(0) \wedge Q_{c_2} \wedge 0 = 0 \equiv$
Q_{c_2}

Me queda $Q_{c_2} \longrightarrow Q_{c_2}$ que es verdadero.

Paso a probar $P_{c_3} \longrightarrow Q_{c_3}$ definiendo un invariante y una función variante. Además, defino B_{c_3} como la guarda del while:
$I \equiv Q_{c_2} \wedge 0 \leq k \leq |e\_d| \wedge_L sumadord = \sum_{r=0}^{k-1} e\_d[r]$
$fv = |e\_d| - k$
$B_{c_1} \equiv k < |e\_d|$

1. $P_{c_3} \longrightarrow I$:
$P_{c_3} \equiv Q_{c_2} \wedge sumadord = 0 \wedge k = 0$
$I \equiv Q_{c_2} \wedge 0 \leq k \leq |e\_d| \wedge_L sumadord = \sum_{r=0}^{k-1} e\_d[r]$

Si asumimos P_{c_3} verdadero, I nos queda:
$True \wedge 0 \leq 0 \leq |e\_d| \wedge_L 0 = \sum_{r=0}^{0-1} e\_s[r] \equiv$
$True \wedge True \wedge_L True \equiv$
$True$

2. $\{I \wedge B_{c_3}\}S\{I\} \equiv I \wedge B_{c_3} \longrightarrow wp(S1; S2, I)$:
$I \equiv Q_{c_2} \wedge 0 \leq k \leq |e\_d| \wedge_L sumadord = \sum_{r=0}^{k-1} e\_d[r]$
$B_{c_3} \equiv k < |e\_d|$

$wp(S1; S2, I) \equiv wp(S1, wp(S2, I))$

$wp(S2, I) \equiv$
$wp(k := k + 1, Q_{c_2} \wedge 0 \leq k \leq |e\_d| \wedge_L sumadord = \sum_{r=0}^{k-1} e\_d[r]) \equiv$
$def(k + 1) \wedge_L Q_{c_2} \wedge 0 \leq k + 1 \leq |e\_d| \wedge_L sumadord = \sum_{r=0}^{k} e\_d[r] \equiv$
$Q_{c_2} \wedge 0 \leq k + 1 \leq |e\_d| \wedge_L sumadord = \sum_{r=0}^{k} e\_d[r] \equiv E_2$

$wp(S1, E_2) \equiv$
$wp(sumadord := sumadord + e\_d[k], Q_{c_2} \wedge 0 \leq k + 1 \leq |e\_d| \wedge_L sumadord = \sum_{r=0}^{k} e\_d[r]) \equiv$
$def(sumadord) \wedge_L def(e\_d[k]) \wedge_L Q_{c_2} \wedge 0 \leq k + 1 \leq |e\_d| \wedge_L sumadord + e\_d[k] = \sum_{r=0}^{k} e\_d[r] \equiv$
$True \wedge_L 0 \leq k < |e\_d| \wedge_L Q_{c_2} \wedge 0 \leq k + 1 \leq |e\_d| \wedge_L sumadord = \sum_{r=0}^{k} e\_s[r] - e\_d[k] \equiv$
$0 \leq k < |e\_d| \wedge_L Q_{c_2} \wedge_L sumadord = \sum_{r=0}^{k-1} e\_d[r] \equiv$
$I \wedge B_{c_3}$

3. $I \wedge \neg B_{c_3} \longrightarrow Q_{c_3}$:
$I \equiv Q_{c_2} \wedge 0 \leq k \leq |e\_d| \wedge_L sumadord = \sum_{r=0}^{k-1} e\_d[r]$
$B_{c_3} \equiv k < |e\_d|$
$Q_{c_3} \equiv Q_{c_2} \wedge sumadord = \sum_{r=0}^{|e\_d|} e\_d[r] \wedge k = |e\_d|$

$I \wedge \neg B_{c_3} \equiv Q_{c_2} \wedge 0 \leq k \leq |e\_d| \wedge_L sumadord = \sum_{r=0}^{k-1} e\_d[r] \wedge k \geq |e\_d|$

Como $k \leq |e\_d| \wedge j \geq |e\_d|$, entonces $k = |e\_d|$. La expresión queda:
$I \wedge \neg B_{c_3} \equiv Q_{c_2} \wedge k = |e\_d| \wedge sumadord = \sum_{r=0}^{|e\_d|-1} e\_d[r]$ **que es equivalente a Q_{c_3}, por lo tanto es verdadero que $I \wedge \neg B_{c_3} \longrightarrow Q_{c_3}$**

