Nombre y Apellido	LU	NO
Lucio Máspero	746/17	86

Corrector:	Luis		
Nota Final / Ejs:	1	2	3
D	A--	I	R

Algoritmos y Estructuras de Datos II
Primer Recuperatorio – 27 de Junio de 2022

Aclaraciones

- El parcial es a libro casi-cerrado. Solo es posible tener impreso el apunte de TADs básicos. Además, pueden tener una hoja (2 carillas), escrita a mano, con los apuntes que se deseen.
- Cada ejercicio debe entregarse **en hojas separadas**. Las mismas deben estar numeradas.
- Incluir en esta hoja: nombre, apellido, el número de orden asignado, número de libreta.
- Incluir en cada hoja entregada: nombre, apellido y número de libreta.
- Cada ejercicio se calificará con Perfecto, Aprobado, Regular, o Insuficiente.
- El parcial estará aprobado si las notas de los tres ejercicios son mejores o iguales a alguna de las siguientes combinaciones: {R, R, A}, {R, A, R} o {I, A, A}.

Ej. 1.

Axiomatizar la función **clavesMayoresA** que, dado un diccionario a naturales y un natural, devuelve el conjunto de claves que tienen un significado mayor o igual a el natural parámetro.

Esto es:

$$\text{clavesMayoresA} : \text{dicc}(\alpha \times \text{nat}) \times \text{nat} \longrightarrow \text{conj}(\alpha)$$

Que cumple que:

$$(\forall\, a : \alpha,\ d : \text{dicc}(\alpha, \text{nat}),\ n: \text{nat})\ a \in \text{clavesMayoresA}(d, n) \iff (a \in \text{claves}(d) \wedge_L \text{obtener}(d, a) \geq n)$$

Ej. 2.

Volvemos a modelar otra competencia entre calamares: "Calamar 43". El juego consiste de una carrera en turnos. En un principio, un conjunto de calamares se anota en la carrera. Los calamares participantes empiezan en la posición 0. La carrera tiene una posición de llegada que se define al inicio. En cada turno, cada calamar intentará avanzar una cantidad positiva de posiciones. A su vez, en cada turno, un árbitro decidirá la cantidad de posiciones máxima que se puede avanzar, la cual es desconocida para los calamares. Al finalizar el turno, todos los calamares que intentaron avanzar más posiciones que las decididas por el árbitro quedan descalificados. La carrera continúa hasta que algún calamar alcance la posición final o quede un único calamar compitiendo. Nos piden conocer, en todo momento, quienes son los calamares que llevan la delantera.

Dada la definición del problema presentado, definir un TAD CALAMAR43 a partir de los siguientes pasos:

(a) Listar en lenguaje natural qué aspectos son relevantes para diferenciar dos Calamar43.

(b) Expresar esto en un conjunto de observadores y una definición de igualdad observacional. Comentar qué parte del enunciado expresa cada observador.

(c) Definir un conjunto de generadores que permitan generar todos los valores relevantes del TAD. Comentar qué parte del enunciado expresa cada generador.

(d) Axiomatizar el/los observadore/s. Para ello, tienen disponible la función **clavesMayoresA** del Ejericio 1.

Ej. 3.

A continuación debemos modelar un Subte Circular con protocolo invernal. Un Subte Circular se caracteriza por tener una serie de paradas que conforman un círculo, de forma que el subte siempre puede moverse hacia adelante. Por otra parte, el protocolo invernal indica que el mismo tiene una capacidad máxima. En consecuencia, los pasajeros ingresan de a uno al tren e indican en qué parada se bajan. En caso de llegarse a la capacidad máxima, el tren no permitirá subir más gente e inmediatamente continuará circulando hasta llegar a una parada donde se baje algún pasajero.

Completar los axiomas del TAD SUBTE CIRCULAR (los indicados con puntos suspensivos) presentado a continuación. Pueden utilizarse las funciones auxiliares ya axiomatizadas.

TAD PARADA es string

TAD SUBTE CIRCULAR

géneros subte

observadores básicos

paradas	: subte	$\longrightarrow$	secu(parada)	
bajanEn	: subte sc × parada p	$\longrightarrow$	nat	{está?(p, paradas(sc))}
maxPasajeros	: subte	$\longrightarrow$	nat	

generadores

inaugurarSC	: secu(parada) ps × nat m	$\longrightarrow$	subte	{longitud(ps) $\geq 2 \wedge$ m ≥ 1}
abordar	: subte sc × parada p	$\longrightarrow$	subte	{está?(p, paradas(sc)) $\wedge$ p $\neq$ paradaActual(sc)}
avanzarParada	: subte sc	$\longrightarrow$	subte	

otras operaciones

rotarParada	: secu(parada)	$\longrightarrow$	secu(parada)
paradaActual	: subte	$\longrightarrow$	parada
cantPasajeros	: secu(parada) ps × subte sc	$\longrightarrow$	nat

axiomas (extracto)

$\forall sc$: subte, $\forall ps$: secu(parada) $\forall p, p'$: parada, $\forall m$: nat

paradas(inaugurarSC(ps, m))	$\equiv$	ps
paradas(abordar(sc, p))	$\equiv$	...
paradas(avanzarParada(sc))	$\equiv$	...
bajanEn(inaugurarSC(ps, m), p')	$\equiv$	...
bajanEn(abordar(sc, p), p')	$\equiv$	...
bajanEn(avanzarParada(sc), p')	$\equiv$	...
maxPasajeros(inaugurarSC(ps), m)	$\equiv$	m
rotarParada(ps)	$\equiv$	fin(ps) $\circ$ prim(ps)
paradaActual(sc)	$\equiv$	prim(paradas(sc))
cantPasajeros(ps, sc)	$\equiv$	**if** vacia?(ps) **then** 0 **else** bajanEn(sc, prim(ps)) + cantPasajeros(fin(ps), sc) **fi**

Fin TAD

Como se puede ver en el TAD, el Subte Circular cuenta con las siguientes funciones:

- **paradas** indica las paradas del subte en el orden que se recorren. La primera parada de la secuencia siempre es la actual.
- **bajanEn** devuelve cuantos pasajeros están esperando para bajar en una parada.
- **maxPasajeros** devuelve la cantidad máxima admitida de pasajeros en el tren.
- **inagurarSC** inicia un nuevo Subte Circular con sus paradas y cantidad máxima de pasajeros.
- **abordar** permite a un pasajero suba al tren, indicando la parada en la que desciende. Notar que no es necesario restringir que el tren no está lleno al ascender, ya que, si el tren se llena, este debe avanzar hasta que se baje alguien.
- **avanzarParada** permite al subte avanzar a la siguiente parada, bajando todos los pasajeros destinados a la misma.