Trabajo Practico 1

Especificaciones, Implementaciones y Demostraciones

3 de noviembre de 2023 — Algoritmos y Estructuras de Datos

Grupo undefined

Integrante	LU	Correo electrónico
Integrante1	-	–
Integrante2	-	–
Integrante3	-	–
Integrante4	-	–

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

Preliminares

Algunos predicados preliminares que seran utilizados en muchos procesos:

`pred sinRepetidos` (lista: seq$\langle\mathbb{Z}\rangle$) {
$(\forall i, j : \mathbb{Z})(0 \leq i, j < |lista| \wedge i \neq j \rightarrow_{\text{L}} lista[i] \neq lista[j])$
}

`pred votosValidos` (votos: $seq\langle\mathbb{Z}\rangle$) {
$(\forall i : \mathbb{Z})(0 \leq i < |votos| \rightarrow votos[i] \geq 0)$
}

`pred matrizNoVacia` (matriz: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
$(|matriz| > 0) \wedge (\forall k : \mathbb{Z})(0 \leq k < |matriz| \rightarrow_{\text{L}} |matriz[k]| > 0)$
}

1. Ejercicio 1: hayBallotage

hayBallotage: verifica si hay ballotage en la elección presidencial.

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

Donde:

- escrutinio: es la cantidad de votos de cada partido a nivel nacional para la elección presidencial.
- devuelve verdadero sii hay ballotage en la elección presidencial.

1.1. Especificación

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

requiere $\{|escrutinio| \geq 3 \wedge\ sinRepetidos(escrutinio) \wedge\ votosValidos(escrutinio)\}$

asegura $\{res = False \iff ((\exists j : \mathbb{Z})(0 \leq j < |escrutinio| \longrightarrow_L escrutinio[j] > 0{,}45 * totalVotos)) \vee ((\forall i : \mathbb{Z})(\exists k : \mathbb{Z})(0 \leq i < |escrutinio| \wedge_L 0 \leq k < |escrutinio| \wedge i \neq k \longrightarrow_L (escrutinio[k] > 0{,}4 * totalVotos \wedge escrutinio[i] < escrutinio[k] - 0{,}1 * totalVotos))\}$

aux totalVotos(escrutinio: $seq\langle\mathbb{Z}\rangle$): $\mathbb{Z} = \sum_{k=0}^{|escrutinio|-1} escrutinio[k]$

1.2. Implementación

```
res := True;
i:= 0;
j:= 0;
mas_votos:= 0;
2do_mas_votos:= 0;
total_votos:= 0;
10%_votos:= 0;
40%_votos:= 0;
45%_votos:= 0;

while (j<escrutinio.size()) do
    total_votos:= escrutinio[j] + total_votos;
    j:= j + 1;

endwhile

10%_votos:= total_votos * 0.1;
40%_votos:= total_votos * 0.4;
45%_votos:= total_votos * 0.45;

while (i<escrutinio.size()) do
    if (escrutinio[i] > 45%_votos) then
        res:= False;
    else
        if (escrutinio[i] > mas_votos) then
            2do_mas_votos:= mas_votos;
            mas_votos:= escrutinio[i];
        else
            if ( escrutinio[i] < mas_votos && escrutinio[i]>2do_mas_votos) then
            2do_mas_votos:= escrutinio[i];

            else
                skip

            endif
        endif
    endif
    i:= i + 1;

endwhile

```

```
if (mas_votos > 40%_votos) then
    if (mas_votos − 2do_mas_votos > 10%_votos) then
        res:= False;
    else
        skip
    endif
else
    skip
endif
```

Código 1: Codigo de Ej1

2. Ejercicio 2: hayFraude

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

Donde:

- los tres escrutinios son a nivel nacional.
- devuelve verdadero sii hay al menos dos escrutinios con diferente cantidad de votos

2.1. Especificación

`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` $\{votosValidos(escrutinio\_presidencial) \land votosValidos(escrutinio\_senadores) \land votosValidos(escrutinio\_diputados)\}$

`requiere` $\{sinRepetidos(escrutinio\_presidencial) \land sinRepetidos(escrutinio\_senadores) \land sinRepetidos(escrutinio\_diputados)\}$

`asegura` $\{res = true \iff (\sum_{i=0}^{|escrutinio\_presidencial|-1} escrutinio\_presidencial[i] \neq \sum_{i=0}^{|escrutinio\_senadores|-1} escrutinio\_senadores[i]) \lor (\sum_{i=0}^{|escrutinio\_senadores|-1} escrutinio\_senadores[i] \neq \sum_{i=0}^{|escrutinio\_diputados|-1} escrutinio\_diputados[i])\}$

Observación: No compare el largo de las listas: escrutinio_presidencial, escrutinio_senadores y escrutinio_diputados, ya que, existe la posibilidad de que un partido presente lista presidencial, pero no para senadores. Por ejemplo, podría haber 10 partidos presentándose para presidente, pero solo 7 presentan diputados. En este caso, los escrutinios serian validos, pero el largo de las listas no seria el mismo.

2.2. Implementación

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

sumaES := 0;
b := 0;
while ( b < escrutinio_senadores.size() ) do
    sumaES := sumaES + escrutinio_senadores[b];
    b := b + 1;
endwhile

sumaED := 0;
c := 0;
while ( c < escrutinio_diputados.size() ) do
    sumaED := sumaED + escrutinio_diputados[c];
    c := c + 1;
endwhile

