



Trabajo práctico 1: Especificación y WP

Elecciones Nacionales

14 de septiembre de 2023

Algoritmos y Estructuras de Datos

`sudo_rm-rf_/*`

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



Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Ciudad Universitaria - (Pabellón I/Planta Baja)

Intendente Güiraldes 2610 - C1428EGA

Ciudad Autónoma de Buenos Aires - Rep. Argentina

Tel/Fax: (+54 +11) 4576-3300

<http://www.exactas.uba.ar>

1. Especificación

1.1. hayBallotage

1.1.1. Main

```
proc hayBallotage (in escrutinio : seq⟨ℤ⟩) : Bool
  requiere {eleccionValida(escrutinio)}
  asegura {res = ¬((cond1HayBallotage(escrutinio)) ∨L (cond2HayBallotage(escrutinio)))}
```

1.1.2. Predicados Especificos

```
pred cond1HayBallotage (in escrutinio : seq⟨ℤ⟩) {
  (∃n : ℤ)(0 ≤ n < |escrutinio| ∧L (porcentajeDeVotos(escrutinio, escrutinio[x]) > 45))
}
pred cond2HayBallotage (in escrutinio : seq⟨ℤ⟩) {
  (∃n : ℤ)(0 ≤ n < |escrutinio| ∧L (porcentajeDeVotos(escrutinio, escrutinio[x]) > 40) ∧L
  ¬(∃x : ℤ)(0 ≤ x < |escrutinio| ∧L (¬(n = x) ∧L ((escrutinio[n] - escrutinio[x]) > 10)))
}
```

1.2. hayFraude

1.2.1. Main

```
proc hayFraude ((in escrutinio_Presidencial: seq⟨ℤ⟩, in escrutinio_senadores: seq⟨ℤ⟩), in escrutinio_diputados: seq⟨ℤ⟩) : Bool
  requiere {umbralElectoral(escrutinio_senadores) ∧L eleccionValida(escrutinio_Presidencial) ∧L
  eleccionValida(escrutinio_senadores) ∧L eleccionValida(escrutinio_diputados)}
  asegura {res = ¬(((sumaDeVotos(escrutinio_Presidencial) = sumaDeVotos(escrutinio_Senadores)) ∧L
  (sumaDeVotos(escrutinio_Presidencial) = sumaDeVotos(escrutinio_Diputados))))}
```

1.3. obtenerSenadoresEnProvincia

1.3.1. Main

```
proc obtenerSenadoresEnProvincia (in escrutinio : seq⟨ℤ⟩) : ℤ×ℤ
  requiere {eleccionValida(escrutinio) ∧ minimoDePartidos(escrutinio)}
  asegura {(∃!x : ℤ)(0 ≤ x < |escrutinio| - 1) ∧L ((∃!y : ℤ)(0 ≤ y < |escrutinio| - 1) ∧L ((∀i : ℤ)(0 ≤ i <
  |escrutinio| ∧ ¬(i = x) ∧ ¬(i = y)) →L escrutinio[i] < escrutinio[resy] < escrutinio[resx]))}
```

1.3.2. Predicados Especificos

```
pred minimoDePartidos (in escrutinio: seq⟨ℤ⟩) {
  |escrutinio| ≥ 3
}
```

1.4. calcularDHondtEnProvincia

1.4.1. Main

```
proc calcularDHondtEnProvincia (in cant_bancas: ℤ, in escrutinio: seq⟨ℤ⟩) : seq⟨seq⟨ℤ⟩⟩
  requiere {eleccionValida(escrutinio) ∧ umbralElectoral(escrutinio) ∧ cant_bancas > 0}
  asegura {(∀x : ℤ)(0 ≤ n < cant_bancas) ∧L (∀x : ℤ)(0 ≤ n < |escrutinio|) →L (res[x][n] =  $\frac{\text{escrutinio}[x]}{n+1}$ )}
```

1.5. obtenerDiputadosEnProvincia

1.5.1. Main

```
proc obtenerDiputadosEnProvincia (in cant_bancas: ℤ, in escrutinio: seq⟨ℤ⟩, in dHondt: seq⟨seq⟨ℤ⟩⟩) : seq⟨ℤ⟩
  requiere {eleccionValida(escrutinio) ∧ umbralElectoral(escrutinio)}
  asegura {(∀r : ℤ)(0 ≤ r < |escrutinio| - 1 →L res[r] = bancasDe(r, cant_bancas, dHondt))}
```

1.5.2. Predicados Especificos

```
pred cocienteGanador (in indicePartido:  $\mathbb{Z}$ , in bancaEnDisputa:  $\mathbb{Z}$ , in dHont:  $seq\langle seq\langle \mathbb{Z} \rangle \rangle$ ) {
  res = True  $\longleftrightarrow$  ( $\forall i : \mathbb{Z}$ )( $0 \leq i < |dHont| - 1 \wedge \neg(i = indicePartido) \longrightarrow_L dHont[bancaEnDisputa][indicePartido] >$ 
  dHont[bancaEnDisputa][i])
}
```

