

Tarea 1: Haaave you met Haskell?

Información del curso

Programación Declarativa - Facultad de Ciencias, UNAM.

- Profesor: Favio E. Miranda Perea
- Ayudante Javier Enríquez Mendoza

Descripción de la práctica

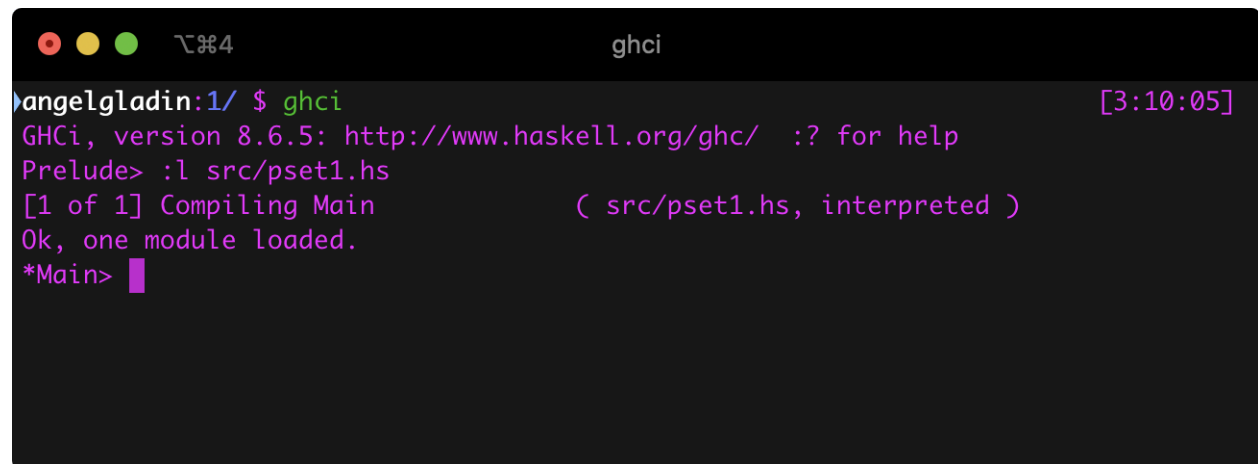
En el archivo de especificación de la práctica viene todo explicado `doc.pdf`.

Entorno

- `ghci`: The Glorious Glasgow Haskell Compilation System, version 8.6.5

Ejecución

Estando ubicado con la terminal en el directorio de la práctica, ejecutar `ghci` y después `:l src/pset1.hs` como se muestra enseguida.



```
angelgladin:1/ $ ghci
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
Prelude> :l src/pset1.hs
[1 of 1] Compiling Main                ( src/pset1.hs, interpreted )
Ok, one module loaded.
*Main>
```

Strings

```
angelgladin:1/ $ ghci [3:17:31]
GHCi, version 8.6.5: http://www.haskell.org/ghc/ :? for help
Prelude> :l src/pset1.hs
[1 of 1] Compiling Main          ( src/pset1.hs, interpreted )
Ok, one module loaded.
*Main> quitaMayusculas "I <3 Haskell"
" <3 askell"
*Main> soloLetras "Oppan Lambda Style!!"
"OppanLambdaStyle"
*Main> prefijo "Haskell" "Que Chido es Haskell"
False
*Main> prefijo "Que Chido" "Que Chido es Haskell"
True
*Main> 
```

Merge Sort

```
ghci

*Main> l1 = [1,2,3,4,5]
*Main> l2 = [9,8,7,6,5,4,3,2,1]
*Main> l3 = [1,65,3,78,9,5]
*Main> mergeSort l1
[1,2,3,4,5]
*Main> mergeSort l2
[1,2,3,4,5,6,7,8,9]
*Main> mergeSort l3
[1,3,5,9,65,78]
*Main> mergeSortCon compare l1
[1,2,3,4,5]
*Main> mergeSortCon compare l2
[1,2,3,4,5,6,7,8,9]
*Main> mergeSortCon compare l3
[1,3,5,9,65,78]
*Main> 
```

