Algunos comandos útiles para la resolución del parcial

Vectores (`vector`):

- `vector <int> v` crea un vector
- `v.size()` obtener el tamaño del vector
- `v.push_back(a)` insertar un elemento al final del vector
- `vector <int> v2 = v` crea una copia completa del vector

Pares (`pair`):

- `p = make_pair(a, b)` crea un par
- `p.first` accede al primer elemento
- `p.second` accede al segundo elemento

Enunciado

En Argentina, el censo de personas es realizado por el INDEC (para más detalles revisar el TP). Sin embargo, las personas no son los únicos seres vivos censados. Se hacen también estudios sobre distintas especies de flora y fauna. Debido a hechos que se hicieron de público conocimiento en los últimos meses, se creó recientemente un centro para estudiar las poblaciones de carpinchos. C.U.A.L.Ca (Centro Único de Avistaje de Lindos CArpinchos) relevó información sobre el hábitat de dichos roedores en una región de Buenos Aires.

Dado que las inundaciones afectan mucho la vida de estos animales, por cada parcela relevada se midió su elevación con respecto al nivel del mar y la cantidad de carpinchos que viven en ese terreno, y con esa información se construyó un mapa como el que se ve a continuación:

$(20, 2)$	$(22, 4)$	$(21, 8)$	$(20, 3)$
$(23, 2)$	$(25, 6)$	$(20, 12)$	$(20, 4)$

(a) Un mapa de zonas de aprox 20 metros sobre el nivel del mar y una población total de 41 carpinchos

Representaremos el mapa como una grilla, de tipo $seq\langle seq\langle \mathsf{zona}\rangle\rangle$, donde cada celda modela una zona. La zona es un par, de tipo altura $\times$ población.

Los mapas serán de tamaño $N \times M$ siendo $N > 1$ y $M > 1$.

Dados los siguientes renombres de tipos:

type posición $= \mathbb{Z} \times \mathbb{Z}$

type altura $= \mathbb{Z}$

type población $= \mathbb{Z}$

type zona $=$ altura $\times$ población

type mapa $= seq\langle seq\langle \mathsf{zona}\rangle\rangle$

Implementar las funciones enumeradas a continuación respetando la especificación.

Nota: Las funciones implementadas pueden ser reutilizadas en otros ejercicios. Para ello copiarlas de un ejercicio a otro.

Ejercicio 1.

```
bool esValle(mapa m, posicion p)
```

Dado un mapa y una posición, devuelve si en dicha posición hay un valle.

```
proc esValle (in m: mapa, in p: posición, out res: Bool) {
    Pre {mapaValido(m) ∧_L posicionEnRango(m, p)}
    Post {res = True ↔ esValle(m, p)}

    pred mapaValido (m: mapa) {
        esMatriz(m) ∧_L posicionesValidas(m)
    }
    pred esMatriz (m: mapa) {
        |m| > 1 ∧_L (|m[0]| > 1 ∧ (∀f : Z)(0 < f < |m| −→_L |m[f]| = |m[f − 1]|))
    }
    pred posicionesValidas (m: mapa) {
```

$(\forall p : \text{posición})(posicionEnRango(m, p) \longrightarrow_L poblacionValida(m, p))$
}
pred poblacionValida (m: mapa, p: posición) {
$(poblacionPosicion(m, p) \geq 0$
}
pred posicionEnRango (m: mapa, p: posición) {
$0 \leq p_0 < |m| \wedge 0 \leq p_1 < |m[0]|$
}
aux alturaPosicion (m: mapa, p: posición) : $\mathbb{Z} = m[p_0][p_1]_0$;
aux poblacionPosicion (m: mapa, p: posición) : $\mathbb{Z} = m[p_0][p_1]_1$;
pred esValle (m: mapa, p: posición) {
$(\forall q : \text{posición})(esVecino(m, p, q) \longrightarrow_L alturaPosicion(m, p) \leq alturaPosicion(m, q))$
}
pred esVecino (m: mapa, p: posición, q: posición) {
$posicionEnRango(m, q) \wedge$
$((p_0 = q_0 + 1 \wedge p_1 = q_1) \vee$
$(p_0 = q_0 - 1 \wedge p_1 = q_1) \vee$
$(p_0 = q_0 \wedge p_1 = q_1 + 1) \vee$
$(p_0 = q_0 \wedge p_1 = q_1 - 1))$
}
}

Ejemplo de input (primeras líneas corresponden a la posición, luego la dimensión del mapa y finalmente los datos del mapa):

```
0
0
2
2
(10,100) (40,100)
(30,100) (500,100)
```

Output esperado:

```
True
```

Ejercicio 2.

```
vector<posicion> bajaNatalidad(mapa m1, mapa m2)
```

Dados dos mapas que corresponden a las mismas zonas pero en distintos momentos del tiempo, determinar en que posiciones hubo menor crecimiento de la población de carpinchos. La lista debe estar ordenada de acuerdo a la cantidad de carpinchos en el mapa m2. **No está permitido usar la función sort()**

proc bajaNatalidad (in m1: mapa, in m2: mapa, out res: $seq\langle$posición$\rangle$) {
Pre $\{(mapaValido(m1) \wedge mapaValido(m2)) \wedge_L mismasAlturas(m1, m2)\}$
Post $\{(\forall q : \text{posición})(q \in res \Leftrightarrow (posicionEnRango(m1, q) \wedge_L menorCrecimientoPoblacional(m1, m2, q))) \wedge_L ordenadaPorCantCarpinchos(res, m2)\}$

