TP de Especificación

Sudoku

21 de Abril de 2017   Algoritmos y Estructuras de Datos I

Grupo 17

Integrante	LU	Correo electrónico
Maqueda, Ignacio	279/14	ignaciomaqueda95@gmail.com
Parral, Guillermo	280/16	guillermoeparral@gmail.com
Quintela, Gonzalo	089/16	gquintela@dc.uba.ar
Sirio, Tomás	440/16	tomassirio@gmail.com

1. Problemas

```
proc  sudoku_esTableroValido (in t: seq⟨seq⟨ℤ⟩⟩, out result: Bool)  {
```
　　Pre $\{True\}$
　　Post $\{result = esTableroValido(t)\}$
```
}
```

```
proc sudoku_esCeldaVacia (in t: seq⟨seq⟨ℤ⟩⟩, in f: ℤ,in c: ℤ, out result: Bool)  {
```
　　Pre $\{esTableroValido(t) \wedge_L 0 \le f, c < |t|\}$
　　Post $\{result = (t[f][c] = 0)\}$
```
}
```

```
proc sudoku_nroDeCeldasVacias (in t: seq⟨seq⟨ℤ⟩⟩, out result: ℤ)  {
```
　　Pre $\{esTableroValido(t)\}$
　　Post $\{\sum_{i=0}^{|t|-1} (\sum_{j=0}^{|t|-1}$ if $t[i][j] = 0$ then 1 else 0 fi$\}$
```
}
```

```
proc sudoku_primeraCeldaVaciaFila (in t: seq⟨seq⟨ℤ⟩⟩, out result: ℤ)  {
```
　　Pre $\{esTableroValido(t)\}$
　　Post $\{$if $(\exists i : ℤ)(0 \le i{<}|t| \wedge_L filaTieneCeldaVacia(t[i]) \wedge_L (\forall j : ℤ)$
　　　　$(0 \le j{<}i \longrightarrow_L \neg filaTieneCeldaVacia(t[j])))$
　　　　then $result = i$
　　　　else $result = -1$ fi$\}$
```
}
```

```
proc sudoku_primeraCeldaVaciaColumna (in t: seq⟨seq⟨ℤ⟩⟩, out result: ℤ)  {
```
　　Pre $\{esTableroValido(t)\}$
　　Post $\{$if $(\exists i : ℤ)(0 \le i{<}|t| \wedge_L filaTieneCeldaVacia(t[i]) \wedge_L (\forall j : ℤ)(0 \le j{<}i \longrightarrow_L \neg filaTieneCeldaVacia(t[j])))$
　　　　then $result = indicePrimeraCeldaVaciaEnFila(t[i])$
　　　　else $result = -1$ fi$\}$
```
}
```

```
proc sudoku_valorEnCelda (in t: seq⟨seq⟨ℤ⟩⟩, in f: ℤ, in c: ℤ, out result: ℤ)  {
```
　　Pre $\{esTableroValido(t) \wedge_L 0 \le f, c \le 8 \wedge_L t[i][j] \neq 0\}$
　　Post $\{result = t[f][c]\}$
```
}
```

```
proc sudoku_llenarCelda (inout t: seq⟨seq⟨ℤ⟩⟩, in f: ℤ, in c: ℤ, in value: ℤ)  {
```
　　Pre $\{esTableroValido(t) \wedge_L 0 \le f, c \le 8 \wedge_L t[i][j] = 0 \wedge_L 1 \le value \le 9 \wedge_L t = t_0\}$
　　Post $\{t[f][c] = value \wedge_L (\forall i : ℤ)(\forall j : ℤ)(0 \le i, j{<}|t| \wedge_L (i \neq f \vee j \neq c)) \longrightarrow_L t[i][j] = t_0[i][j]\}$
```
}
```

```
proc sudoku_vaciarCelda (inout t: seq⟨seq⟨ℤ⟩⟩, in f: ℤ, in c: ℤ)  {
```
　　Pre $\{esTableroValido(t) \wedge_L 0 \le f, c \le 8 \wedge_L t[i][j] \neq 0 \wedge_L t = t_0\}$
　　Post $\{t[f][c] = 0 \wedge_L (\forall i : ℤ)(\forall j : ℤ)((0 \le i, j{<}|t| \wedge_L (i \neq f \vee j \neq c)) \longrightarrow_L t[i][j] = t_0[i][j])\}$
```
}
```

```
proc sudoku_esTableroParcialmenteResuelto (in t: seq⟨seq⟨ℤ⟩⟩, out result: Bool))  {
```
　　Pre $\{True\}$
　　Post $\{result = esTableroParcialmenteResuelto(t)\}$
```
}
```

```
proc sudoku_esTableroTotalmenteResuelto (in t: seq⟨seq⟨ℤ⟩⟩, out result: Bool)  {
        Pre {true}
        Post {esParcialmenteResuelto(t)∧_L ∀i : ℤ)(0 ≤ i < |t|⟶_L ¬filaTieneCeldaVacia(t[i])}
}


proc sudoku_esTablero (in t_0, t_1 : seq⟨seq⟨ℤ⟩⟩, out result : Bool){
        Pre {true}
        Post {esSubtablero(t_0, t_1) = result}
}


proc sudoku_esSubTablero (in t_0, t_1 : seq⟨seq⟨ℤ⟩⟩, out result : Bool){
        Pre {true}
        Post {esSubtablero(t_0, t_1) = result}
}
```