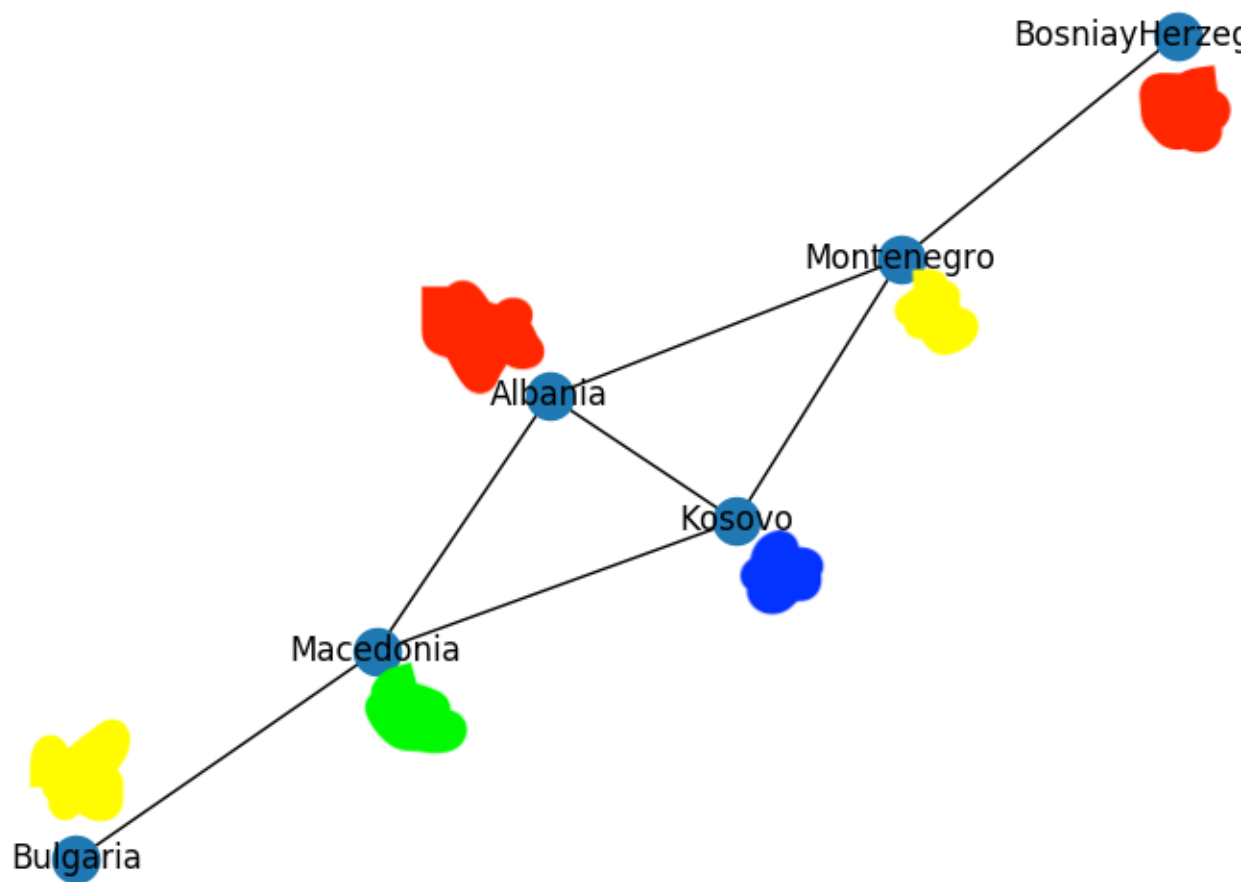
Coloración

Para la función `esBuena` primero hice la función `coloraciones` para generar todas las posibles buenas coloraciones.

Para ello lo primero que se hará es obtener una coloración arbitraria de `coloraciones`, por ejemplo tomemos la coloración indexada de esa lista número 42.

```
ghci
*Main> coloraciones adyacencias !! 42
[(Rojo,Albania),(Amarillo,Bulgaria),(Rojo,BosniayHerzegovina),(Azul,Kosovo),
(Verde,Macedonia),(Amarillo,Montenegro)]
*Main> █
```

Podemos verificar que en efecto es una coloración válida.



Por cierto, me salieron 432 coloraciones válidas con las configuraciones que se nos dieron.

