Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2115 – Programación como Herramienta para la Ingeniería (II/2019)

Actividad de participación 2 - Estructuras de datos

Objetivos

- Consolidar los conocimientos de estructuras de datos.
- Comprender el uso estructuras de datos y sus aplicaciones en redes.

Entrega

- **Lenguaje a utilizar:** Python 3.6
- **Lugar:** repositorio privado en GitHub. Recuerde incluir todo en una carpeta de nombre **P02**.
- **Entrega:** jueves 5 de septiembre de 2019
- **Tiempo:** 80 minutos desde la publicación del enunciado en Syllabus
- **Formato de entrega:** un archivo jupyter notebook llamado (**solucionP02.ipynb**) y un archivo Python (**solucionP02.py**). ambos archivos deben contener la misma versión de la actividad y deben estar ubicados en la carpeta **P02**. No se debe subir ningún otro archivo a la carpeta. Utilice múltiples celdas de texto y código para facilitar la revisión de su trabajo.
- **Descuentos:** Las actividades que no cumplan el formato tendrán descuento de 0.5 ptos.
- **Atrasos:** La no recepción del archivo a tiempo implica nota **1.0** sin excepción.
- **Tareas con errores de sintaxis y/o que generen excepciones serán calificadas con nota 1.0.**

Ejercicio

¡El Tigre ha sido herido! El famoso héroe de Progra Muy Muy Lejana, se encuentra débil en algún punto del bosque encantado. Sus fieles escuderos Pablo y Felipe (P & F) deciden acudir en su ayuda. Para ello, cuentan con el mapa **bosque.txt** en donde cada letra representa una casilla del bosque. El formato del mapa se describe a continuación:

- O: representa una casilla vacía. Desde cada casilla vacía se puede ir a otra (también vacía) que esté inmediatamente arriba, abajo, a la izquierda o a la derecha, si es que estas existen.
- X: representa una casilla ocupada, es decir, no se puede transitar por ella.

El sistema de coordenadas del mapa se representará de la siguiente forma:

$$\begin{bmatrix} a_{00} & a_{01} & a_{02} & \dots & a_{0m} \\ a_{10} & a_{11} & a_{12} & \dots & a_{1m} \\ a_{20} & a_{21} & a_{22} & \dots & a_{2m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n0} & a_{n1} & a_{n2} & \dots & a_{nm} \end{bmatrix}$$

Para poder rescatar a El Tigre, P & F te contratan como experto en Estructuras de Datos. Pero antes es necesario familiares con estructuras de datos un poco más complejas. Para eso vamos a trabjar en conjunto creando la estructura necesaria para rescatar al Tigre.

- Vamos a crear una función **leer_mapa(path)** que lea un mapa **.txt** con el formato descrito anteriormente y que reciba como parámetro el **path** (ruta al archivo). Debe retornar una lista de las filas del mapa, en donde cada fila es un string con una secuencia de X y O. Recuerde que si el archivo **.txt** se encuentra en la misma carpeta que el **.ipynb**, basta con que el path sea el nombre del archivo **.txt**.
- Ahora vamos a crear la función **crear_grafo(lista)** que reciba como input la lista anterior y cree un grafo en donde cada nodo sea una casilla vacía. Cada nodo debe guardar información sobre su ubicación en el mapa y tener una lista de los nodos adyacentes. Se espera que el output de esta función sea un diccionario de la forma {**(i, j): nodo**} ($\forall\ (i,j) : a_{ij} = O$) en donde cada llave sea una coordenada de la forma **(i, j)** explicada anteriormente. Los valores del diccionario deben ser objetos de la clase Nodo, en donde cada uno guarde en una lista sus vecinos en el orden: (arriba, izquierda, abajo, derecha). Se adjunta una visualización del orden de prioridades.

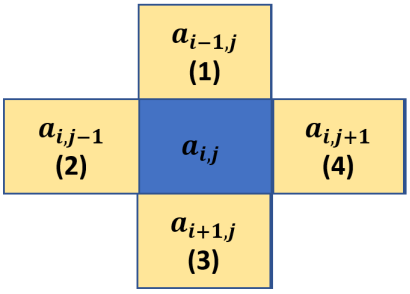

Figure 1: Orden de prioridades