2.   Predicados y Auxiliares generales

```
pred esMatriz (t: seq⟨seq⟨ℤ⟩⟩) {
(∀i : ℤ)(∀j : ℤ)(0 ≤ i, j < |t|⟶_L | t[i] | = | t[j] |)}
```

```
fun cantidadFilas (t: seq⟨seq⟨ℤ⟩⟩) : ℤ = |t| ;
```

```
fun cantidadColumnas (t: seq⟨seq⟨ℤ⟩⟩) : ℤ = if cantidadFilas(t) > 0 then |t[0]| else 0 fi ;
```

```
pred esMatrizCuadrada (t: seq⟨seq⟨ℤ⟩⟩) {
esMatriz(t) ∧ ( cantidadFilas(t) = cantidadColumnas(t) )}
```

```
pred esTableroValido (t: seq⟨seq⟨ℤ⟩⟩) {esMatrizCuadrada(t)∧_L |t|=9 ∧_L
(∀i : ℤ)(∀j : ℤ)(0 ≤ i, j < |t|⟶_L 0 ≤ t[i][j] ≤ 9)}
```

```
pred filaTieneCeldaVacia (f: seq⟨ℤ⟩) {
(∃i : ℤ)(0 ≤ i<|f| ∧_L f[i] = 0)}
```

```
fun indicePrimeraCeldaVaciaEnFila (s: seq⟨ℤ⟩) : ℤ = if ((∃ : iℤ)(0 ≤ i<|s| ∧_L s[i] = 0 ∧_L
(∀j : ℤ)(0 ≤ j<i ⟶_L s[j] ≠ 0))) then i else − 1 fi ;
```

```
pred noHayRepetidosEnRegion (s: seq⟨ℤ⟩) {(∀i : ℤ)(∀j : ℤ)(∀k : ℤ)(∀l : ℤ)
(0 ≤ i, j, k, l<9 ∧_L (idiv3 = kdiv3) ∧_L (jdiv3 = ldiv3) ∧_L (i ≠ k ∨ j ≠ l) ⟶_L t[i][j] ≠ t[k][l]}
```

```
pred noHayRepetidosEnFila (s: ℤ) {(∀i : ℤ)(∀j : ℤ)(0 ≤ i, j<|s| ∧_L j ≠ i ⟶_L s[i] ≠ s[j])}
```

```
pred noHayRepetidosEnColumna (t: ℤ) {(∀j : ℤ)(0 ≤ j<|t| ⟶_L (∀l : ℤ)(∀k : ℤ)
(0 ≤ l, k<|t| ∧_L l ≠ k ⟶_L t[i][j] ≠ t[k][j]))}
```

```
pred esTableroParcialmenteResuelto (t: ℤ) {esTableroValido(t) ∧_L (∀i : ℤ)(0 ≤ i<|t| ⟶_L
(noHayRepetidosEnFila(t[i]) ∧_L noHayRepetidosEnColumna(t) ∧_L noHayRepetidosEnReligion(t))}
```

```
pred esSubTablero (t_0, t_1 : seq⟨seq⟨ℤ⟩⟩){
( esTableroValido(t_0) ∧_L esTableroValido(t_1)) ∧_L ((∀i : ℤ)(∀j : ℤ)
(0 ≤ i|t| ∧_L t_0[i][j] ≠ 0) ⟶_L (t_0[i][j] = t_1[i][j])}
```

3.   Decisiones tomadas