4. $\{I \wedge B_{c_3} \wedge v_0 = fv\}S\{fv < v_0\} \equiv I \wedge B_{c_3} \wedge v_0 = fv \longrightarrow wp(S_1; S_2, fv < v_0)$**:**
$I \equiv Q_{c_2} \wedge 0 \leq k \leq |e\_d| \wedge_L sumadord = \sum_{r=0}^{k-1} e\_d[r]$
$B_{c_3} \equiv k < |e\_d|$
$fv = |e\_d| - k$

$I \wedge B_{c_3} \wedge fv = v_0 \equiv Q_{c_2} \wedge 0 \leq k \leq |e\_d| \wedge_L sumadord = \sum_{r=0}^{k-1} e\_d[r] \wedge k < |e\_d| \wedge |e\_d| - k = v_0 \equiv$
$Q_{c_2} \wedge 0 \leq k < |e\_d| \wedge_L sumadord = \sum_{r=0}^{k-1} e\_d[r] \wedge |e\_d| - k = v_0$

$wp(S_1; S_2, fv < v_0) \equiv wp(S_1, wp(S_2, fv < v_0))$

$wp(S_2, fv < v_0) \equiv wp(k := k + 1, fv < v_0) \equiv$
$def(k + 1) \wedge_L |e\_d| - (k + 1) < v_0 \equiv$
$True \wedge_L |e\_d| - k - 1 < v_0 \equiv$
$|e\_d| - k - 1 < v_0 \equiv E_2$

$wp(S_1, E_2) \equiv wp(sumadord := sumadord + e\_d[k], |e\_d| - k - 1 < v_0) \equiv$
$def(sumadord) \wedge_L def(e\_d[k]) \wedge_L |e\_d| - k - 1 < v_0 \equiv$
$True \wedge_L 0 \leq k < |e\_d| \wedge_L |e\_d| - k - 1 < v_0 \equiv$
$0 \leq k < |e\_d| \wedge_L |e\_d| - k - 1 < v_0$

Tenemos de $I \wedge B_{c_3} \wedge v_0 = fv$ que $0 \leq k < |e\_d|$ y que $|e\_d| - k = v_0$. Asumimos esto como verdadero y queda:
$True \wedge_L v_0 - 1 < v_0 \equiv$
$True \wedge_L True \equiv$
$True$

5. $I \wedge fv \leq 0 \longrightarrow \neg B_{c_3}$**:**
$I \equiv Q_{c_2} \wedge 0 \leq k \leq |e\_d| \wedge_L sumadord = \sum_{r=0}^{k-1} e\_d[r]$
$fv \leq 0 \equiv |e\_d| - k \leq 0$
$B_{c_3} \equiv k < |e\_d|$

$I \wedge fv \leq 0 \equiv$
$I \wedge |e\_d| - k \leq 0 \equiv$
$I \wedge k \geq |e\_d| \equiv$
$I \wedge \neg B_{c_3}$
$I \wedge \neg B_{c_3} \longrightarrow \neg B_{c_3}$ **es tautología.**

Pasamos al if final:

```
if (sumadors != sumadorp || sumadors != sumadord || sumadorp != sumadord) then
   res := True;
else
    res := False;
endif
```

Definimos Q como el asegura del programa:
$Q \equiv res = True \leftrightarrow sumarEltos(e\_s) \neq sumarEltos(e\_p) \vee sumarEltos(e\_s) \neq sumarEltos(e\_d) \vee sumarEltos(e\_p) \neq sumarEltos(e\_d) \equiv$
$res = True \leftrightarrow \sum_{i=0}^{|e\_s|-1} e\_s[i] \neq \sum_{i=0}^{|e\_p|-1} e\_p[i] \wedge \sum_{i=0}^{|e\_s|-1} e\_s[i] \neq \sum_{i=0}^{|e\_d|-1} e\_d[i] \wedge \sum_{i=0}^{|e\_p|-1} e\_p[i] \neq \sum_{i=0}^{|e\_d|-1} e\_d[i]$

Si probamos que $Q_{c_3} \longrightarrow wp(if..then..else..fi, Q)$, ya nos alcanza para probar la correctitud del programa completo.
$wp(if..then..else..fi, Q) \equiv$
$def(B) \wedge_L (B \wedge wp(S1, Q)) \vee (\neg B \wedge wp(S2, Q))$ (B es la guarda del if) $\equiv$
$def(sumadorp) \wedge_L def(sumadors) \wedge_L def(sumadord) \wedge_L ((sumadors \neq sumadorp \vee sumadors \neq sumadord \vee sumadorp \neq sumadord \wedge wp(S1, Q)) \vee (sumadors = sumadorp \wedge sumadors = sumadord \wedge sumadorp = sumadord \wedge wp(S2, Q))) \equiv$
$(sumadors \neq sumadorp \vee sumadors \neq sumadord \vee sumadorp \neq sumadord \wedge wp(S1, Q)) \vee (sumadors = sumadorp \wedge sumadors = sumadord \wedge sumadorp = sumadord \wedge wp(S2, Q))$

