ALGORITMOS Y ESTRUCTURAS DE DATOS III - 1$^{\text{er}}$ Recuperatorio
Fecha examen: 14-JUL-2017 / Fecha notas: 19-JUL-2017

	Nº Orden	Apellido y nombre	L.U.	Cant. hojas[1]
Completar:				
	Nota (Nº)	Nota (Letras)	Docente	
No completar:				

1. Decidir si las siguientes afirmaciones son verdaderas o no. En caso afirmativo describir un grafo H que cumpla lo indicado y justificar; en caso negativo demostrar.

   (a) Existe un grafo H tal que para todo grafo G, H tiene un subgrafo isomorfo a G. 0.5 p.

   (b) Existe un grafo H tal que para todo grafo G de $n \leq 2718$ vértices, H tiene un subgrafo isomorfo a G. 0.5 p.

   (c) Existe un grafo H tal que para todo grafo G, G tiene un subgrafo isomorfo a H. 0.5 p.

   (d) Existe un grafo H tal que para todo grafo G de $n \leq 314159$ vértices, G tiene un subgrafo isomorfo a H. 0.5 p.

2. Dada $M \in \mathbb{R}^{m \times n}$, junto con $i_1, i_2, j_1, j_2 \in \mathbb{Z}$ tales que $1 \leq i_1 \leq i_2 \leq m$ y $1 \leq j_1 \leq j_2 \leq n$, definimos $\text{suma}(M, i_1, i_2, j_1, j_2) = \sum_{i_1 \leq i \leq i_2} \sum_{j_1 \leq j \leq j_2} M_{i,j}$. Diseñar un algoritmo que preprocese la matriz M con complejidad estrictamente mejor que $\Theta(m^2 n^2)$, de manera tal que luego pueda calcularse $\text{suma}(M, i_1, i_2, j_1, j_2)$ con complejidad estrictamente mejor que $\Theta(mn)$ para cualquier entrada válida. Mostrar que el algoritmo propuesto es correcto y determinar su complejidad (temporal y espacial). Justificar. El mejor algoritmo que conocemos tiene complejidad $O(mn)$ para el preprocesamiento y $O(1)$ para calcular cada suma, todo lo cual es necesario para obtener puntaje máximo en este ejercicio. 2 p.

3. Sea G_n el grafo de $n \geq 2$ vértices definido inductivamente de la siguiente manera: $G_2 = K_2$ y $G_n = K_1 + G^c_{n-1}$ para $n > 2$ (grafo junta).

   (a) Demostrar que G_n es conexo y G^c_n es no conexo. 0.5 p.

   (b) Demostrar que G_n tiene un único par de vértices de igual grado, y que lo mismo vale para G^c_n. 1.5 p.
   SUGERENCIA: Demostrar ambas cosas a la vez.

4. Ruth está tan especializada en su trabajo que gana fortunas con cada cosa que hace. Como contrapartida, sólo algunas empresas requieren sus servicios. Ruth conoce las empresas que podrían convocarla, así como las ciudades donde están ubicadas. Un conjunto de rutas comunican las ciudades entre sí, y cada ruta cuenta con una estación de peaje. Usualmente cada estación de peaje cobra cierto importe a cualquiera que transite por la ruta correspondiente, pero algunas rutas están tan subsidiadas que entregan dinero a los viajeros en vez de cobrarles. *Ruth cree firmemente en la redistribución de la riqueza, por lo que quiere vivir en una ciudad tal que para cada empresa todas la maneras de ir a la misma tengan el importe neto de los peajes en su contra.* 2 p.

   Hay n ciudades, p ciudades que tienen empresas, y m rutas. Cada ciudad se identifica por un entero distinto entre 1 y n. Cada ruta conecta determinado par de ciudades y se puede recorrer desde la primera hacia la segunda. Se puede ir de cualquier ciudad a cualquier otra recorriendo las rutas, pasando eventualmente por ciudades intermedias.

   Diseñar un algoritmo de complejidad $O(n^3)$ que decida si existe alguna ciudad en la que Ruth quiera vivir. La entrada del algoritmo es la cantidad n de ciudades, la cantidad p de ciudades que tienen empresas, la cantidad m de rutas, la lista de ciudades que tienen empresas, y para cada ruta su ciudad inicial, su ciudad final y el importe de su peaje (negativo cuando la estación de peaje entrega dinero en vez de cobrarlo). Mostrar que el algoritmo propuesto es correcto y determinar su complejidad. Justificar. El mejor algoritmo que conocemos tiene complejidad $O(mn)$, lo cual es necesario para obtener puntaje máximo en este ejercicio.

5. (a) Sea G un grafo conexo. Sean $T_1 \neq T_2$ dos árboles generadores de G. Demostrar que existen dos ejes e_1 y e_2 tales que e_1 está en T_1 y no en T_2, e_2 está en T_2 y no en T_1, y tanto $T_1 + e_2 - e_1$ como $T_2 + e_1 - e_2$ son árboles generadores de G. 1 p.

   (b) Sea G un grafo conexo con pesos asociados a sus ejes. Sea T_1 un árbol generador de G. Demostrar que T_1 no tiene peso mínimo si y sólo si existe un árbol generador de G que tiene peso estrictamente menor que T_1 y que difiere de T_1 en exactamente un eje. 1 p.
   SUGERENCIA: Para la ida tomar un árbol generador mínimo de G que tenga la mayor cantidad posible de ejes en común con T_1, y usar o no el primer punto.

[1] Incluyendo a esta hoja. Entregar esta hoja junto al examen.