DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Trabajo práctico 1: Especificacíon 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 : $seq\langle\mathbb{Z}\rangle$) : Bool
requiere $\{eleccionValida(escrutinio))\}$
asegura $\{res = \neg((cond1HayBallotage(escrutinio)) \vee_L (cond2HayBallotage(escrutinio)))\}$

1.1.2. Predicados Especificos

pred cond1HayBallotage (in escrutinio : $seq\langle\mathbb{Z}\rangle$) {
$(\exists n : \mathbb{Z})(0 \leq n < |escrutinio| \wedge_L (porcentajeDeVotos(escrutinio, escrutinio[x]) > 45)$
}
pred cond2HayBallotage (in escrutinio : $seq\langle\mathbb{Z}\rangle$) {
$(\exists n : \mathbb{Z})(0 \leq n < |escrutinio| \wedge_L (porcentajeDeVotos(escrutinio, escrutinio[x]) > 40) \wedge_L$
$\neg(\exists x : \mathbb{Z})(0 \leq x < |escrutinio| \wedge_L (\neg(n = x) \wedge_L ((escrutinio[n] - escrutinio[x]) > 10))$
}

1.2. hayFraude

1.2.1. Main

proc hayFraude ((in escrutinio_Presidencial: $seq\langle\mathbb{Z}\rangle$, in escrutinio_senadores: $seq\langle\mathbb{Z}\rangle\rangle$, in escrutinio_diputados: $seq\langle\mathbb{Z}\rangle$) : Bool
requiere $\{umbralElectoral(escrutinio\_senadores) \wedge_L eleccionValida(escrutinio\_Presidencial) \wedge_L$
$eleccionValida(escrutinio\_senadores) \wedge_L eleccionValida(escrutinio\_diputados)\}$
asegura $\{res = \neg(((|sumaDeVotos(escrutinio\_Presidencial) = sumaDeVotos(escrutinio\_Senadores)) \wedge_L$
$(sumaDeVotos(escrutinio\_Presidencial) = sumaDeVotos(escrutinio\_Diputados)))\}$

1.3. obtenerSenadoresEnProvincia

1.3.1. Main

proc obtenerSenadoresEnProvincia (in escrutinio : $seq\langle\mathbb{Z}\rangle$) : $\mathbb{Z}\times\mathbb{Z}$
requiere $\{eleccionValida(escrutinio) \wedge minimoDePartidos(escrutinio)\}$
asegura $\{(\exists! x : \mathbb{Z})(0 \leq x < |escrutinio| - 1) \wedge_L ((\exists! y : \mathbb{Z})(0 \leq y < |escrutinio| - 1) \wedge_L ((\forall i : \mathbb{Z})(0 \leq i <$
$|escrutinio| \wedge \neg(i = x) \wedge \neg(i = y)) \longrightarrow_L escrutinio[i] < escrutinio[res_y] < escrutinio[res_x]))\}$

1.3.2. Predicados Especificos

pred minimoDePartidos (in escrutinio: $seq\langle\mathbb{Z}\rangle$) {
$|escrutinio| \geq 3$
}

1.4. calcularDHondtEnProvincia

1.4.1. Main

proc calcularDHondtEnProvincia (in cant_bancas: $\mathbb{Z}$, in escrutinio: $seq\langle\mathbb{Z}\rangle$) : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$
requiere $\{eleccionValida(escrutinio) \wedge umbralElectoral(escrutinio) \wedge cant\_bancas > 0)\}$
asegura $\{((\forall x : \mathbb{Z})(0 \leq n < cant\_bancas) \wedge_L (\forall x : \mathbb{Z})(0 \leq n < |escrutinio|)) \longrightarrow_L (res[x][n] = \frac{escrutinio[x]}{n+1})\}$

1.5. obtenerDiputadosEnProvincia

1.5.1. Main

proc obtenerDiputadosEnProvincia (in cant_bancas: $\mathbb{Z}$, in escrutinio: $seq\langle\mathbb{Z}\rangle$, in dHondt: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) : $seq\langle\mathbb{Z}\rangle$
requiere $\{eleccionValida(escrutinio) \wedge umbralElectoral(escrutinio)\}$
asegura $\{(\forall r : \mathbb{Z})(0 \leq r < |escrutinio| - 1 \longrightarrow_L res[r] = bancasDe(r, cant\_bancas, dHondt))\}$

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 $\{(cant\_bancas > 0) \wedge (dni > 0) \wedge (1 \leq genero \geq 2)\}$
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: $seq\langle dni : \mathbb{Z} \times genero : \mathbb{Z}\rangle$) {
$(((\forall n : \mathbb{Z})(n > 0)) \longrightarrow_L (((n \mod 2 = 0) \longrightarrow_L (partido[n, 1] = 1)) \wedge_L ((n \mod 2 = 1) \longrightarrow_L (partido[n, 1] = 2)) \vee_L ((n \mod 2 = 0) \longrightarrow_L (partido[n, 1] = 2 \wedge_L (n \mod 2 = 1) \longrightarrow_L partido[n, 1] = 1))$
}

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 escrutinioValdio (in escrutinio: $seq\langle\mathbb{Z}\rangle$) {
$|escrutinio| \geq 2$
}
pred EleccionValida (in escrutinio: $seq\langle\mathbb{Z}\rangle$) {
$nohayRepetidos(escrutinio) \wedge cantVotosValidos(escrutinio) \wedge escrutinioValido(escrutinio)$
}
pred umbralElectoral (in 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

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

Código 1: ()

2.1.2. hayFraude

```
res := True
sumaPres:=0
sumaDip:=0
sumaSen:=0
while (escrutinio_Presidencial.size() > i) do
    sumaPres:= sumaPres + escrutinio_Presidencial[i]
    i:=i+1
endwhile
i:=0
while (escrutinio_Diputados.size() > i) do
    sumaDip:= sumaDip + escrutinio_Diputados[i]
    i:=i+1
endwhile
i:=0
while (escrutinio_Senadores.size() > i) do
    sumaSen:= sumaSen + escrutinio_Senadores[i]
    i:=i+1
endwhile
if (sumaPres = sumaDip && sumaPres = sumaSen) then
    res := False
else:
    skip
endif
```

Código 2: ()

2.1.3. obetenerSenadores

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

Código 3: ()

2.1.4. obetenerSenadores

```
                    res1:=True

                    i :=0
                    while (listas.size() - 1 > i) do
                        If (listas[i].size() != listas[i+1].size()) then
                            res:=false
                        else:
                            skip
                        endif
                        i := i+1
                    endhwihile
                    if (listas[0].size() != cant_bancas) then
                        res := false
                    else:
                        skip
                    endif
                    i := 0
                    j := 0
                    while (listas.size() > i) do
                        aux := listas[i][0][1]
                        while (listas[i].size() - 1 > j) do
                            if (listas[i][j][1] == listas[i][0][1] && listas[i][j+1][1] != listas[i][0][1]
                                then
                                res:=false
                            else:
                                skip
                            endif
                            j := j + 2
                        endwhile
                        i := i + 1
                    endwhile
%%%%%%%%%%%%%%%%%%%%PROBLEMAS CON CANDIDATOS IMPARES
```

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:

$$\sum_{i=0}^{n} i \tag{1}$$

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.

Otro párrafo. 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.

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

(a) Logo de LaTeX

(b) Logo de TeX

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

```
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)

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)
}
```