res := ( (sumaEP ≠ sumaES) ∨ (sumaES ≠ sumaED) )
```

Código 2: Codigo de Ej2

2.3. Demostración de Correctitud

Paso 1:
Elección de P_c, Q_c, B, I, fv para cada uno de los 3 ciclos. Como los 3 ciclos son similares, sus Invariantes también, por lo tanto trataremos el caso de un ciclo representativo y luego analizaremos el caso particular de cada uno. Llamemos a este programa S_C

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

Luego, nos queda que los componentes del Invariante serian:

- $P_c \equiv \text{suma} = 0 \wedge \text{i} = 0$
- $Q_c \equiv \text{suma} = \sum_{k=0}^{|S|-1} S[k]$
- $B \equiv i < |S|$
- $I \equiv (0 \leq i \leq |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k])$
- $fv \equiv |S| - i$

Analicemos si el Invariante cumple con las pruebas de correctitud.

1. $P_c \longrightarrow I$
$P_c \equiv \text{suma} = 0 \wedge \text{i} = 0$
$I \equiv 0 \leq i \leq |S| \wedge suma = \sum_{k=0}^{i-1} S[k]$
$P_c \longrightarrow I \equiv ((\text{suma} = 0) \wedge (\text{i} = 0)) \longrightarrow ((0 \leq i \leq |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k]))$
$P_c \longrightarrow I \equiv ((0 \leq 0 \leq |S|) \wedge (0 = \sum_{k=0}^{0-1} S[k]))$
$P_c \longrightarrow I \equiv ((True) \wedge (0 = \sum_{k=0}^{-1} S[k]))$
$P_c \longrightarrow I \equiv ((True) \wedge (0 = 0)$
$P_c \longrightarrow I \equiv ((True) \wedge (True)) \equiv True$

2. $I \wedge \neg B \longrightarrow Q_c$ $I \equiv 0 \leq i \leq |S| \wedge suma = \sum_{k=0}^{i-1} S[k]$
$B \equiv i < |S|$
$Q_c \equiv \text{suma} = \sum_{k=0}^{|S|-1} S[k]$
$I \wedge \neg B \longrightarrow Q_c \equiv (0 \leq i \leq |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k]) \wedge \neg(i < |S|) \longrightarrow \text{suma} = \sum_{k=0}^{|S|-1} S[k]$
$I \wedge \neg B \longrightarrow Q_c \equiv (0 \leq i \leq |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k]) \wedge (i \geq |S|) \longrightarrow \text{suma} = \sum_{k=0}^{|S|-1} S[k]$
$I \wedge \neg B \longrightarrow Q_c \equiv (0 \leq i \leq |S|) \wedge (i \geq |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k]) \longrightarrow \text{suma} = \sum_{k=0}^{|S|-1} S[k]$
$I \wedge \neg B \longrightarrow Q_c \equiv (i = |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k]) \longrightarrow \text{suma} = \sum_{k=0}^{|S|-1} S[k]$
$I \wedge \neg B \longrightarrow Q_c \equiv suma = \sum_{k=0}^{|S|-1} S[k] \longrightarrow \text{suma} = \sum_{k=0}^{|S|-1} S[k]$
$I \wedge \neg B \longrightarrow Q_c \equiv True$

3. $I \wedge fv \leq 0 \longrightarrow \neg B$
$I \wedge fv \leq 0 \equiv (0 \leq i \leq |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k]) \wedge |S| - i \leq 0 \equiv (i = |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k])$
$I \wedge fv \leq 0 \longrightarrow \neg B \equiv (i = |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k]) \longrightarrow (i \geq |S|)$
$I \wedge fv \leq 0 \longrightarrow \neg B \equiv (suma = \sum_{k=0}^{i-1} S[k]) \longrightarrow (|S| \geq |S|) \equiv True$

4. $\{I \wedge B\} S_C \{I\}$
Quiero ver que $I \wedge B \longrightarrow wp(S_C, I)$

4.a $wp(S_C, I)$
$wp(S_C, I) \equiv wp(suma := suma + S[i]; i := i + 1; (0 \leq i \leq |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k]))$
$wp(S_C, I) \equiv wp(suma := suma + S[i], wp(i := i + 1, (0 \leq i \leq |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k])))$
$wp(S_C, I) \equiv wp(suma := suma + S[i], wp(i := i + 1, (0 \leq i \leq |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k])))$

Veamos $wp(i := i + 1, (0 \leq i \leq |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k]))$
$\equiv def(i + 1) \wedge (0 \leq i + 1 \leq |S|) \wedge (suma = \sum_{k=0}^{i+1-1} S[k])$
$\equiv (0 \leq i \leq |S| - 1) \wedge (suma = \sum_{k=0}^{i} S[k])$
$\equiv (0 \leq i < |S|) \wedge (suma = \sum_{k=0}^{i} S[k]) \equiv E_1$

Ahora veamos $wp(suma := suma + S[i], E_1)$
$\equiv def(suma + S[i]) \wedge (0 \leq i < |S|) \wedge (suma + S[i] = \sum_{k=0}^{i} S[k])$
$\equiv (0 \leq i < |S|) \wedge (0 \leq i < |S|) \wedge (suma + S[i] = \sum_{k=0}^{i-1} S[k] + S[i])$
$\equiv (0 \leq i < |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k])$
Entonces $wp(S, I) \equiv (0 \leq i < |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k])$.

4.b
$I \wedge B \longrightarrow wp(S_C, I)$
$\equiv (0 \leq i \leq |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k]) \wedge (i < |S|) \longrightarrow (0 \leq i < |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k])$
$\equiv (0 \leq i < |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k]) \longrightarrow (0 \leq i < |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k]) \equiv True$

5.$\{I \wedge B \wedge v_0 = |S| - i\} S_C \{|S| - i < v_0\}$ Veamos la $wp(S_C, |S| - i < v_0)$