1.6. validarListasDiputadosEnProvincia

1.6.1. 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)  $\wedge$  (dni > 0)  $\wedge$  (1  $\leq$  genero  $\leq$  2)}
  asegura {( $\forall x : \mathbb{Z}$ )( $0 \leq x < |listas|$ )  $\longrightarrow_L$  (cantCandidatosCorrecta(cant_bancas, listas[x])  $\wedge_L$  altGenero(listas[x]))}
```

1.6.2. Predicados Especificos

```
pred cantCandidatosCorrecta (cant_bancas:  $\mathbb{Z}$ , partido:  $seq\langle dni : \mathbb{Z} \times genero : \mathbb{Z} \rangle$ ) {
  cant_bancas = |partido|
}
pred altGenero (partido:  $seq\langle dni : \mathbb{Z} \times genero : \mathbb{Z} \rangle$ ) {
  (( $\forall n : \mathbb{Z}$ )( $n > 0$ ))  $\longrightarrow_L$  (( $(n \bmod 2 = 0) \longrightarrow_L (partido[n, 1] = 1)$ )  $\wedge_L$  (( $n \bmod 2 = 1$ )  $\longrightarrow_L (partido[n, 1] = 2)$ )  $\vee_L$  (( $n \bmod 2 = 0$ )  $\longrightarrow_L (partido[n, 1] = 2 \wedge_L (n \bmod 2 = 1) \longrightarrow_L partido[n, 1] = 1)$ ))
}
```

1.7. Auxiliares

```
aux sumaDeVotos (in escrutinio :  $seq\langle \mathbb{Z} \rangle$ ) :  $\mathbb{Z} = \sum_{i=0}^{|escrutinio|-1} escrutinio[i]$ ;
aux porcentajeDeVotos (in escrutinio:  $seq\langle \mathbb{Z} \rangle$ , in partido:  $seq\langle \mathbb{Z} \rangle$ ) :  $\mathbb{R} = sumaDeVotos(escrutinio)^{-1} * escrutinio[partido] * 10^2$ ;
aux bancasDe (in indicePartido:  $\mathbb{Z}$ , in bancas, in dHont  $seq\langle seq\langle \mathbb{Z} \rangle \rangle$ ) :  $\mathbb{Z} = \sum_{p=0}^{bancas-1} if\ cocienteGanador(indicePartido, p, dHont) then 1 else 0$ ;
```

1.8. 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 escrutinioSen :  $seq\langle \mathbb{Z} \rangle$ ) {
  (( $\forall x : \mathbb{Z}$ )( $0 \leq x < |escrutinio| \longrightarrow_L (escrutinioSen[x] > 3)$ ))
}
```

2. Implementaciones y demostraciones de correctitud

2.1. Implementaciones

2.1.1. hayBallotage

```
1      res := true
2      tans := 0
3      primero := 0
4      segundo := 0
5      i := 0
6      suma := 0
7      while (escrutinio.size() > i) do
8          suma:= suma + escrutinio[i]
9          i := i + 1
10     endwhile
11     i := 0
12     while (escrutinio.size() > i) do
13         escrutinio[i] := (escrutinio[i] * 100)/suma
14         i := i + 1
15     endwhile
16     i := 0
17     while (escrutinio.size() > i) do
18         if (segundo < escrutinio[i]) then
19             segundo := escrutinio[i]
20         else:
21             skip
22         endif
23         if (primero < segundo) then
24             trans := primero
25             primero := segundo
26             segundo := trans
27         else:
28             skip
29         endif
30         i := i + 1
31     endwhile
32     if (primero > 45) then
33         res := false
34     else
35         if ((primero > 40) && (primero - segundo >= 10)) then
36             res := false
37         else
38             skip
39         endif
40     endif
41     endif
```

Código 1: ()

2.1.2. hayFraude

```
1      res := true
2      sumaPres := 0
3      sumaDip := 0
4      sumaSen := 0
5      while (escrutinio_Presidencial.size() > i) do
6          sumaPres:= sumaPres + escrutinio_Presidencial[i]
7          i := i + 1
8      endwhile
9      i := 0
10     while (escrutinio_Diputados.size() > i) do
11         sumaDip:= sumaDip + escrutinio_Diputados[i]
12         i := i + 1
13     endwhile
14     i := 0
15     while (escrutinio_Senadores.size() > i) do
16         sumaSen := sumaSen + escrutinio_Senadores[i]
17         i := i + 1
18     endwhile
19     if (sumaPres = sumaDip && sumaPres = sumaSen) then
20         res := false
21     else:
22         skip
23     endif
```

Código 2: ()

