Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
Laboratorio de Algoritmos y Estructuras III
Trimestre Septiembre-Diciembre 2016

Ayudante de plan de estudios

1. Descripción del problema

El objetivo de esta práctica es el de realizar un programa que ayude a los estudiantes de una carrera a formular su plan de estudios. La entrada del programa es la lista de materias del *pensum* que les faltan por ver, y las materias que son requisitos de cada una de las materias restantes. Para ello el programa debe llevar a cabo las siguientes tareas:

1. Indicar el mínimo número de períodos académicos que debe cursar el estudiante.
2. Mostrar la lista de las materias que si se retrasa su aprobación, inevitablemente aumenta el números total de períodos académicos a cursar en la carrera. A estas materias se las llamaremos materias críticas. En la lista las materias deben estar en el orden en que se deben aprobar las materias. Observe que puede haber más de una lista de materias críticas, en cuyo caso debe mostrarlas todas.
3. Para cada una de las materias debe indicar cuantos períodos puede pasar sin ser aprobado, sin que esto altere el número mínimo de períodos académicos a ser cursados. A este punto lo llamamos la holgura de las materias.

La solución a este problema debe esta contenida en un programa llamado `PlanDeEstudios.java` que se debe poder ejecutar desde la consola con el siguiente comando:

```
>java PlanDeEstudios <instancia>
```

donde `instancia` es el nombre del archivo con la entrada de los datos.

1.1. Entrada de los datos

El archivo de entrada tiene el siguiente formato. La primera línea tiene dos números separados por un espacio. El primer número m corresponde al número de materias, el segundo al número r es el de los pares materias que indican que una materia es requisito de la otra. Luego siguen m líneas, cada línea contiene a una materia. Después se presentan r par de materias separadas por un espacio, en donde la primera es requisito de la segunda.

1.2. Salida de los datos

El resultado del programa debe hacerse por la salida estándar, y el formato del mismo es como sigue. La primera línea se indica el número mínimo de períodos académicos que debe cursar el estudiante. La siguientes líneas muestran las listas de materias críticas. Cada lista se muestra en una sola línea, en donde cada una de las materias de la lista se imprimen separadas con un espacio. Finalmente se imprimen m líneas en donde para cada materia, se muestra la materia y su holgura, separadas por un espacio en blanco.

1.3. Ejemplos prácticos

1.3.1. Ejemplo 1

Considere un archivo llamado `caso1.txt`, con el siguiente contenido:

```
6 6
EstGeneral
Pasantia
BaseDeDatos
Redes
Alg3
Lenguajes
Alg3 BaseDeDatos
BaseDeDatos Lenguajes
Lenguajes Pasantia
Alg3 Redes
Redes Pasantia
EstGeneral Pasantia
```

Si se ejecuta el siguiente comando:

```
>java PlanDeEstudios caso1.txt
```

Se debe obtener por la salida estándar un resultado equivalente al siguiente:

```
4
Alg3 BaseDeDatos Lenguajes Pasantia
EstGeneral 2
Alg3 0
BaseDeDatos 0
Redes 1
Lenguajes 0
Pasantia 0
```

1.3.2. Ejemplo 2

Sea `caso2.txt` un archivo con el siguiente contenido:

```
22 23
Fisica2
Matematica4
Ingles
Logica
Discretas2
Matematica5
CalculoNumerico
Probabilidad
Estadistica
Alg1
Alg2
Prog.Dinamica
Org.DelComputador
Alg3
Prog.Funcional
I.A.
LenguajesProg.
Robotica
Traductores
SistemasDeOperacion
Redes
Interfaces
Fisica2 Estadistica
Matematica5 CalculoNumerico
Matematica5 Probabilidad
CalculoNumerico Estadistica
Probabilidad Estadistica
Matematica4 Discretas2
Matematica4 Logica
Matematica4 Alg1
Logica Alg2
Alg1 Alg2
Alg1 Prog.Dinamica
Alg1 Org.DelComputador
Discretas2 Org.DelComputador
Discretas2 Interfaces
Alg2 Alg3
Prog.Dinamica Prog.Funcional
Org.DelComputador Alg3
```

```
Org.DelComputador SistemasDeOperacion
Alg3 I.A.
I.A. Robotica
Alg3 LenguajesProg.
LenguajesProg. Traductores
SistemasDeOperacion Redes
```

Si se ejecuta en la consola de comandos:

```
>java PlanDeEstudios caso2.txt
```

El resultado obtenido debe ser similar al que sigue:

```
6
Matematica4 Logica Alg2 Alg3 LenguajesProg. Traductores
Matematica4 Discretas2 Org.DelComputador Alg3 LenguajesProg. Traductores
Matematica4 Logica Alg2 Alg3 I.A. Robotica
Matematica4 Alg1 Alg2 Alg3 I.A. Robotica
Matematica4 Alg1 Org.DelComputador Alg3 I.A. Robotica
Matematica4 Alg1 Org.DelComputador Alg3 LenguajesProg. Traductores
Matematica4 Discretas2 Org.DelComputador Alg3 I.A. Robotica
Matematica4 Alg1 Alg2 Alg3 LenguajesProg. Traductores
Fisica2 4
Matematica4 0
Ingles 5
Logica 0
Discretas2 0
Matematica5 3
CalculoNumerico 3
Probabilidad 3
Estadistica 3
Alg1 0
Alg2 0
Prog.Dinamica 2
Org.DelComputador 0
Alg3 0
Prog.Funcional 2
I.A. 0
LenguajesProg. 0
Robotica 0
Traductores 0
SistemasDeOperacion 1
Redes 1
Interfaces 3
```

2. Sobre la implementación

Para representar el grafo a trabajar debe hacer uso del archivo `EdgeWeightedDigraph.java`. La clase `EdgeWeightedDigraph` debe ser usada en todos sus algoritmos sobre grafos. Puede implementar una tabla en donde cada una de las materias le corresponda a número entero. Al finalizar la ejecución de sus algoritmos, puede utilizar la tabla para crear la salida del programa usando las materias de la entrada. Para resolver el problema se quiere que utilice los algoritmos que se indican en la sección V.5.4 *Planificación de Proyectos* del libro de Meza y Ortega [1], con las modificaciones que sean necesarias. En específico se quiere que utilice el *algoritmo de Bellman* (sección V.2) y el *algoritmo para el cálculo de TAC* (sección V.5.4).

3. Condiciones de entrega

Debe entregar en la página del curso en el aula virtual, el día 23 de Noviembre de 2016 antes de las 4:30 pm, un archivo comprimido llamado **LabSem11-X-Y.tar.gz** con todos los códigos en Java y el archivo Makefile correspondiente. Las letras **X** y **Y** del archivo comprimido son los número de carné de los integrantes del equipo.

Referencias

[1] Ortega, M., and Meza, O. *Grafos y Algoritmos*. Equinoccio, 2006.