DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Trabajo práctico

Especificacion de TADs

September 14, 2025 — Algoritmos y Estructuras de Datos

BobElConstructorPorCopia

Integrante	LU	Correo electrónico
Choque, Leandro	252/25	leandroch2002@gmail.com
Musi, Santino	965/24	santinomusi1@gmail.com
Rojas, Damian	209/25	dam.rojas1@gmail.com
Martell, Juan Bautista	622/25	Juanbamartell@hotmail.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 Supongo que acá iría una descripción

Breve descripción.

Luego veremos bien el formato, esto de momento es para tener un esqueleto.

2 Especificacion

TAD EdR {

obs aula : $Aula$

obs solucion : $Paso$

obs entregas : seq$\langle\langle Coordenadas, Paso\rangle\rangle$

proc EdR(in $dimensionAula : \mathbb{Z}$, in $s : Paso$, in $cantEstudiantes : \mathbb{Z}$) : EdR {

requiere {

$(dimensionAula > 0) \wedge_L$

$rtaValida(s) \wedge_L$

$cantValidaEstudiantes(dimensionAula, cantEstudiantes)$

}

asegura {

$(|res.aula| = dimensionAula) \wedge_L$

$aulaCuadrada(res.aula) \wedge_L$

$noHayAlumnosJuntos(res.aula) \wedge_L$

$(cuantosEstudiantes(res.aula) = cantEstudiantes) \wedge_L$

$examenesSinResponder(res.aula) \wedge_L$

$(res.solucion = s) \wedge_L$

$(res.entregas = \langle\rangle)$

}

}

pred rtaValida($s : Paso$) {

$(\forall i : \mathbb{Z})\ (0 \leq i < |s| \rightarrow_L s[i] \in$ conj$\langle$"0","1","2","3","4","5","6","7","8","9"$\rangle)$

}

pred cantValidaEstudiantes($a : Aula, e : \mathbb{Z}$) {

$(e \leq ifThenElse(mod(|a|, 2) == 0, \frac{|a|^2}{2}, \frac{|a|+1}{2} * |a|))$

}

pred aulaCuadrada($a : Aula$) {

$(\forall i : \mathbb{Z})\ (0 \leq i < |a| \rightarrow_L |a[i]| = |a|)$

}

pred noHayAlumnosJuntos($a : Aula$) {

$(\forall i : \mathbb{Z})\ (0 \leq i < |a| \rightarrow_L$

$(\forall j : \mathbb{Z})\ (0 \leq j < |a[i]| - 1 \rightarrow_L (a[i][j] \neq \langle\rangle \rightarrow a[i][j+1] = \langle\rangle)))$

}

aux cuantosEstudiantes($a : Aula$) : $\mathbb{Z} = \sum_{i=0}^{|a|-1} \sum_{j=0}^{|a[i]|-1} ifThenElse(a[i][j] \neq \langle\rangle, 1, 0)$

pred examenesSinResponder($a : Aula$) {

$(\forall i : \mathbb{Z})\ (0 \leq i < |a| \rightarrow_L (\forall j : \mathbb{Z})\ (0 \leq j < |a[i]| \rightarrow_L examenSinResponder(a[i][j])))$

}

pred examenSinResponder($e : Examen$) {

$(|e| = 1 \wedge (\forall i : \mathbb{Z})\ (0 \leq i < |e[0]| \wedge_L e[0][i] =$ ""))