2.1.3. obtenerSenadoresEnProvincia

```
1      trans := 0
2      primero := 0
3      segundo := 0
4      i := 0
5      while (escrutinio.size() > i) do
6          if (escrutinio[segundo] < escrutinio[i]) then
7              segundo := i
8          else:
9              skip
10         endif
11         if (escrutinio[primero] < escrutinio[segundo]) do
12             trans := primero
13             primero := segundo
14             segundo := trans
15         else:
16             skip
17         endif
18         i := i + 1
19     endwhile
20     res := (primero, segundo)
```

Código 3: ()

2.1.4. validarListasDiputadosEnProvincia

```

1      res := true
2      i := 0
3      while (listas.size() > i) do
4          if (listas[i].size() != cant_bancas) then
5              res:= false
6          else:
7              skip
8          endif
9          i := i + 1
10     endwhile
11     i := 0
12     j := 1
13     while (listas.size() > i) do
14         genero := listas[i][0][1]
15         while (listas[i].size() > j) do
16             if (listas[i][j][1] == genero) then
17                 res:=false
18             else:
19                 genero := listas[i][j][1]
20                 j := j + 1
21             endif
22         endwhile
23         i := i + 1
24     endwhile

```

Código 4: ()

Lo principal: las fórmulas. Se puede poner en una linea, como $x_i = x_{i-1} + x_{i-2}$, o ponerse más grande:

$$\sum_{i=0}^n i \quad (1)$$

Y se pueden citar ecuaciones con `\eqref{nombreDeEq}`: (1)

Ejemplo de itemizado:

- Item 1
- Item 2
- Item 3

Ejemplo de enumerado con menor distancia entre items:

1. Item 1
2. Item 2
3. Item 3

Podemos escribir mucho texto. Mucho texto. Mucho texto. Mucho texto. Mucho texto. Mucho texto. Mucho texto. Mucho texto. Mucho texto. Mucho texto. Mucho texto. Mucho texto.

Otro párrafo. Otro párrafo. Otro párrafo. Otro párrafo. Otro párrafo. Otro párrafo. Otro párrafo. Otro párrafo. Otro párrafo. Otro párrafo. Otro párrafo. Otro párrafo.

Le agregamos una separación entre párrafos. Le agregamos una separación entre párrafos. Le agregamos una separación entre párrafos. Le agregamos una separación entre párrafos. Le agregamos una separación entre párrafos.

La tabla 1 es un ejemplo de cómo se hace una tabla.

La figura 2 es un ejemplo de cómo se agrega una imagen.

```

1  res := 0;
2  i := 0;
3  while (i < s.size()) do
4      res := res + s[i];
5      i := i + 1
6  endwhile

```

Código 5: Ejemplo de código (usando los estilos de la cátedra, ver las macros para más detalles)

Col1	Col2	Col2	Col3
1	6	87837	787
2	7	78	5415
3	545	778	7507
4	545	18744	7560
5	88	788	6344

Tabla 1: Ejemplo de tabla



Figura 1: Ejemplo de figura

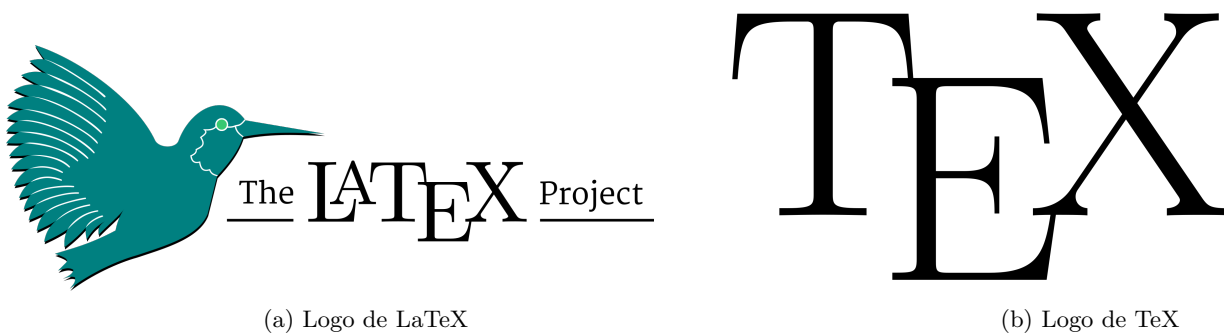


Figura 2: Ejemplo para poner dos figuras juntas. Y citarlas por separado a (a) y (b).

Si se pone un label al `\lstlisting`, se puede referenciar: Código 5.

2.2. Macros de la cátedra para especificar

```

proc nombre (in paramIn : N, inout paramInout : seq(Z)) : tipoRes
  requiere {expresionBooleana1}
  asegura {expresionBooleana2}
  aux auxiliar1 (parametros) : tipoRes = expresion ;
  pred pred1 (parametros) {
    expresion
  }

aux auxiliarSuelto (parametros) : tipoRes = expresion ;
pred predSuelto (parametros) {
  (∀variable : tipo) (algo →L expresion)
}
pred predSuelto (parametros) {
  (∃variable : tipo) (algo ∧L expresion)
}

```