$wp(S1, Q) \equiv wp(res := True, res = True \leftrightarrow \sum_{i=0}^{|e\_s|-1} e\_s[i] \neq \sum_{i=0}^{|e\_p|-1} e\_p[i] \wedge \sum_{i=0}^{|e\_s|-1} e\_s[i] \neq \sum_{i=0}^{|e\_d|-1} e\_d[i] \wedge \sum_{i=0}^{|e\_p|-1} e\_p[i] \neq \sum_{i=0}^{|e\_d|-1} e\_d[i]) \equiv def(True) \wedge_L$
$True = True \leftrightarrow \sum_{i=0}^{|e\_s|-1} e\_s[i] \neq \sum_{i=0}^{|e\_p|-1} e\_p[i] \wedge \sum_{i=0}^{|e\_s|-1} e\_s[i] \neq \sum_{i=0}^{|e\_d|-1} e\_d[i] \wedge \sum_{i=0}^{|e\_p|-1} e\_p[i] \neq \sum_{i=0}^{|e\_d|-1} e\_d[i]) \equiv$
$\sum_{i=0}^{|e\_s|-1} e\_s[i] \neq \sum_{i=0}^{|e\_p|-1} e\_p[i] \wedge \sum_{i=0}^{|e\_s|-1} e\_s[i] \neq \sum_{i=0}^{|e\_d|-1} e\_d[i] \wedge \sum_{i=0}^{|e\_p|-1} e\_p[i] \neq \sum_{i=0}^{|e\_d|-1} e\_d[i])$

$wp(S2,Q) \equiv wp(res := False, res = True \leftrightarrow \sum_{i=0}^{|e\_s|-1} e\_s[i] \neq \sum_{i=0}^{|e\_p|-1} e\_p[i] \wedge \sum_{i=0}^{|e\_s|-1} e\_s[i] \neq \sum_{i=0}^{|e\_d|-1} e\_d[i] \wedge \sum_{i=0}^{|e\_p|-1} e\_p[i] \neq \sum_{i=0}^{|e\_d|-1} e\_d[i]) \equiv def(False) \wedge_L$
$False = True \leftrightarrow \sum_{i=0}^{|e\_s|-1} e\_s[i] \neq \sum_{i=0}^{|e\_p|-1} e\_p[i] \wedge \sum_{i=0}^{|e\_s|-1} e\_s[i] \neq \sum_{i=0}^{|e\_d|-1} e\_d[i] \wedge \sum_{i=0}^{|e\_p|-1} e\_p[i] \neq \sum_{i=0}^{|e\_d|-1} e\_d[i]) \equiv$
$\sum_{i=0}^{|e\_s|-1} e\_s[i] = \sum_{i=0}^{|e\_p|-1} e\_p[i] \vee \sum_{i=0}^{|e\_s|-1} e\_s[i] = \sum_{i=0}^{|e\_d|-1} e\_d[i] \vee \sum_{i=0}^{|e\_p|-1} e\_p[i] = \sum_{i=0}^{|e\_d|-1} e\_d[i])$

La expresión que me queda entonces es:
$wp(if..then..else..fi, Q) \equiv (sumadors \neq sumadorp \vee sumadors \neq sumadord \vee sumadorp \neq sumadord \wedge$
$\sum_{i=0}^{|e\_s|-1} e\_s[i] \neq \sum_{i=0}^{|e\_p|-1} e\_p[i] \wedge \sum_{i=0}^{|e\_s|-1} e\_s[i] \neq \sum_{i=0}^{|e\_d|-1} e\_d[i] \wedge \sum_{i=0}^{|e\_p|-1} e\_p[i] \neq \sum_{i=0}^{|e\_d|-1} e\_d[i]) \vee$
$(sumadors = sumadorp \wedge sumadors = sumadord \wedge sumadorp = sumadord \wedge$
$\sum_{i=0}^{|e\_s|-1} e\_s[i] = \sum_{i=0}^{|e\_p|-1} e\_p[i] \vee \sum_{i=0}^{|e\_s|-1} e\_s[i] = \sum_{i=0}^{|e\_d|-1} e\_d[i] \vee \sum_{i=0}^{|e\_p|-1} e\_p[i] = \sum_{i=0}^{|e\_d|-1} e\_d[i])$

¿$Q_{c_3} \longrightarrow wp(if..then..else..fi, Q)$? **Vamos por partes:**

Agarramos la siguiente expresión de Q_{c_3}, que es más débil que esta:
$E_1 \equiv sumadorp = \sum_{n=0}^{|e\_p|} e\_s[n] \wedge sumadors = \sum_{m=0}^{|e\_s|} e\_s[m] \wedge sumadord = \sum_{r=0}^{|e\_s|} e\_s[r].$

