

Enunciado-Examen-Segunda-Convoca...



Juandf03



Estructuras de Datos



2º Grado en Ingeniería Informática



Escuela Técnica Superior de Ingeniería Informática
Universidad de Málaga



MÁSTER EN

Inteligencia Artificial & Data Management

MADRID

Formamos
talento para un futuro
Sostenible

saber más



¿Listo para aprobar tus oposiciones?

Academia fernauro, formación a tu medida



¡Elige el curso
que mejor se
adapte a ti!

[Ver más](#)

Grupos
reducidos

Presencial y
Online

Cursos
Intensivos

Preparación
continua

Enunciado Examen Segunda Convocatoria Febrero 2024

(0.25 puntos) Define la función `empty` que devuelve un `SetChain` vacío.

```
empty :: SetChain a
```

(0.25 puntos) Define la función `isEmpty` que dado un `SetChain`, devuelve `True` si está vacío o `False` en caso contrario.

```
isEmpty :: SetChain a -> Bool
```

(1.5 puntos) Define la función `getEpoch` que dada una transacción y un `SetChain`, devuelve la época en la que se validó dicha transacción o `-1` si no es una transacción validada.

```
getEpoch :: (Eq a) => a -> SetChain a -> Integer
```

(0.75 puntos) Define la función `size` que dado un `SetChain`, devuelve el número de transacciones validadas que contiene.

```
size :: SetChain a -> Int
```

(0.25 puntos) Define la función `pendingTransactions` que dado un `SetChain`, devuelve `True` si hay transacciones pendientes de validación.

```
pendingTransactions :: SetChain a -> Bool
```

(0.5 puntos) Define la función `add` que dados una transacción y un `SetChain`, devuelve un nuevo `SetChain` como el proporcionado pero que, además, incluye la transacción dada como no validada.

```
add :: (Eq a) => a -> SetChain a -> SetChain a
```

(1.5 puntos) Define la función `validate` que, dado un `SetChain`, valida todas las transacciones no validadas. Este proceso incluye los siguientes casos: 1) Si no hay transacciones no validadas, el `SetChain` no se modifica. 2) Si alguna de las transacciones de `mempool` está en algún bloque de `history`, entonces se para el proceso de validación y se lanza un error con mensaje "transaction already validated". 3) En otro caso, hay que añadir a `history` una

WUOLAH

nueva asociación entre el nuevo bloque de transacciones validadas y la época en la que se ha realizado la validación. Además, hay que incrementar el valor de epochNum en 1 y dejar la mempool vacía.

```
validate :: (Eq a) => SetChain a -> SetChain a
```

(0.5 puntos) Define la función addAll que, dados una lista de transacciones y un SetChain, añade todas las transacciones de la lista al SetChain sin validar.

```
addAll :: (Eq a) => [a] -> SetChain a -> SetChain a
```

(1.25 puntos) Define la función de plegado fold que, dados una función, un valor inicial y un SetChain, pliega, comenzando por el valor inicial y usando la función f de derecha a izquierda, las transacciones validadas de un SetChain.

```
fold :: (a -> b -> b) -> b -> SetChain a -> b
```

(0.25 puntos) Define la función toList que, dado un SetChain, devuelve una lista que contiene todas las transacciones validadas. La implementación tiene que usar la función de plegado anterior.

```
toList :: SetChain a -> [a]
```