DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Trabajo práctico 1: Especificacíon y WP

Elecciones Nacionales

17 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. General

1.1.1. Predicados Universales

pred noHayRepetidos (in escrutinio : $seq\langle\mathbb{Z}\rangle$) {
$(\forall x : \mathbb{Z})(0 \leq x < |escrutinio| \longrightarrow_L ((\forall y : \mathbb{Z})(0 \leq y < |escrutinio| \wedge \neg(x = y) \longrightarrow_L \neg(escrutinio[x] = escrutinio[y])))$
}
pred cantVotosValidos (in escrutinio : $seq\langle\mathbb{Z}\rangle$) {
$((\forall x : \mathbb{Z})(0 \leq x < |escrutinio|) \longrightarrow_L (escrutinio[x] \geq 0))$
}
pred 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 : $seq\langle\mathbb{Z}\rangle$) {
$((\forall x : \mathbb{Z})(0 \leq x < |escrutinio|) \longrightarrow_L (escrutinioSen[x] > 3))$
}
pred minimoDePartidos (in escrutinio: $seq\langle\mathbb{Z}\rangle$) {
$|escrutinio| \geq 3$
}

1.1.2. Auxiliares

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

1.2. hayBallotage

1.2.1. Main

proc hayBallotage (in escrutinio : $seq\langle\mathbb{Z}\rangle$) : Bool
requiere {$eleccionValida(escrutinio)$)}
asegura {$res = \neg((partidoMayorA45\,\%(escrutinio)) \vee (partidoMayorA40\,\%ConDiferencia(escrutinio)))$}

1.2.2. Predicados Especificos

pred partidoMayorA45% (in escrutinio : $seq\langle\mathbb{Z}\rangle$) {
$(\exists n : \mathbb{Z})(0 \leq n < |escrutinio| - 1 \wedge_L (porcentajeDeVotos(escrutinio, escrutinio[n]) > 45)$
}
pred partidoMayorA40%ConDiferencia (in escrutinio : $seq\langle\mathbb{Z}\rangle$) {
$(\exists n : \mathbb{Z})(0 \leq n < |escrutinio| - 1 \wedge_L (porcentajeDeVotos(escrutinio, escrutinio[n]) > 40) \wedge_L$
$\neg(\forall x : \mathbb{Z})(0 \leq x < |escrutinio| - 1 \wedge (\neg(n = x) \longrightarrow_L ((escrutinio[n] - escrutinio[x]) > 10))$
}

1.3. hayFraude

1.3.1. Main

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 {$umbralElectoral(escrutinio\_senadores) \wedge eleccionValida(escrutinio\_Presidencial) \wedge eleccionValida(escrutinio\_senadores) \wedge eleccionValida(escrutinio\_diputados) \wedge minimoDePartidos(escrutinio\_Senadores)$
asegura {$res = \neg(((|sumaDeVotos(escrutinio\_Presidencial) = sumaDeVotos(escrutinio\_Senadores)) \wedge (sumaDeVotos(escrutinio\_Presidencial) = sumaDeVotos(escrutinio\_Diputados)))$}

1.4. obtenerSenadoresEnProvincia

1.4.1. Main

proc obtenerSenadoresEnProvincia (in escrutinio : $seq\langle\mathbb{Z}\rangle$) : $\mathbb{Z}\times\mathbb{Z}$
requiere $\{eleccionValida(escrutinio) \land minimoDePartidos(escrutinio)\}$
asegura $\{(\exists!x : \mathbb{Z})(0 \leq x < |escrutinio| - 1 \land_L ((\exists!y : \mathbb{Z})(0 \leq y < |escrutinio| - 1 \land_L ((\forall i : \mathbb{Z})(0 \leq i < |escrutinio| - 1 \land \neg(i = x) \land \neg(i = y) \longrightarrow_L escrutinio[i] < escrutinio[res_1 = y] < escrutinio[res_0 = x])))))\}$

1.4.2. Predicados Especificos

1.5. calcularDHondtEnProvincia

1.5.1. Main

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

1.6. obtenerDiputadosEnProvincia

1.6.1. Main

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

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

1.7. validarListasDiputadosEnProvincia

1.7.1. Main

proc (in cant_bancas: $\mathbb{Z}$, in listas: $seq\langle seq\langle dni : \mathbb{Z} \times genero : \mathbb{Z}\rangle\rangle$ (Bool) :
requiere $\{(cant\_bancas > 0) \land (dni > 0) \land (1 \leq genero \geq 2)\}$
asegura $\{(\forall\ partido : \mathbb{Z})(0 \leq partido < |listas|) \longrightarrow_L (cantCandidatosCorrecta(cant\_bancas,\ listas[partido]) \land altGenero(listas[partido])\}$

1.7.2. Predicados Especificos

pred cantCandidatosCorrecta (cant_bancas: $\mathbb{Z}$, partido: $seq\langle dni : \mathbb{Z} \times genero : \mathbb{Z}\rangle$) {
$cant\_bancas = |partido|$
}
pred altGenero (partido: $seq\langle dni : \mathbb{Z} \times genero : \mathbb{Z}\rangle$) {
$(((\forall n : \mathbb{Z})(n > 0)) \longrightarrow_L (((n\ \text{ mód } 2 = 0) \longrightarrow_L (partido[n, 1] = 1)) \land_L ((n\ \text{ mód } 2 = 1) \longrightarrow_L (partido[n, 1] = 2)) \lor_L ((n \text{ mód } 2 = 0) \longrightarrow_L (partido[n, 1] = 2 \land_L (n\ \text{ mód } 2 = 1) \longrightarrow_L partido[n, 1] = 1))$
}

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])
19         segundo := escrutinio[i]
20     else:
21         skip
22     endif
23     if (primero < segundo)
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)
33     res := false
34 else
35     if ((primero > 40) && (primero - segundo >= 10))
36         res := false
37     else
38         skip
39     endif
40 endif
```

Código 1: ()

2.1.2. hayFraude

```
                                i:=0
                                sumaPres := 0
                                while (escrutinio_Presidencial.size() > i) do
                                    sumaPres:= sumaPres + escrutinio_Presidencial[i]
                                    i := i + 1
                                endwhile
                                i := 0
                                sumaDip := 0
                                while (escrutinio_Diputados.size() > i) do
                                    sumaDip:= sumaDip + escrutinio_Diputados[i]
                                    i := i + 1
                                endwhile
                                i := 0
                                sumaSen := 0
                                while (escrutinio_Senadores.size() > i) do
                                    sumaSen := sumaSen + escrutinio_Senadores[i]
                                    i := i + 1
                                endwhile
<<<<<<< HEAD
                                res := true
                                if (sumaPres = sumaDip && sumaPres = sumaSen) then
=======
                                if (sumaPres = sumaDip && sumaPres = sumaSen)
>>>>>>> 4044d73e3a04188e7e8761646fac4a6fabb4b00d
                                    res := false
                                else:
                                    skip
                                endif
```

Código 2: ()

2.1.3. obtenerSenadoresEnProvincia

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

Código 3: ()

2.1.4. validarListasDiputadosEnProvincia

```
res := true
i := 0
while (listas.size() > i) do
    if (listas[i].size() != cant_bancas)
        res:= false
    else:
        skip
    endif
    i := i + 1
endwhile
i := 0
j := 1
while (listas.size() > i) do
    genero := listas[i][0][1]
    while (listas[i].size() > j) do
        if (listas[i][j][1] == genero)
            res:=false
        else:
            genero := listas[i][j][1]
            j := j + 1
        endif
    endwhile
    i := i + 1
endwhile
```

Código 4: ()