Definimos R de la siguiente manera:
$R \equiv \sum_{i=0}^{|e\_s|-1} e\_s[i] \neq \sum_{i=0}^{|e\_p|-1} e\_p[i] \wedge \sum_{i=0}^{|e\_s|-1} e\_s[i] \neq \sum_{i=0}^{|e\_d|-1} e\_d[i] \wedge \sum_{i=0}^{|e\_p|-1} e\_p[i] \neq \sum_{i=0}^{|e\_d|-1} e\_d[i]$

Tenemos entonces que:
$\neg R \equiv \sum_{i=0}^{|e\_s|-1} e\_s[i] = \sum_{i=0}^{|e\_p|-1} e\_p[i] \vee \sum_{i=0}^{|e\_s|-1} e\_s[i] = \sum_{i=0}^{|e\_d|-1} e\_d[i] \vee \sum_{i=0}^{|e\_p|-1} e\_p[i] = \sum_{i=0}^{|e\_d|-1} e\_d[i]$

Se puede ver que si reemplazamos los valores de sumadorp, sumadors y sumadord por los de E_1 en $wp(if..then..else..fi, Q)$, me queda una expresión de la forma $(R \wedge R) \vee (\neg R \wedge \neg R) \equiv R \vee \neg R \equiv True$

Por lo tanto, probamos que $Q_{c_3} \longrightarrow E_1 \longrightarrow wp(if..then..else..fi, Q)$ y por lo tanto $Q_{c_3} \longrightarrow wp(if..then..else..fi, Q)$

2.3. Weakest Precondition de obtenerSenadoresEnProvincia

(definimos escr = escrutinio por comodidad en las sumatorias)
Primera parte:

```
i:=1
indiceMax1:=0
while (i < escrutinio.size()−1) do
    if (escrutinio[i] > escrutinio[indiceMax1]) then
        indiceMax1:=i
    else
        skip
    endif
    i := i+1
endwhile
```

Definimos P como el requiere del programa:
$P \equiv escrutinioValido(escr)$

$P_{c_1} \equiv P \wedge i = 1 \wedge indiceMax1 = 0$

$Q_{c_1} \equiv P \wedge i = |escr| - 1 \wedge_L esMax(indiceMax1, escr, |escr| - 1)$

$B_{c_1} \equiv i < |escr| - 1$

$I \equiv P \wedge ((1 \leq i < |escr| - 1 \wedge_L ((escr[i] > escr[indiceMax1] \wedge esMax(i, escr, i+1)) \vee (escr[i] \leq escr[indiceMax1] \wedge esMax(indiceMax1, escr, i+1)))) \vee (i = |escr| - 1 \wedge esMax(indiceMax1, escr, i)))$

$fv = (|escr| - 1) - i$

1. P $\longrightarrow_L$ wp(i:=1;indiceMax1:=0,P_{c_1}):
$P \equiv escrutinioValido(escr)$
$P_{c_1} \equiv P \wedge i = 1 \wedge indiceMax1 = 0$

$wp(i := 1; indiceMax1 := 0, P_{c_1}) \equiv wp(i := 1, wp(indiceMax1 := 0, P_{c_1}))$

$wp(indiceMax1 := 0, P_{c_1}) \equiv$
$def(0) \wedge_L P \wedge i = 1 \wedge 0 = 0 \equiv$
$P \wedge i = 1$

$wp(i := 1, P \wedge i = 1) \equiv$
$def(1) \wedge P \wedge 1 = 1 \equiv$
P

Me queda $P \longrightarrow_L P$ que es verdadero.

2. $P_{c_1} \longrightarrow_L$ wp(while.., Q_c):

2.1 $P_{c_1} \longrightarrow_L$ I:
$P_{c_1} \equiv P \wedge i = 1 \wedge indiceMax1 = 0$
$I \equiv P \wedge ((1 \leq i < |escr| - 1 \wedge_L ((escr[i] > escr[indiceMax1] \wedge esMax(i, escr, i + 1)) \vee (escr[i] \leq escr[indiceMax1] \wedge esMax(indiceMax1, escr, i + 1)))) \vee (i = |escr| - 1 \wedge esMax(indiceMax1, escr, i)))$

Si asumimos P_{c_1} verdadero, entonces el invariante queda:
$True \wedge ((1 \leq 1 < |escr| - 1 \wedge_L ((escr[1] > escr[0] \wedge esMax(1, escr, 2)) \vee (escr[1] \leq escr[0] \wedge esMax(0, escr, 1)))) \vee (1 = |escr| - 1 \wedge esMax(0, escr, 1))) \equiv (1 = |escr| - 1 \equiv False$ porque $escrutinioValido$ me dice que $|escr| \geq 3)$
$(escr[1] > escr[0] \wedge esMax(1, escr, 2)) \vee (escr[1] \leq escr[0] \wedge esMax(0, escr, 1)) \equiv E_1$