5.a $wp(S_C, |S| - i < v_0)$
$wp(S_C, I) \equiv wp(suma := suma + S[i]; i := i + 1; |S| - i < v_0)$
$wp(S_C, I) \equiv wp(suma := suma + S[i], wp(i := i + 1, |S| - i < v_0))$
$wp(S_C, I) \equiv wp(suma := suma + S[i], wp(i := i + 1, |S| - i < v_0))$

Veamos $wp(i := i + 1, |S| - i < v_0)$
$\equiv def(i + 1) \wedge |S| - i - 1 < v_0$
$\equiv |S| - i < v_0 + 1$
Ahora veamos $wp(suma := suma + S[i], |S| - i < v_0 + 1)$
$\equiv def(suma + S[i]) \wedge |S| - i < v_0 + 1$
$\equiv (0 \leq i \leq |S|) \wedge (|S| - i < v_0 + 1)$

Entonces $wp(S_C, |S| - i < v_0) \equiv (0 \leq i < |S|) \wedge (|S| - i < v_0 + 1)$

Veamos ahora que $(I \wedge B \wedge v_0 = |S| - i) \longrightarrow (0 \leq i < |S|) \wedge (|S| - i < v_0 + 1)$
$(I \wedge B \wedge v_0 = |S| - i) \equiv (0 \leq i < |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k]) \wedge v_0 = |S| - i$
$(0 \leq i < |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k]) \wedge (v_0 = |S| - i) \longrightarrow (0 \leq i < |S|) \wedge (|S| - i < v_0 + 1)$
$\equiv (0 \leq i < |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k]) \longrightarrow (0 \leq i < |S|) \wedge (|S| - i < |S| - i + 1)$
$\equiv (0 \leq i < |S|) \wedge (suma = \sum_{k=0}^{i-1} S[k]) \longrightarrow (0 \leq i < |S|) \wedge (|S| < |S| + 1) \equiv True$

Habiendo demostrado cada uno de los pasos, queda demostrado que la tripla de Hoare:

$$\{True\} S_C \{suma = \sum_{k=0}^{|S|-1} S[k]\}. \tag{1}$$

Teniendo esto en mente si vemos el código escrito (Codigo 2) en SmallLang, podemos ver que si remplazamos i por a, b y c y $suma$ por $SumaEP, SumaES$ y $SumaED$ en cada uno de los tres ciclos, nos queda que las tres triplas de Hoare son validad, por tanto nos falta demostrar que la tripla:
$\{(sumaEP = \sum_{i=0}^{|escrutinio\_presidente|-1} escrutinio\_presidente[i]) \wedge$
$(sumaES = \sum_{i=0}^{|escrutinio\_senadores|-1} escrutinio\_senadores[i]) \wedge$
$(sumaED = \sum_{i=0}^{|escrutinio\_diputados|-1} escrutinio\_diputados[i]) \equiv Pre\}$
res := ((sumaEP $\neq$ sumaES) $\vee$ (sumaES $\neq$ sumaED))
$\{res = true \iff (\sum_{i=0}^{|escrutinio\_presidencial|-1} escrutinio\_presidencial[i] \neq \sum_{i=0}^{|escrutinio\_senadores|-1} escrutinio\_senadores[i])$
$\vee(\sum_{i=0}^{|escrutinio\_senadores|-1} escrutinio\_senadores[i] \neq \sum_{i=0}^{|escrutinio\_diputados|-1} escrutinio\_diputados[i]) \equiv Post\}$

veamos el $wp(res := ((sumaEP \neq sumaES) \vee (sumaES \neq sumaED)), Post)$
$\equiv def(sumaEP) \wedge def(sumaES) \wedge def(sumaED) \wedge ((sumaEP \neq sumaES) \vee (sumaES \neq sumaED)) = true \iff$
$(\sum_{i=0}^{|escrutinio\_presidencial|-1} escrutinio\_presidencial[i] \neq \sum_{i=0}^{|escrutinio\_senadores|-1} escrutinio\_senadores[i])$
$\vee(\sum_{i=0}^{|escrutinio\_senadores|-1} escrutinio\_senadores[i] \neq \sum_{i=0}^{|escrutinio\_diputados|-1} escrutinio\_diputados[i]$
$\equiv ((sumaEP \neq sumaES) \vee (sumaES \neq sumaED)) = true \iff$
$(\sum_{i=0}^{|escrutinio\_presidencial|-1} escrutinio\_presidencial[i] \neq \sum_{i=0}^{|escrutinio\_senadores|-1} escrutinio\_senadores[i])$
$\vee(\sum_{i=0}^{|escrutinio\_senadores|-1} escrutinio\_senadores[i] \neq \sum_{i=0}^{|escrutinio\_diputados|-1} escrutinio\_diputados[i]$

Podemos ver que esta expresión es tautológica en el caso en que:

- $sumaEP = \sum_{i=0}^{|escrutinio\_presidente|-1} escrutinio\_presidente[i]$
- $sumaES = \sum_{i=0}^{|escrutinio\_senadores|-1} escrutinio\_senadores[i]$
- $sumaED = \sum_{i=0}^{|escrutinio\_diputados|-1} escrutinio\_diputados[i]$

Por tanto llegamos a que: $wp(res := ((sumaEP \neq sumaES) \vee (sumaES \neq sumaED)), Post) \equiv$
$\{(sumaEP = \sum_{i=0}^{|escrutinio\_presidente|-1} escrutinio\_presidente[i]) \wedge$
$(sumaES = \sum_{i=0}^{|escrutinio\_senadores|-1} escrutinio\_senadores[i]) \wedge$
$(sumaED = \sum_{i=0}^{|escrutinio\_diputados|-1} escrutinio\_diputados[i])\}$
Que es exactamente nuestra precondion, por lo tanto queda demostrado que el código (Codigo 2) es correcto.

3. Ejercicio 3: obtenerSenadoresEnProvincia

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⟨Z⟩) : Z × Z

Donde:

- escrutinio: es la cantidad de votos de cada partido en la provincia.
- devuelve una tupla que contiene el id de los dos partidos con mayor cantidad de votos.

3.1. Especificación

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

`requiere` $\{|escrutinio| \geq 2 \land votosValidos(escrutinio) \land sinRepetidos(escrutinio)\}$

`asegura` $\{(res = (res_0, res_1) \leftrightarrow (\exists res_0 : \mathbb{Z})(\exists res_1 : \mathbb{Z})(\forall x : \mathbb{Z})(x \neq res_0 \land x \neq res_1) \land_L (0 \leq res_0, res_1, x \leq |escrutinio|) \rightarrow_L (escrutinio[x] < escrutinio[res_1]) \land (escrutinio[res_1] < escrutinio[res_0]))\}$

3.2. Implementacion