Comentarios

Con la biblioteca de `networkx` y `matplotlib` en Python hice dibujé la gráfica para apoyarme visualmente para ver como se veía la gráfica y en checar rápidamente las coloraciones que obtuve, para eso hice el siguiente script.

```
import networkx as nx
import matplotlib.pyplot as plt

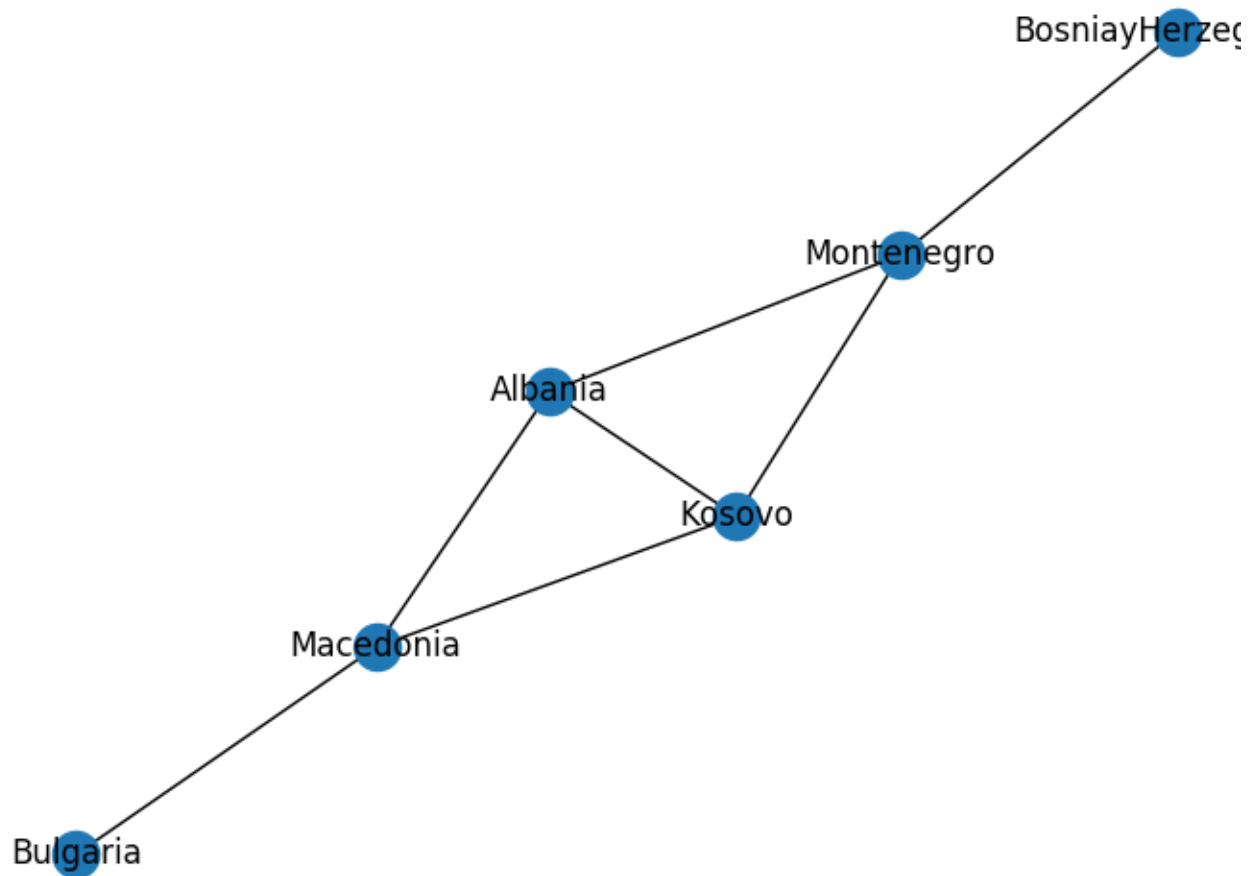
g = nx.Graph()

edges = [('Albania', 'Montenegro'), ('Albania', 'Kosovo'), ('Albania', 'Macedonia'),
        ('Bulgaria', 'Macedonia'), ('BosniayHerzegovina', 'Montenegro'),
        ('Kosovo', 'Macedonia'), ('Kosovo', 'Montenegro')]

for (a, b,) in edges:
    g.add_edge(a, b)

nx.draw(g, with_labels=True)
plt.savefig("graph.png")
```

Dando como resultado la siguiente imagen.



En la parte de `#coloraciones` en `src/pset1.hs` de la línea 101 a 113 quise hacerlo más dinámico pero ya me quería dormir e hice restringido los colores y las ciudades.

Referencias

- <https://hackage.haskell.org/package/base-4.12.0.0/docs/Data-Char.html>
- http://zvon.org/other/haskell/Outputprelude/filter_f.html
- <https://www.hackerearth.com/practice/algorithms/sorting/merge-sort/tutorial/>
- http://zvon.org/other/haskell/Outputsyntax/caseQexpressions_reference.html
- <https://hackage.haskell.org/package/base-4.12.0.0/docs/src/Data.List.NonEmpty.html#splitAt>
- <https://hackage.haskell.org/package/base-4.12.0.0/docs/src/GHC.List.html#lookup>
- https://en.wikipedia.org/wiki/Cartesian_product

Integrante(s)

- Ángel Iván Gladín García - *angelgladin@ciencias.unam.mx*