$esMax(1, escr, 2) \equiv 0 \leq 1 < 2 \wedge_L (\forall k : \mathbb{Z})\ (0 \leq k < 2 \longrightarrow_L escr[1] \geq escr[k]) \equiv escr[1] \geq escr[0]$

$esMax(0, escr, 2) \equiv 0 \leq 0 < 2 \wedge_L (\forall k : \mathbb{Z})\ (0 \leq k < 2 \longrightarrow_L escr[0] \geq escr[k]) \equiv escr[0] \geq escr[1]$

$E_1 \equiv escr[1] \geq escr[0] \vee escr[0] \geq escr[1] \equiv True$

2.2 $I \wedge \neg B_{c_1} \longrightarrow_L Q_{c_1}$:
$I \equiv P \wedge ((1 \leq i < |escr| - 1 \wedge_L ((escr[i] > escr[indiceMax1] \wedge esMax(i, escr, i + 1)) \vee (escr[i] \leq escr[indiceMax1] \wedge esMax(indiceMax1, escr, i + 1)))) \vee (i = |escr| - 1 \wedge esMax(indiceMax1, escr, i)))$
$\neg B_{c_1} \equiv i \geq |escr| - 1$
$Q_{c_1} \equiv P \wedge i = |escr| - 1 \wedge_L esMax(indiceMax1, escr, |escr| - 1)$

En $I \wedge \neg B_{c_1}$, tengo que $1 \leq i \leq |escr| - 1 \wedge i \geq |escr| - 1$, por lo tanto $i = |escr| - 1$

Si asumimos $I \wedge \neg B_{c_1}$ verdadero, Q_{c_1} queda así:
$True \wedge ((1 \leq i < i \wedge_L ...) \vee (i = |escr| - 1 \wedge esMax(indiceMax1, escr, |escr| - 1))) \equiv$
$True \wedge (False \vee (i = |escr| - 1 \wedge esMax(indiceMax1, escr, |escr| - 1))) \equiv$
$i = |escr| - 1 \wedge esMax(indiceMax1, escr, |escr| - 1) \equiv$
$True \wedge True \equiv$
$True$

2.3 $I \wedge$ fv $\leq$ 0 $\longrightarrow \neg B_{c_1}$
$I \equiv P \wedge ((1 \leq i < |escr| - 1 \wedge_L ((escr[i] > escr[indiceMax1] \wedge esMax(i, escr, i + 1)) \vee (escr[i] \leq escr[indiceMax1] \wedge esMax(indiceMax1, escr, i + 1)))) \vee (i = |escr| - 1 \wedge esMax(indiceMax1, escr, i)))$
$fv = (|escr| - 1) - i$
$\neg B_{c_1} \equiv i < |escr| - 1$

$I \wedge (|escr| - 1) - i \leq 0 \longrightarrow \neg B$
$I \wedge |escr| - 1 \leq i \longrightarrow \neg B$
Pero $|escr| - 1 \leq i \equiv \neg B$, y además $I \wedge \neg B \longrightarrow \neg B$ es tautología.

2.4 {$I \wedge B_{c_1}$} S {I}
$I \equiv P \wedge ((1 \leq i < |escr| - 1 \wedge_L ((escr[i] > escr[indiceMax1] \wedge esMax(i, escr, i + 1)) \vee (escr[i] \leq escr[indiceMax1] \wedge esMax(indiceMax1, escr, i + 1)))) \vee (i = |escr| - 1 \wedge esMax(indiceMax1, escr, i)))$
$B_{c_1} \equiv i < |escr| - 1$

Entonces: $I \wedge B_{c_1} \equiv P \wedge (1 \leq i < |escr| - 1 \wedge_L ((escr[i] > escr[indiceMax1] \wedge esMax(i, escr, i + 1)) \vee (escr[i] \leq escr[indiceMax1] \wedge esMax(indiceMax1, escr, i + 1))))$

Veo si $\{I \wedge B_{c_1}\} \longrightarrow wp(if...then...else...fi; i := i+1, I)$:
$wp(i := i+1; I) \equiv$
$def(i+1) \wedge_L P \wedge ((1 \leq i+1 < |escr|-1 \wedge_L ((escr[i+1] > escr[indiceMax1] \wedge esMax(i+1, escr, i+2)) \vee (escr[i+1] \leq escr[indiceMax1] \wedge esMax(indiceMax1, escr, i+2)))) \vee (i+1 = |escr|-1 \wedge esMax(indiceMax1, escr, i+1))) \equiv$
$P \wedge ((1 \leq i+1 < |escr|-1 \wedge_L ((escr[i+1] > escr[indiceMax1] \wedge esMax(i+1, escr, i+2)) \vee (escr[i+1] \leq escr[indiceMax1] \wedge esMax(indiceMax1, escr, i+2)))) \vee (i+1 = |escr|-1 \wedge esMax(indiceMax1, escr, i+1))) \equiv E_1$