```
pos := 0;
res_0 := 0;
res_1 := 1;

if (escrutinio [res_0] < escrutinio [res_1]) do
    res_0 := 1;
    res_1 := 0;
else
    skip
endif

while pos < escrutinio.size() do
    if escrutinio [pos] > escrutinio [res_0] then
        res_0 := posicion;
    else
        skip;
    endif
    pos := pos + 1;
endwhile

pos := 0;

while pos < escrutinio.size() do
    if escrutinio [pos] > escrutinio [res_1] && escrutinio [res_1] != escrutinio [res_0] then
        res_1 := posicion;
    else
        skip;
    endif
    pos := pos + 1;
endwhile

res := (res_0, res_1)
```

Código 3: Codigo de Ej3

3.3. Demostración de Correctitud

Elección de P_c, Q_c, B, I, fv para ambos ciclos. Como los ambos ciclos son similares, sus Invariantes también, por lo tanto trataremos el caso de un ciclo representativo y luego analizaremos el caso particular de cada uno. Llamemos a este programa S_C

```
pos := 0

while pos < escrutinio.size() do
    if escrutinio [pos] > escrutinio [res] then
        res := pos;
    else
        skip;
    endif

    pos := pos + 1;
endwhile
```

Los componentes del Invariante, para este caso, serian:

- $P_c \equiv (|escrutinio| \geq 2) \wedge (res = 0 \vee res = 1) \wedge (pos = 0)$
- $Q_c \equiv (|escrutinio| \geq 2) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < |escrutinio|) \rightarrow_L (escrutinio[X] \leq escrutinio[res])$
- $B \equiv pos < |escrutinio|$
- $I \equiv (|escrutinio| \geq 2) \wedge (0 \leq pos \leq |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < pos) \rightarrow_L (escrutinio[X] \leq escrutinio[res])$
- $fv \equiv |escrutinio| - pos$

Analicemos si el Invariante cumple con las pruebas de correctitud.

1. $P_c \longrightarrow I$

$P_c \equiv (|escrutinio| \geq 2) \wedge (res = 0 \vee res = 1) \wedge (pos = 0)$
$I \equiv (|escrutinio| \geq 2) \wedge (0 \leq pos \leq |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < pos) \rightarrow_L (escrutinio[X] \leq escrutinio[res])$

$P_c \longrightarrow I \equiv (|escrutinio| \geq 2) \wedge (res = 0 \vee res = 1) \wedge (pos = 0) \rightarrow (|escrutinio| \geq 2) \wedge (0 \leq pos \leq |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < pos) \rightarrow_L (escrutinio[X] \leq escrutinio[res])$
$P_c \longrightarrow I \equiv (|escrutinio| \geq 2) \wedge (pos = 0) \rightarrow (|escrutinio| \geq 2) \wedge (0 \leq pos \leq |escrutinio|) \wedge (0 \leq 0 < |escrutinio| \vee 0 \leq 1 < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < pos) \rightarrow_L (escrutinio[X] \leq escrutinio[res])$
$P_c \longrightarrow I \equiv (|escrutinio| \geq 2) \rightarrow (|escrutinio| \geq 2) \wedge (0 \leq 0 \leq |escrutinio|) \wedge (0 \leq 0 < |escrutinio| \vee 0 \leq 1 < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < 0) \rightarrow_L (escrutinio[X] \leq escrutinio[res])$
$P_c \longrightarrow I \equiv (\mathbf{True}) \wedge (\mathbf{True}) \wedge (\mathbf{True} \vee \mathbf{True}) \wedge (\forall X : \mathbb{Z})(\mathbf{False}) \rightarrow_L (escrutinio[X] \leq escrutinio[res])$
$P_c \longrightarrow I \equiv (\mathbf{True}) \wedge (\mathbf{True}) \wedge (\mathbf{True}) \wedge (\mathbf{True}) = \mathbf{True}$

2. $I \wedge \neg B \longrightarrow Q_c$

$I \equiv (|escrutinio| \geq 2) \wedge (0 \leq pos \leq |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < pos) \rightarrow_L (escrutinio[X] \leq escrutinio[res])$
$B \equiv pos < |escrutinio|$
$Q_c \equiv (|escrutinio| \geq 2) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < |escrutinio|) \rightarrow_L (escrutinio[X] < escrutinio[res])$

$I \wedge \neg B \longrightarrow Q_c \equiv (((|escrutinio| \geq 2) \wedge (0 \leq pos \leq |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < pos) \rightarrow_L (escrutinio[X] \leq escrutinio[res])) \wedge (pos \geq |escrutinio|)) \longrightarrow Q_c$
$I \wedge \neg B \longrightarrow Q_c \equiv def(pos = |escrutinio|) \wedge_L (((|escrutinio| \geq 2) \wedge (0 \leq |escrutinio| \leq |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < |escrutinio|) \rightarrow_L (escrutinio[X] \leq escrutinio[res])) \wedge (|escrutinio| \geq |escrutinio|)) \longrightarrow Q_c$
$I \wedge \neg B \longrightarrow Q_c \equiv (((|escrutinio| \geq 2) \wedge (\mathbf{True}) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < |escrutinio|) \rightarrow_L (escrutinio[X] \leq escrutinio[res])) \wedge (\mathbf{True})) \longrightarrow Q_c$
$I \wedge \neg B \longrightarrow Q_c \equiv ((0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < |escrutinio|) \rightarrow_L (escrutinio[X] \leq escrutinio[res])) \rightarrow Q_c$
$I \wedge \neg B \longrightarrow Q_c \equiv Q_c \longrightarrow Q_c = \mathbf{True}$

3. $I \wedge fv \leq 0 \longrightarrow \neg B$

$I \equiv (|escrutinio| \geq 2) \wedge (0 \leq pos \leq |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < pos) \rightarrow_L (escrutinio[X] \leq escrutinio[res])$
$fv \equiv |escrutinio| - pos$
$B \equiv pos < |escrutinio|$

$(I \wedge fv \leq 0) \longrightarrow \neg B \equiv (I \wedge (|escrutinio| - pos) \leq 0) \longrightarrow \neg B)$
$(I \wedge fv \leq 0) \longrightarrow \neg B \equiv (I \wedge |escrutinio| \leq pos) \longrightarrow \neg B)$
$(I \wedge fv \leq 0) \longrightarrow \neg B \equiv (I \wedge \neg B) \longrightarrow \neg B) \equiv \textbf{True}$

4. $\{I \wedge B\} S_C \{I\}$

$I \equiv (|escrutinio| \geq 2) \wedge (0 \leq pos \leq |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < pos) \rightarrow_L (escrutinio[X] \leq escrutinio[res])$
$B \equiv pos < |escrutinio|$

Quiero ver que $I \wedge B \longrightarrow wp(S_C, I)$

4.a $I \wedge B$

$I \wedge B$
$\equiv (((|escrutinio| \geq 2) \wedge (0 \leq pos \leq |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < pos) \rightarrow_L (escrutinio[X] \leq escrutinio[res])) \wedge (pos < |escrutinio|))$
$\equiv def(pos < |escrutinio|) \wedge_L (((|escrutinio| \geq 2) \wedge (0 \leq pos < |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < |escrutinio|) \rightarrow_L (escrutinio[X] \leq escrutinio[res])) \wedge (\textbf{True}))$
$\equiv (|escrutinio| \geq 2) \wedge (0 \leq pos < |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < |escrutinio|) \rightarrow_L (escrutinio[X] \leq escrutinio[res])$

4.b $wp(S_C, I)$

$wp(S_C, I) \equiv wp(if(escrutinio[pos] > escrutinio[res])then(res = pos)else(skip)endif; (pos = pos + 1), I)$
$wp(S_C, I) \equiv wp(if(escrutinio[pos] > escrutinio[res])then(res = pos)else(skip)endif, wp(pos = pos + 1, I))$

En un principio analizo:

$wp(pos = pos + 1, I) \equiv (pos = pos + 1) \wedge (|escrutinio| \geq 2) \wedge (0 \leq pos \leq |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < pos) \rightarrow_L (escrutinio[X] \leq escrutinio[res])$
$wp(pos = pos + 1, I) \equiv def(pos = pos + 1)\{(|escrutinio| \geq 2) \wedge (0 \leq pos + 1 \leq |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < pos + 1) \rightarrow_L (escrutinio[X] \leq escrutinio[res])\}$
$wp(pos = pos + 1, I) \equiv \{(|escrutinio| \geq 2) \wedge (0 \leq pos + 1 \leq |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < pos + 1) \rightarrow_L (escrutinio[X] \leq escrutinio[res])\}$
$wp(pos = pos + 1, I) \equiv E1$

Al tratarse de un IF, se divide la implicancia para ambos casos

$((I \wedge B) \wedge (escrutinio[pos] > escrutinio[res])) \longrightarrow wp(res := pos, E1)$
$((I \wedge B) \wedge (escrutinio[pos] > escrutinio[res])) \longrightarrow def(res) \wedge_L (|escrutinio| \geq 2) \wedge (0 \leq pos + 1 \leq |escrutinio|) \wedge (0 \leq pos < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < pos + 1) \rightarrow_L (escrutinio[X] \leq escrutinio[pos])$
$((I \wedge B) \wedge (escrutinio[pos] > escrutinio[res])) \longrightarrow def(res = pos) \wedge_L (|escrutinio| \geq 2) \wedge (0 \leq pos{+}1 \leq |escrutinio|) \wedge (0 \leq pos < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < pos + 1) \rightarrow_L (escrutinio[X] \leq escrutinio[pos])$
$((I \wedge B) \wedge (escrutinio[pos] > escrutinio[res])) \longrightarrow def(res = pos) \wedge_L (|escrutinio| \geq 2) \wedge (0 \leq pos < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < pos + 1) \rightarrow_L (escrutinio[X] \leq escrutinio[pos])$
$(((|escrutinio| \geq 2) \wedge (0 \leq pos < |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < |escrutinio|) \rightarrow_L (escrutinio[X] \leq escrutinio[res])) \wedge (escrutinio[pos] > escrutinio[res])) \longrightarrow def(res = pos) \wedge_L (|escrutinio| \geq 2) \wedge (0 \leq pos < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < pos + 1) \rightarrow_L (escrutinio[X] \leq escrutinio[pos]) \equiv \textbf{True}$

$((I \wedge B) \wedge (escrutinio[pos] \leq escrutinio[res])) \longrightarrow wp(skip, E1)$
$((I \wedge B) \wedge (escrutinio[pos] \leq escrutinio[res])) \longrightarrow (|escrutinio| \geq 2) \wedge (0 \leq pos + 1 \leq |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < pos + 1) \rightarrow_L (escrutinio[X] \leq escrutinio[res])$

$((I \wedge B) \wedge (escrutinio[pos] \leq escrutinio[res])) \longrightarrow (|escrutinio| \geq 2) \wedge (0 \leq pos + 1 \leq |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < |escrutinio|) \rightarrow_L (escrutinio[X] \leq escrutinio[res])$

$(((|escrutinio| \geq 2) \wedge (0 \leq pos < |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < |escrutinio|) \rightarrow_L (|escrutinio| \geq 2) \wedge (escrutinio[X] \leq escrutinio[res])) \wedge (escrutinio[pos] \leq escrutinio[res])) \longrightarrow (0 \leq pos+1 \leq |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < |escrutinio|) \rightarrow_L (escrutinio[X] \leq escrutinio[res]) \equiv \textbf{True}$

5. $\{I \wedge B \wedge vo = fv\}S_C\{fv < vo\}$

$I \equiv (|escrutinio| \geq 2) \wedge (0 \leq pos \leq |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < pos) \rightarrow_L (escrutinio[X] \leq escrutinio[res])$
$B \equiv pos < |escrutinio|$
$fv \equiv |escrutinio| - pos$

$\{I \wedge B \wedge vo = |escrutinio| - pos\}S_C\{|escrutinio| - pos < vo\}$

Quiero ver que $(I \wedge B \wedge vo = fv) \rightarrow wp(Sc, (fv < vo))$

5.a $I \wedge B \wedge vo = fv$

$I \wedge B \wedge vo = fv \equiv ((|escrutinio| \geq 2) \wedge (0 \leq pos < |escrutinio|) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < |escrutinio|) \rightarrow_L (escrutinio[X] \leq escrutinio[res])) \wedge (vo = |escrutinio| - pos)$

5.b $wp(S_C, fv < vo)$

$wp(S_C, fv < vo) \equiv wp(if(escrutinio[pos] > escrutinio[res])then(res = pos)else(skip)endif, wp(pos = pos+1, fv < vo))$

En un pricipio analizo:

$wp(pos = pos + 1, fv < vo) \equiv (pos = pos + 1) \wedge (|escrutinio| - pos < vo)$
$wp(pos = pos + 1, fv < vo) \equiv def(pos = pos + 1) \wedge_L (\textbf{True}) \wedge (|escrutinio| - pos - 1 < vo)$
$wp(pos = pos + 1, fv < vo) \equiv (|escrutinio| - pos - 1 < vo) \equiv E2$

Al tratarse de un IF, se divide la implicancia para ambos casos

$(I \wedge B \wedge vo = fv \wedge (escrutinio[pos] > escrutinio[res])) \rightarrow wp(res := pos, E2)$
$(I \wedge B \wedge vo = fv \wedge (escrutinio[pos] > escrutinio[res])) \rightarrow def(res := pos) \wedge_L (|escrutinio| - res - 1 < vo)$
$(I \wedge B \wedge vo = (|escrutinio| - pos) \wedge (escrutinio[pos] > escrutinio[res])) \rightarrow def(res := pos) \wedge_L (|escrutinio| - res - 1 < vo)$
$(I \wedge B \wedge vo = (|escrutinio| - pos) \wedge (escrutinio[pos] > escrutinio[res])) \rightarrow def(vo := |escrutinio| - pos) \wedge_L def(res := pos) \wedge_L (|escrutinio| - res - 1 < |escrutinio| - pos) \equiv \textbf{True}$

$(I \wedge B \wedge vo = fv \wedge (escrutinio[pos] \leq escrutinio[res])) \rightarrow wp(res := pos, E2)$
$(I \wedge B \wedge vo = fv \wedge (escrutinio[pos] \leq escrutinio[res])) \rightarrow def(res := pos) \wedge_L (|escrutinio| - res - 1 < vo)$
$(I \wedge B \wedge vo = (|escrutinio| - pos) \wedge (escrutinio[pos] \leq escrutinio[res])) \rightarrow def(res := pos) \wedge_L (|escrutinio| - res - 1 < vo)$
$(I \wedge B \wedge vo = (|escrutinio| - pos) \wedge (escrutinio[pos] \leq escrutinio[res])) \rightarrow def(vo := |escrutinio| - pos) \wedge_L def(res := pos) \wedge_L (|escrutinio| - res - 1 < |escrutinio| - pos) \equiv \textbf{True}$

Habiendo demostrado cada uno de los pasos, queda demostrado que la tripla de Hoare:

$$\{True\} \tag{2}$$

$$S_C \tag{3}$$

$$\{(|escrutinio| \geq 2) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < |escrutinio|) \rightarrow_L (escrutinio[X] \leq escrutinio[res])\}. \tag{4}$$

Teniendo esto en mente, si vemos el código escrito (Codigo 3) en SmallLang, podemos ver que si remplazamos res por res_0 y res_1 para cada uno de los ciclos, nos queda que las dos triplas de Hoare son validad, siempre y cuando analicemos el segundo ciclo como si en el escrutinio ignorara el valor de res_0, dado por la segunda condición en su IF, agregando a su postcondicion el hecho de que $(res_1 \neq res_0) \wedge (escrutinio[res_1] < escrutinio[res_0])$. Por tanto queda demostrar la tripla: $\{Pre\}S\{Post\}$

Para ello primero es necesario demostrar la validez de:

```
pos := 0;
res_0 := 0;
res_1 := 1;

if (escrutinio [res_0] < escrutinio [res_1]) do
    res_0 := 1;
    res_1 := 0;
else
    skip
endif
```

Código 4: Codigo de Ej3

Quedando una tripla de Hoare del tipo:

$\{Pre_c\}S_c\{Post_c\}$

$Pre_c \equiv \{(|escrutinio| \geq 2) \wedge (pos = 0) \wedge (res_0 = 0) \wedge (res_1 = 1)\}$
$S_c \equiv if(escrutinio[res_0] < escrutinio[res_1])then(res_0 := 1; res_1 := 0)else(skip)endif$
$Post_c \equiv \{(|escrutinio| \geq 2) \wedge (pos = 0) \wedge ((res_0 = 0 \wedge res_1 = 1) \vee (res_0 = 1 \wedge res_1 = 0)) \wedge (escrutinio[res_0] > escrutinio[res_1])\}$

Para probar su correctitud se debe demostrar que:

$(Pre_c \wedge (escrutinio[res_0] < escrutinio[res_1])) \rightarrow wp(res_0 := 1; res_1 := 0, Post_c)$

$(Pre_c \wedge (escrutinio[res_0] \geq escrutinio[res_1])) \rightarrow wp(skip, Post_c)$

Primero analizo cada wp individualmente:

$wp(res_0 := 1; res_1 := 0, Post_c)$
$\equiv wp(res_0 := 1, wp(res_1 := 0, Post_c)))$
$\equiv wp(res_0 := 1, def(res_1 := 0) \wedge_L (((res_0 = 0 \wedge 0 = 1) \vee (res_0 = 1 \wedge 0 = 0)) \wedge (escrutinio[res_0] > escrutinio[0])))$
$\equiv def(res_0 := 1) \wedge_L ((\textbf{False}) \vee (\textbf{True})) \wedge (escrutinio[1] > escrutinio[0]))$
$\equiv (\textbf{True}) \wedge (escrutinio[1] > escrutinio[0]))$

$wp(skip, Post) \equiv def(skip) \wedge_L ((res_0 = 0 \wedge res_1 = 1) \vee (res_0 = 1 \wedge res_1 = 0)) \wedge (escrutinio[res_0] > escrutinio[res_1])$

Volviendo a las implicancias:

$(Pre_c \wedge (escrutinio[res_0] < escrutinio[res_1])) \rightarrow wp(res_0 := 1; res_1 := 0, Post_c)$
$((|escrutinio| \geq 2) \wedge (pos = 0) \wedge (res_0 = 0) \wedge (res_1 = 1) \wedge (escrutinio[res_0] < escrutinio[res_1])) \rightarrow (|escrutinio| \geq 2) \wedge (pos = 0) \wedge ((res_0 = 0 \wedge 0 = 1) \vee (res_0 = 1 \wedge 0 = 0)) \wedge (escrutinio[1] > escrutinio[0]) \equiv \textbf{True}$

$(Pre_c \wedge (escrutinio[res_0] \geq escrutinio[res_1])) \rightarrow wp(skip, Post_c)$
$((|escrutinio| \geq 2) \wedge (pos = 0) \wedge (res_0 = 0) \wedge (res_1 = 1) \wedge (escrutinio[res_0] \geq escrutinio[res_1])) \rightarrow (|escrutinio| \geq 2) \wedge (pos = 0) \wedge ((res_0 = 0 \wedge 0 = 1) \vee (res_0 = 1 \wedge 0 = 0)) \wedge (escrutinio[res_0] > escrutinio[res_1]) \equiv \textbf{True}$

Con esto queda demostrado la validez de la tripla de Horade:

$$\{(|escrutinio| \geq 2) \wedge (pos = 0) \wedge (res_0 = 0) \wedge (res_1 = 1)\} \tag{5}$$
$$S_c \tag{6}$$
$$\{(|escrutinio| \geq 2) \wedge (pos = 0) \wedge ((res_0 = 0 \wedge res_1 = 1) \vee (res_0 = 1 \wedge res_1 = 0)) \wedge (escrutinio[res_0] > escrutinio[res_1])\}. \tag{7}$$

Con la validez del condicional demostrada, se puede comenzar a analizar la validez del programa completo, con la tripla de Horade de $\{Pre\}S\{Post\}$

$Pre \equiv \{|escrutinio| \geq 2\}$
$Post \equiv \{(res = (res_0, res_1) \leftrightarrow (\exists res_0 : \mathbb{Z})(\exists res_1 : \mathbb{Z})(\forall x : \mathbb{Z})(x \neq res_0 \wedge x \neq res_1)(0 \leq res_0, res_1, x \leq |escrutinio|) \rightarrow_L (escrutinio[x] < escrutinio[res_1]) \wedge (escrutinio[res_1] < escrutinio[res_0])\}$

Para demostrar $\{Pre\}S\{Post\}$ es necesario que $Pre \rightarrow wp(S, Post)$, y para demostrar ello se deben cumplir:

$Pre \rightarrow wp(pos := 0; res_0 := 0; res_1 := 1, Pre_{Cond})$
$Pre_{Cond} \rightarrow wp(S_{Cond}, Post_{Cond})$
$Post_{Cond} \rightarrow wp(skip, Pre_{Ci1})$
$Pre_{Ci1} \rightarrow wp(S_{Ci1}, Post_{Ci1})$
$Post_{Ci1} \rightarrow wp(pos := 0, Pre_{Ci2})$
$Pre_{Ci2} \rightarrow wp(S_{Ci2}, Post_{Ci2})$
$Post_{Ci2} \rightarrow wp(res := (res_0, res_1), Post)$

Por las demostraciones de correctitud antes dadas, solo quedaria demostrar que:

$Pre \rightarrow wp(pos := 0; res_0 := 0; res_1 := 1, Pre_{Cond})$
$Post_{Cond} \rightarrow wp(skip, Pre_{Ci1})$
$Post_{Ci1} \rightarrow wp(pos := 0, Pre_{Ci2})$
$Post_{Ci2} \rightarrow wp(res := (res_0, res_1), Post)$

Entonces:

$Pre \rightarrow wp(pos := 0; res_0 := 0; res_1 := 1, Pre_{Cond})$
$Pre \rightarrow wp(pos := 0, wp(res_0 := 0, wp(res_1 := 1, Pre_{Cond})))$
$Pre \rightarrow wp(pos := 0, wp(res_0 := 0, def(res_1 := 1) \wedge_L Pre_{Cond}))$
$Pre \rightarrow wp(pos := 0, wp(res_0 := 0, (|escrutinio| \geq 2) \wedge (pos = 0) \wedge (res_0 = 0) \wedge (1 = 1)))$