pred mismasAlturas (m1: mapa, m2: mapa) {
$mismasDimensiones(m1, m2) \wedge_L$
$(\forall p : \text{posición})(posicionEnRango(m1, p) \longrightarrow_L alturaPosicion(m1, p) = alturaPosicion(m2, p))$
}
pred menorCrecimientoPoblacional (m1: mapa, m2: mapa, p: posición) {
$(\forall q : \text{posición})(posicionEnRango(m1, q) \longrightarrow_L$
$crecimientoPoblacional(m1, m2, q) \geq crecimientoPoblacional(m1, m2, p)$
}
aux crecimientoPoblacional (m1: mapa, m2: mapa, p: posición) : $\mathbb{Z} =$
$poblacionPosicion(m2, p) - poblacionPosicion(m1, p)$;
pred ordenadaPorCantCarpinchos (l: $seq\langle res$:posición$\rangle$, m: mapa) {
$(\forall i : \mathbb{Z})(0 < i < |l| \longrightarrow_L poblacionPosicion(m, l[i-1]) \leq poblacionPosicion(m, l[i]))$
}
}

Ejemplo de input:

```
2
2
(10,100) (40,100)
(30,100) (500,100)
2
2
(10,105) (40,120)
(30,150) (500,160)
```

Output esperado:

```
(0,0)
```

Ejercicio 3.

```
void vivenEnAltura(mapa m, int &alt, int &cantidad)
```

Dado un mapa queremos saber la altura de las zonas más altas, y cuantos carpinchos viven a esa altitud.

proc vivenEnAltura (in m: mapa, out altura: $\mathbb{Z}$, out cantidad: $\mathbb{Z}$) {
Pre $\{mapaValido(m)\}$
Post $\{(\exists p : \text{posición})(posicionEnRango(m,p) \wedge_L (m[p_0][p_1]_0 = altura \wedge mayorAlturaDelMapa(m, altura) \wedge cantidad = poblacionAesaAltura(m, altura)))\}$

aux poblacionAesaAltura (m: mapa, h: altura) : $\mathbb{Z}$ = $\sum_{f=0}^{|m|-1} \sum_{c=0}^{|m[0]|-1}$ if $m[f][c]_0 = altura$ then $m[f][c]_1$ else 0 fi ;
pred mayorAlturaDelMapa (m: mapa, h:altura) {
$(\forall p : \text{posición})(posicionEnRango(m,p) \longrightarrow_L h \geq m[p_0][p_1]_0)$
}
}

Ejemplo de input:

```
2
2
(10,105) (40,120)
(30,150) (500,160)
```

Output esperado (altura y cantidad de carpinchos) :

```
500 160
```

Ejercicio 4.

```
int rellenarValles(mapa &m)
```

Se desea simular como quedaría un terreno en caso de elevar las zonas más bajas para evitar inundaciones. Es decir, rellenar los valles. Además se debe devolver la cantidad de metros que se elevó el terreno.

proc rellenarValles (inout m: mapa, out res: $\mathbb{Z}$) {
Pre $\{m = m_0 \wedge mapaValido(m_0)\}$
Post $\{res = diferenciaAlturas(m, m_0) \wedge mantienePoblacion(m, m_0) \wedge_L (\forall p : \text{posición})(posicionEnRango(m,p) \longrightarrow_L vallesRellenos(m_0, m, p))\}$

aux diferenciaAlturas (m: mapa, m0: mapa) : $\mathbb{Z}$ = $\sum_{f=0}^{|m|-1} \sum_{c=0}^{|m[0]|-1} alturaPosicion(m,(f,c)) - \sum_{f=0}^{|m0|-1} \sum_{c=0}^{|m0[0]|-1} alturaPosicion(m0,(f,c))$;

pred mantienePoblacion (m1: mapa, m2: mapa) {
$mismasDimensiones(m1, m2) \wedge_L (\forall p : \text{posición})(posicionEnRango(m1,p) \longrightarrow_L poblacionPosicion(m1,p) = poblacionPosicion(m2,p))$
}
pred vallesRellenos (m1: mapa, m2: mapa, p: posición) {
$(esValle(m1,p) \longrightarrow esPromedioVecinos(m1,m2,p)) \wedge$
$(\neg esValle(m1,p) \longrightarrow alturaPosicion(m1,p) = alturaPosicion(m2,p))$
}
pred esPromedioVecinos (m1: mapa, m2: mapa, p: posición) {

$(\exists v : seq\langle \text{posición} \rangle) esListaVecinos(v, m1, p) \wedge alturaPosicion(m2, p) = (\sum_{i=0}^{|v|-1} alturaPosicion(m1, v[i]))\ div\ |v|$

}

`pred esListaVecinos` (v: $seq\langle$posición$\rangle$, m: mapa, p: posición) {

$(\forall pos : \text{posición}) pos \in v \Leftrightarrow esVecino(m, p)$

}

}

Hint: Recuerde que m_0 representa el estado inicial del mapa mientras que m el valor que modificará, por lo que se sugiere no trabajar unicamente con m.

Ejemplo de input:

```
2
2
(10,105) (40,120)
(30,150) (500,160)
```

Output esperado:

```
25
2
2
(35,105) (40,120)
(30,150) (500,160)
```