

Solucion-febrero-2012-haskell.pdf



angelgg0700



Estructuras de Datos



2º Grado en Ingeniería Informática



Escuela Técnica Superior de Ingeniería Informática
Universidad de Málaga

BBVA**1/6**

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

BBVA está adherido al Fondo de Garantía de Depósitos de Entidades de Crédito de España. La cantidad máxima garantizada es de 100.000 euros por la totalidad de los depósitos constituidos en BBVA por persona.

Ábrete la Cuenta Online de BBVA y llévate 1 año de **Wuolah PRO**

Ventajas Cuenta Online de BBVA

0€

Sin comisión de administración o mantenimiento de cuenta. (0 % TIN 0 % TAE)

0€

Sin comisión por emisión y mantenimiento de Tarjeta Aqua débito.

0

Sin necesidad de domiciliar nómina o recibos.

Las ventajas de **WUOLAH PRO**



Di adiós a la publi en los apuntes y en la web



Descarga carpetas completas de un tirón



Acumula tickets para los sorteos

cómo??





1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

BBVA está adherido al Fondo de Garantía de Depósitos de Entidades de Crédito de España. La cantidad máxima garantizada es de 100.000 euros por la totalidad de los depósitos constituidos en BBVA por persona.

ventajas

PRO



Di adiós a la publi en los apuntes y en la web



Acumula tickets para los sorteos



Descarga carpetas completas

estudia sin publi
WUOLAH PRO

```
module BiPartite(
  biColored -- :: Ord v => Graph v -> Maybe (D.Dictionary v Color)
  , Color(..)
) where
```

```
import Graph
import Data.Maybe(isJust)
import qualified Dictionary as D
import qualified Stack as S
```

```
data Color = Red | Blue deriving (Eq,Show,Ord)
nextColor Red = Blue
nextColor Blue = Red
```

```
pushAll :: S.Stack a -> [a] -> S.Stack a
pushAll = foldr S.push
```

```
biColored :: Ord v => Graph v -> Maybe (D.Dictionary v Color)
biColored g
| null vs = Just D.empty -- empty graph is bipartite
| otherwise = aux g D.empty (S.push (src ,Red) S.empty)
where
  vs = vertices g
  src = head vs -- initial vertex
```

```
aux :: Ord v => Graph v -> D.Dictionary v Color -> S.Stack (v, Color) ->
  Maybe (D.Dictionary v Color)
```

```
aux g dict stack
| S.isEmpty stack = Just dict
| not(colored v) = aux g dict' (pushAll stack' newsuc)
| c == c' = aux g dict stack'
| otherwise = error "no bipartito"
```

```
--
-- ¡¡¡ completad el resto de guardas !!
--
```

```
where
  colored v = D.isDefinedAt v dict
  stack' = S.pop stack
  (v, c) = S.top stack
  dict' = D.insert v c dict
  c' = D.valueOf v dict
  newsuc = [(v,nextColor c)| v <- successors g v, not(colored v)]
```

```
--
-- ¡¡¡ completad las variables locales necesarias !!!
--
```

```
-----
--- EXAMPLES -----
-----
```

Llévate 1 año de WUOLAH PRO con BBVA. ¿Cómo? ¡+Info aquí!

WUOLAH