Ahora resuelvo $wp(if...then...else..fi, E_1) \equiv def(B_1) \wedge_L (((B_1) \wedge wp(S_1, E_1)) \vee (\neg(B_1) \wedge wp(S_2, E_1)))$ **siendo B_1 la guarda del if:**
$B_1 \equiv escr[i] > escr[indiceMax1]$

$wp(if...then...else..fi, E_1) \equiv def(escr[i]) \wedge_L def(escr[indiceMax1]) \wedge_L ((escr[i] > escr[indiceMax1] \wedge wp(S_1, E_1)) \vee (escr[i] \leq escr[indiceMax1] \wedge wp(S_2, E_1))) \equiv$
$0 \leq i < |escr| \wedge_L 0 \leq indiceMax1 < |escr| \wedge_L ((escr[i] > escr[indiceMax1] \wedge wp(S_1, E_1)) \vee (escr[i] \leq escr[indiceMax1] \wedge wp(S_2, E_1)))$

$wp(S_1, E_1) \equiv wp(indiceMax1 := i, E_1)$

$wp(indiceMax1 := i, E_1) \equiv$
$def(i) \wedge_L P \wedge ((1 \leq i+1 < |escr|-1 \wedge_L ((escr[i+1] > escr[i] \wedge esMax(i+1, escr, i+2)) \vee (escr[i+1] \leq escr[i] \wedge esMax(i, escr, i+2)))) \vee (i+1 = |escr|-1 \wedge esMax(i, escr, i+1))) \equiv$
$P \wedge ((1 \leq i+1 < |escr|-1 \wedge_L ((escr[i+1] > escr[i] \wedge esMax(i+1, escr, i+2)) \vee (escr[i+1] \leq escr[i] \wedge esMax(i, escr, i+2)))) \vee (i+1 = |escr|-1 \wedge esMax(i, escr, i+1))) \equiv E_2$

$wp(S_2, E_1) \equiv wp(skip, E_1) \equiv E_1$

$I \wedge B_{c_1} \equiv P \wedge (1 \leq i < |escr|-1 \wedge_L ((escr[i] > escr[indiceMax1] \wedge esMax(i, escr, i+1)) \vee (escr[i] \leq escr[indiceMax1] \wedge esMax(indiceMax1, escr, i+1))))$

¿$I \wedge B_{c_1} \longrightarrow wp(if..then..else..fi, E_1)$?
Vamos por partes:
¿$I \wedge B_{c_1} \longrightarrow E_2$?
$esMax(i+1, escr, i+2) \equiv 0 \leq i+1 < i+2 \wedge_L (\forall k : \mathbb{Z})\ (0 \leq k < i+2 \longrightarrow_L escr[i+1] \geq escr[k]) \equiv$
$(\forall k : \mathbb{Z})\ (0 \leq k < i+1 \longrightarrow_L escr[i] \geq escr[k]) \wedge escr[i+1] \geq escr[i] \equiv$
$True \wedge escr[i+1] \geq escr[i] \equiv$ (ya que esMax(i,escr,i+1) es verdadero)
$escr[i+1] \geq escr[i]$

$esMax(i, escr, i+2) \equiv 0 \leq i < i+2 \wedge_L (\forall k : \mathbb{Z})\ (0 \leq k < i+2 \longrightarrow_L escr[i] \geq escr[k]) \equiv$
$(\forall k : \mathbb{Z})\ (0 \leq k < i+1 \longrightarrow_L escr[i] \geq escr[k]) \wedge escr[i] \geq escr[i+1] \equiv$
$True \wedge escr[i] \geq escr[i+1] \equiv$ (ya que esMax(i,escr,i+1) es verdadero)
$escr[i] \geq escr[i+1]$

Entonces E_2 queda:
$True \wedge ((1 \leq i+1 < |escr|-1 \wedge_L ((escr[i+1] > escr[i] \wedge escr[i+1] \geq escr[i]) \vee (escr[i+1] \leq escr[i] \wedge escr[i] \geq escr[i+1]))) \vee (i+1 = |escr|-1 \wedge True)) \equiv$
$(1 \leq i+1 < |escr|-1 \wedge_L (escr[i+1] > escr[i] \vee escr[i+1] \leq escr[i])) \vee (i+1 = |escr|-1) \equiv$
$(1 \leq i+1 < |escr|-1) \vee (i+1 = |escr|-1) \equiv$
$True$ (ya que en $I \wedge B_{c_1}$ tengo que $1 \leq i < |escr|-1$)