$Pre \rightarrow wp(pos := 0, def(res_0 := 0) \wedge_L (|escrutinio| \geq 2) \wedge (pos = 0) \wedge (0 = 0) \wedge (\textbf{True}))$
$Pre \rightarrow def(pos := 0) \wedge_L (|escrutinio| \geq 2) \wedge (0 = 0) \wedge (\textbf{True})$
$(|escrutinio| \geq 2) \rightarrow (|escrutinio| \geq 2) \wedge (\textbf{True}) \equiv \textbf{True}$

$Post_{Cond} \rightarrow wp(skip, Pre_{Ci1})$
$Post_{Cond} \rightarrow Pre_{Ci1}$
$Post_{Cond} \rightarrow (|escrutinio| \geq 2) \wedge (res_0 = 0 \vee res_0 = 1) \wedge (pos = 0)$
$((|escrutinio| \geq 2) \wedge (pos = 0) \wedge ((res_0 = 0 \wedge res_1 = 1) \vee (res_0 = 1 \wedge res_1 = 0)) \wedge (escrutinio[res_0] > escrutinio[res_1])) \rightarrow (|escrutinio| \geq 2) \wedge (res_0 = 0 \vee res_0 = 1) \wedge (pos = 0) \equiv \textbf{True}$

$Post_{Ci1} \rightarrow wp(pos := 0, Pre_{Ci2})$
$Post_{Ci1} \rightarrow def(pos := 0) \wedge_L (|escrutinio| \geq 2) \wedge (res_1 = 0 \vee res_1 = 1) \wedge (0 = 0)$
$((|escrutinio| \geq 2) \wedge (0 \leq res < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < |escrutinio|) \rightarrow_L (escrutinio[X] \leq escrutinio[res])) \rightarrow (|escrutinio| \geq 2) \wedge (res_1 = 0 \vee res_1 = 1) \wedge (\textbf{True}) \equiv \textbf{True}$

$Post_{Ci2} \rightarrow wp(res := (res_0, res_1), Post)$
$Post_{Ci2} \rightarrow def(res := (res_0, res_1)) \wedge_L ((res_0, res_1) = (res_0, res_1) \leftrightarrow (\exists res_0 : \mathbb{Z})(\exists res_1 : \mathbb{Z})(\forall x : \mathbb{Z})(x \neq res_0 \wedge x \neq res_1)(0 \leq res_0, res_1, x \leq |escrutinio|) \rightarrow_L (escrutinio[x] < escrutinio[res_1]) \wedge (escrutinio[res_1] < escrutinio[res_0])$
$Post_{Ci2} \rightarrow (\textbf{True} \leftrightarrow (\exists res_0 : \mathbb{Z})(\exists res_1 : \mathbb{Z})(\forall x : \mathbb{Z})(x \neq res_0 \wedge x \neq res_1)(0 \leq res_0, res_1, x \leq |escrutinio|) \rightarrow_L (escrutinio[x] < escrutinio[res_1]) \wedge (escrutinio[res_1] < escrutinio[res_0])$
$(((|escrutinio| \geq 2) \wedge (0 \leq res_1 < |escrutinio|) \wedge (\forall X : \mathbb{Z})(0 \leq X < |escrutinio| \wedge X \neq res_0) \rightarrow_L (escrutinio[X] \leq escrutinio[res_1])) \wedge (res_1 \neq res_0) \wedge (escrutinio[res_1] < escrutinio[res_0])) \rightarrow (\textbf{True} \leftrightarrow (\exists res_0 : \mathbb{Z})(\exists res_1 : \mathbb{Z})(\forall x : \mathbb{Z})(x \neq res_0 \wedge x \neq res_1)(0 \leq res_0, res_1, x \leq |escrutinio|) \rightarrow_L (escrutinio[x] < escrutinio[res_1]) \wedge (escrutinio[res_1] < escrutinio[res_0]) \equiv \textbf{True}$

Con estas ultimas demostraciones, queda demostrada la validez del codigo y el cumplimineto de la tripla de Hoarde: $\{Pre\}S\{Post\}$

4. Ejercicio 4: calcularDHondtEnProvincia

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 cant_bancas: $\mathbb{Z}$, in escrutinio: seq$\langle\mathbb{Z}\rangle$) : seq$\langle$seq$\langle\mathbb{Z}\rangle\rangle$

Donde:

- cantBancas: es la cantidad de bancas en disputa en la provincia
- escrutinio: es la cantidad de votos de cada partido en la provincia
- devuelve la matriz de dimensión partidos $\times$ cocientes de los cocientes del método d'Hondt.

4.1. Especificación

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

`requiere` $\{sinRepetidos(escrutinio) \land\ votosValidos(escrutinio)\}$

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

5. Ejercicio 5: obtenerDiputadosEnProvincia

obtenerDiputadosEnProvincia: calcula la cantidad de bancas de diputados obtenidas por cada partido en una provincia.
proc obtenerDiputadosEnProvincia (in cant_bancas: Z, in escrutinio: seq⟨Z⟩, in dHondt: seq⟨seq⟨Z⟩⟩) : seq⟨Z⟩
Donde:

- cant_bancas: es la cantidad de bancas en disputa en la provincia.
- escrutinio: es la cantidad de votos de cada partido en la provincia.
- dHondt: es la matriz de dimensión #partidos × #cocientes de los cocientes del método dHondt.
- devuelve la cantidad de bancas obtenidas por cada partido.

5.1. Especificación

`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` $\{cant\_bancas > 0\}$

`requiere` $\{sinRepetidos(escrutinio) \land votosValidos(escrutinio)\}$

`requiere` $\{sinComponentesRepetidos(dHondt)\}$

`requiere` $\{dHondtValido(cant\_bancas, escrutinio, dHondt)\}$