}
proc igualdad(in $edr1, edr2 : EdR, $) : Bool {
  requiere { $True$ }
  asegura {
    $(res = True) \leftrightarrow$
    $(edr1.aula = edr2.aula) \wedge$
    $(edr1.solucion = edr2.solucion) \wedge$
    $(edr1.entregas = edr2.entregas)$
  }
}
proc copiarse(in $c1, c2 : Coordenada$, inout $edr : EdR$) : {
  requiere {
    $perteneceAlumno(c1, edr.aula) \wedge_L$
    $perteneceAlumno(c2, edr.aula) \wedge_L$
    $edr = edr0 \wedge_L$
    $adyacentes(c1, c2, edr.aula) \wedge_L$
    $(\exists k : \mathbb{Z})(ultimoPasoExamen(c1.edr.aula) = "" \wedge ultimoPasoExamen(c2.edr.aula) \neq "")$
  }
  asegura {
    $|edr.aula[c1.f][c1.c]| = |edr0.aula[c1.f][c1.c]| + 1 \wedge_L$
    $(\exists j : \mathbb{Z})(ultimoPasoExamen(c1, edr0)[j] = "" \wedge ultimoPasoExamen(c2, edr0)[j] \neq "" \wedge ultimoPasoExamen(c1, edr) = setAt(ultimoPasoExamen(c1, edr0), j, ultimoPasoExamen(c2, edr0)[j]))$
    $(\forall c3 : Coordenada)\ ((cordenadaValida(c3, edr0.Aula) \wedge c1.f \neq c3.f \vee c1.c \neq c3.c) \rightarrow_L edr.Aula[c3.f][c3.c] = edr0.Aula[c3.f][c3.c]) \wedge_L$
    $edr0.solucion = edr.solucion \wedge_L$
    $edr0.entregas = edr.entregas$
  }
}
pred perteneceAlumno($coord : Coordenada, a : Aula$) {
  $(cordenadaValida(coord, edr.aula) \wedge_L$
  $|a[i][j]| > 0)$
}
pred cordenadaValida($coord : Cordenada, a : Aula$) { $0 \leq coord.f < |a| \wedge 0 \leq coord.c < |a|$ }

pred adyacentes($coord1, coord2 : Cordenada, a : Aula$) {
  $cordenadaValida(coord1, a) \wedge cordenadaValida(coord2, a) \wedge$
  $((|c1.c - c0.c| = 2 \wedge c1.f = c0.f) \vee$
  $(c1.c = c0.c \wedge c1.f - c0.f = 1))$
}
aux ultimoPasoExamen($c1 : Coordenada, edr : EdR$) : $Examen = edr.aula[c1.f][c1.c][|edr.aula[c1.f][c1.c]| - 1]$

proc consultarDarkWeb(in $s : Paso$, in $posiblesAccesos : \mathbb{Z}$, inout $edr : EdR$) {
  requiere {
    $rtaValida(s) \wedge_L$
    $posiblesAccesos \geq 0 \wedge_L$

```
    }
    asegura { res }
  }
  proc resolver(in alumno : Cordenada, inout edr : EdR) : Paso {
    requiere {
      (existeAlumno(alumno)) ∧_L
      (∃i : ℤ)(0 ≤ i < |alumno.examen[0]|∧_L alumno.examen[|alumno.examen|][k] ==
      "")
    }
    asegura { res }
  }
  proc entregar(in alumno : Alumno, inout edr : EdR) : {
    requiere {
      perteneceAlumno(alumno.Coordenada, edr.Aula) ∧_L
      edr = edr0
    }
    asegura {
      edr.entregas = edr0.entregas++⟨alumno.coordenada, alumno.examen[|alumno.examen|−
      1]⟩ ∧
      edr.aula[alumno.coord.f][alumno.coord.c] = ⟨⟩ ∧
      (∀c1 : Coordenada) (coordenadaValida(c1) ∧_L c1 ≠ alumno.coordenada →_L
      edr.aula[c1.f][c1.c] = edr0.aula[c1.f][c1.c]) ∧_L
      edr.solucion = edr0.solucion
    }
  }
  proc chequearCopias(in alumnos : seq < Alumno >) : seq < Alumno > {
    requiere { True }
    asegura { res }
  }
  proc corregir(inout edr, EdR) : seq << Alumno, Nota >> {
    requiere { True }
    asegura { res }
  }
}
```