Ahora veo si $I \wedge B_{c_1} \longrightarrow E_1$
ya vimos que $esMax(i+1, escr, i+2) \equiv escr[i] \geq escr[i+1]$ si asumimos $I \wedge B_{c_1} verdadero$

$esMax(indiceMax1, escr, i+2) \equiv$
$0 \leq indiceMax1 < i+2 \wedge_L (\forall k : \mathbb{Z})\ (0 \leq k < i+2 \longrightarrow_L escr[indiceMax1] \geq escr[k]) \equiv$
$(\forall k : \mathbb{Z})\ (0 \leq k < i+1 \longrightarrow_L escr[indiceMax1] \geq escr[k]) \wedge escr[indiceMax1] \geq escr[i+1] \equiv$
$escr[indiceMax1] \geq escr[i+1]$

Entonces E_1 queda:
$True \wedge ((1 \leq i+1 < |escr|-1 \wedge_L ((escr[i+1] > escr[indiceMax1] \wedge escr[i] \geq escr[i+1]) \vee (escr[i+1] \leq escr[indiceMax1] \wedge escr[indiceMax1] \geq escr[i+1]))) \vee (i+1 = |escr|-1 \wedge True)) \equiv$
$(1 \leq i+1 < |escr|-1 \wedge_L ((escr[i+1] > escr[indiceMax1] \vee (escr[i+1] \leq escr[indiceMax1])) \vee (i+1 = |escr|-1) \equiv$
$(1 \leq i+1 < |escr|-1 \vee i+1 = |escr|-1) \equiv$

$True$

La expresión completa me queda: $0 \leq i < |escr| \wedge_L 0 \leq indiceMax1 < |escr| \wedge_L ((escr[i] > escr[indiceMax1] \wedge True) \vee (escr[i] \leq escr[indiceMax1] \wedge True)) \equiv$
$True \wedge_L True \wedge_L (escr[i] > escr[indiceMax1] \vee escr[i] \leq escr[indiceMax1]) \equiv$
$True$

2.5 $\{\mathbf{I} \wedge \mathbf{B} \wedge \mathbf{v_0} = |\mathbf{escr}|\textbf{-i-1}\}$ S $\{|\mathbf{escr}|\textbf{-i-1}<\mathbf{v_0}\}$
Veo si $\{\mathbf{I} \wedge \mathbf{B} \wedge \mathbf{v_0} = |\mathbf{escr}|\textbf{-i-1}\} \longrightarrow$ wp(if...then...else...fi; i:=i+1, $|\mathbf{escr}|\textbf{-i-1}<\mathbf{v_0}$):
$I \wedge B \wedge v_0 = |escrutinio| - i - 1 \equiv$
$P \wedge ((1 \leq i < |escr| - 1 \wedge_L ((escr[i] > escr[indiceMax1] \wedge esMax(i, escr, i+1)) \vee (escr[i] \leq escr[indiceMax1] \wedge esMax(indiceMax1, escr, i+1)))) \vee (i = |escr| - 1 \wedge esMax(indiceMax1, escr, i))) \wedge i < |escr| - 1 \wedge v_0 = (|escr| - 1) - i$

$wp(if..then..else..fi; i := i+1, |escr| - i - 1 < v_0) \equiv wp(if..then..else..fi, wp(i := i+1, |escr| - i - 1 < v_0))$

$wp(i := i+1, |escr| - i - 1 < v_0) \equiv$
$def(i+1) \wedge_L |escr| - (i+1) - 1 < v_0 \equiv$
$|escr| - i - 2 < v_0$

$wp(if..then..else..fi, |escr| - i - 2 < v_0) \equiv$
$def(escr[i]) \wedge_L def(escr[indiceMax1]) \wedge_L ((escr[i] > escr[indiceMax1] \wedge wp(S_1, |escr| - i - 2 < v_0)) \vee (escr[i] \leq escr[indiceMax1] \wedge wp(S_2, |escr| - i - 2 < v_0))) \equiv$
$0 \leq i < |escr| \wedge_L 0 \leq indiceMax1 < |escr| \wedge_L ((escr[i] > escr[indiceMax1] \wedge wp(S_1, |escr| - i - 2 < v_0)) \vee (escr[i] \leq escr[indiceMax1] \wedge wp(S_2, |escr| - i - 2 < v_0)))$

$wp(S_1, |escr| - i - 2 < v_0) \equiv$
$wp(indiceMax1 := i, |escr| - i - 2 < v_0) \equiv$
$def(i) \wedge_L |escr| - i - 2 < v_0 \equiv$
$|escr| - i - 2 < v_0$