`requiere` $\{(\exists i : \mathbb{Z})(0 \leq i < |escrutinio| - 1 \land_L superaUmbral(escrutinio[i], escrutinio))\}$

`asegura` $\{|res| = |escrutinio| - 1\}$

`asegura` $\{(\sum_{i=0}^{|res|-1} res[i]) = cant\_bancas\}$

`asegura` $\{(\forall i : \mathbb{Z})(0 \leq i < |escrutinio| - 1 \rightarrow_L (\neg(superaUmbral(escrutinio[i], escrutinio)) \land res[i] = 0) \lor (superaUmbral(escrutinio[i], escrutinio) \land (\exists s : seq\langle\mathbb{Z}\rangle)(sonLosMaximos(s, dHondt, cant\_bancas) \land$ res[i]=$\sum_{j=0}^{cant\_bancas}$ **if** $pertenece(dHondt[i][j], s)$ **then** 1 **else** 0 **fi**$))\}$

`pred` `sonLosMaximos` (s: $seq\langle\mathbb{Z}\rangle$, matriz: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, n:$\mathbb{Z}$) {
$(esMaximoComponente(s[0], matriz) \land (\forall i : \mathbb{Z})(0 \leq i < n - 1 \rightarrow_L (s[i] > s[i+1]) \land (perteneceAMatriz(s[i], matriz))$
}

`pred` `esMaximoComponente` (elem : $\mathbb{Z}$, matriz: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
$matrizNoVacia(matriz) \land (\forall i, j : \mathbb{Z})(0 \leq i < |matriz| \land 0 \leq j < |matriz[i]| \rightarrow_L elem > matriz[i][j])$
}

`pred` `pertenece` (elem: $\mathbb{Z}$, lista: $seq\langle\mathbb{Z}\rangle$) {
$(|lista| > 0) \land (\exists j : \mathbb{Z})(0 \leq j < |lista| \land_L lista[j] = elem)$
}

`pred` `perteneceAMatriz` (elem: $\mathbb{Z}$, matriz: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
$matrizNoVacia(matriz) \land (\exists i : \mathbb{Z})(0 \leq i < |matriz| \land_L pertenece(elem, matriz[i]))$
}

`pred` `superaUmbral` (votos: $\mathbb{Z}$, escrutinio: seq$\langle\mathbb{Z}\rangle$) {
$votos \geq (\sum_{j=0}^{|escrutinio|-1} escrutinio[j]) * 0{,}03$
}

`pred` `sinComponentesRepetidos` (matriz: seq$\langle$seq$\langle\mathbb{Z}\rangle\rangle$) {
$matrizNoVacia(matriz) \land (\forall i, j, k, s : \mathbb{Z})(0 \leq i, k < |matriz| \land 0 \leq j, s < |matriz[i]| \land (i, j) \neq (k, s) \rightarrow_L matriz[i][j] \neq matriz[k][s])$
}

`pred` `dHondtValido` (cant_bancas: $\mathbb{Z}$, escrutinio: seq$\langle\mathbb{Z}\rangle$, dHondt: seq$\langle$seq$\langle\mathbb{Z}\rangle\rangle$) {
$(|escrutinio| = |dHondt|) \land ((\forall i, j : \mathbb{Z})(0 \leq i < |escrutinio| - 1 \land 0 \leq j < cant\_bancas \rightarrow_L |dHondt[i]| = cant\_bancas \land dHondt[i][j] = \frac{escrutinio[i]}{j+1}))$
}

6. Ejercicio 6: validarListasDiputadosEnProvincia

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 cant_bancas: Z, in listas: seq⟨seq⟨dni : Z × genero : Z⟩⟩) : Bool

Donde:

- cant_bancas: es la cantidad total de bancas en disputa en la provincia.
- listas: son las listas de diputados de cada partido. Cada candidato/a está representado/a con una tupla que contiene el dni y el género.
- devuelve verdadero sii las listas de todos los partidos: 1) presentan la cantidad correcta de candidatos, y 2) verifican la alternancia de género.

6.1. Especificación

`proc` `validarListasDiputadosEnProvincia` (in cant_bancas: $\mathbb{Z}$, in listas: seq⟨seq⟨dni: $\mathbb{Z}$ × genero: $\mathbb{Z}$⟩⟩) : Bool

`requiere` $\{cant\_bancas > 0\}$

`requiere` $\{matrizNoVacia(listas)\}$

`requiere` $\{(\forall i, j : \mathbb{Z})(0 \leq i < |listas| \land 0 \leq j < |listas[0]| \rightarrow_{\mathrm{L}} (listas[i][j][0] > 0) \land ((listas[i][j][1] = 1) \lor (listas[i][j][1] = 2)))\}$

`asegura` $\{res = True \leftrightarrow ((\forall i : \mathbb{Z})(0 \leq i < |listas| \rightarrow_{\mathrm{L}} |listas[i]| = cant\_bancas) \land (\forall i : \mathbb{Z})(0 \leq i < |listas| \rightarrow_{\mathrm{L}} (\forall j : \mathbb{Z})(0 \leq j < |listas[i]| - 1 \rightarrow_{\mathrm{L}} listas[i][j][1] \neq listas[i][j+1][1]))))\}$

6.2. Implementación

```
res := True;
partido_index := 0;

while ( partido_index < listas.size() ) do

    partido := listas[partido_index];

    if cant_bancas != partido.size() then
        res := False;
    else
        skip
    endif

    diputado_index := 0;

    while ( diputado_index < partido.size() - 1 ) do

        if ( partido[diputado_index][1] == partido[diputado_index + 1][1] ) then
            res := False;
        else
            skip
        endif

        diputado_index := diputado_index + 1;

    endwhile

    partido_index := partido_index + 1;

endwhile
```

Código 5: Codigo de Ej6