$wp(skip, |escr| - i - 2 < v_0) \equiv |escr| - i - 2 < v_0$

¿$I \wedge B \wedge v_0 = |escr| - 1 - i \longrightarrow wp(if..then..else..fi; i := i+1, |escr| - 1 - i < v_0)$?
Veo que si asumo $I \wedge B \wedge v_0 = |escr| - 1 - i$ verdadero, entonces se cumple que $0 \leq i < |escr|$ y que $0 \leq indiceMax1 < |escr|$ (por los esMax). También $v_0 = |escr| - 1 - i$ Me queda
$True \wedge_L True \wedge_L ((escr[i] > escr[indiceMax1] \wedge |escr| - i - 2 < |escr| - i - 1) \vee (escr[i] \leq escr[indiceMax1] \wedge |escr| - i - 2 < |escr| - i - 1)) \equiv$
$(escr[i] > escr[indiceMax1] \wedge True) \vee (escr[i] \leq escr[indiceMax1] \wedge True \equiv$
$escr[i] > escr[indiceMax1] \vee escr[i] \leq escr[indiceMax1] \equiv$
$True$

Segundo ciclo:

```
j:=1
indiceMax2:=0
while (j < escrutinio.size()−1) do
    if (escrutinio[j] > escrutinio[indiceMax1] & escrutinio[j] < escrutinio[indiceMax2]) then
        indiceMax2:=j
    else
        skip
    endif
    j := j+1
endwhile
res:=(indiceMax1,indiceMax2)
```

$P_{c_2} \equiv Q_{c_1} \wedge j = 1 \wedge indiceMax2 = 0$

$Q_{c_2} \equiv Q_{c_1} \wedge j = |escr| - 1 \wedge_L esMax2(indiceMax2, escr, |escr| - 1)$

$B_{c_2} \equiv j < |escr| - 1$

$I \equiv Q_{c_1} \wedge ((1 \leq j < |escr| - 1 \wedge_L ((escr[j] > escr[indiceMax2] \wedge j \neq indiceMax1 \wedge esMax2(j, escr, j+1)) \vee (escr[j] \leq escr[indiceMax2] \vee j = indiceMax1 \wedge esMax2(indiceMax2, escr, j+1)))) \vee (j = |escr| - 1 \wedge esMax2(indiceMax2, escr, j)))$

$fv = (|escr| - 1) - j$

1. $Q_{c_1} \longrightarrow_L wp(j := 1; indiceMax2 := 0, P_{c_2})$:
$P_{c_1} \equiv Q_{c_1} \wedge j = 1 \wedge indiceMax2 = 0$
$Q_{c_1} \equiv P \wedge i = |escr| - 1 \wedge_L esMax(indiceMax1, escr, |escr| - 1)$
$wp(j := 1; indiceMax2 := 0, P_{c_2}) \equiv wp(j := 1, wp(indiceMax2 := 0, P_{c_2}))$

$wp(indiceMax2 := 0, P_{c_2}) \equiv$
$def(0) \wedge_L Q_{c_1} \wedge j = 1 \wedge 0 = 0 \equiv$
$Q_{c_1} \wedge j = 1$
$wp(i := 1, Q_{c_1} \wedge j = 1) \equiv$
$def(1) \wedge Q_{c_1} \wedge 1 = 1 \equiv$
Q_{c_1}

Me queda $Q_{c_1} \longrightarrow_L Q_{c_1}$ que es verdadero.

3. $Q_{c_2} \longrightarrow_L wp(res := (indiceMax1, indiceMax2), Q)$:
$Q_{c_2} \equiv Q_{c_1} \wedge j = |escr| - 1 \wedge_L esMax2(indiceMax2, escr, |escr| - 1) \equiv$
$P \wedge i = |escr| - 1 \wedge_L esMax(indiceMax1, escr, |escr| - 1) \wedge j = |escr| - 1 \wedge_L esMax2(indiceMax2, escr, |escr| - 1)$

$Q \equiv esMax(res_0, escr, |escr| - 1 \wedge esMax2(res_1, escr, |escr| - 1)$

$wp(res := (indiceMax1, indiceMax2), Q) \equiv$
$def(indiceMax1) \wedge_L def(indiceMax2) \wedge_L esMax(indiceMax1, escr, |escr| - 1 \wedge esMax2(indiceMax2, escr, |escr| - 1) \equiv$
$True \wedge_L True \wedge_L esMax(indiceMax1, escr, |escr| - 1 \wedge esMax2(indiceMax2, escr, |escr| - 1) \equiv$
$esMax(indiceMax1, escr, |escr| - 1 \wedge esMax2(indiceMax2, escr, |escr| - 1)$

$Q_{c_2} \longrightarrow_L wp(res := (indiceMax1, indiceMax2), Q)$ **es verdadero ya que si asumo que Q_{c_2} es verdadero, entonces $esMax(indiceMax1, escr, |escr| - 1 \wedge esMax2(indiceMax2, escr, |escr| - 1)$ también lo es.**