Examen de Programación 3 y III (16/07/2015)

Instituto de Computación, Facultad de Ingeniería, UdelaR

1. Este examen dura 4 horas y contiene **2** carillas. El total de puntos es 100 y se requieren 60 para su aprobación.
2. En los enunciados llamamos C* a la extensión de C al que se agrega el operador de pasaje por referencia &, y las sentencias *new, delete,* el uso de *cout* y *cin y el tipo bool*.
3. NO se puede utilizar ningún tipo de material de consulta.
4. No se contestarán dudas durante la última media hora.

Se requiere:

- Numerar todas las hojas e incluir en cada una el nombre, la cédula de identidad, el número de hoja y el TOTAL de hojas.
- Utilizar las hojas de un solo lado y escribir con lápiz, iniciando cada ejercicio en hoja nueva.
- En la **carátula**: completar el índice indicando en qué hoja se comenzó la respuesta a cada problema/ejercicio y el total de hojas entregadas.

Parte Obligatoria (Ejercicios 1 a 4)

Esta parte es eliminatoria, para la aprobación del examen debe obtenerse un mínimo del **50% de esta parte (20 puntos)**. En caso de no llegar a dicho mínimo, **NO** se corregirán los problemas.

Ejercicio 1 (10 puntos)

1. Dados dos algoritmos que con una entrada de tamaño **n** tienen un tiempo de ejecución en $\Theta(f(n))$ en todos los casos. Para una entrada de tamaño 100, demorarían tiempos similares? Justifique su respuesta.
2. Demuestre o dé un contraejemplo:
   a) Si $f \in O(g) \Rightarrow f/h \in O(g)$
   b) Si $f \in \Theta(g) \Rightarrow f/h \in \Theta(g/h)$

(notación: $(f/g)(n) = f(n)/g(n)$; $f,g: \mathbb{N} \to \mathbb{R}^+$, $g(n)\neq 0$)
Para demostrar debe usar las definiciones. En caso de usar propiedades vistas en el curso deberá demostrarlas.

Ejercicio 2 (10 puntos)

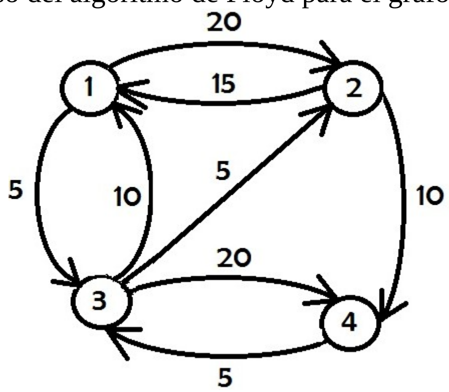
1. Defina el concepto de árbol de decisión.
2. Escriba una implementación de S**election Sort** y diagrame su árbol de decisión para **n=3**.

Ejercicio 3 (10 puntos)

1. Enunciar y demostrar la propiedad MST (Minimum Spanning Tree).
2. Dar un ejemplo de COMO se aplica dicha propiedad, en el contexto de los algoritmos greedy para hallar el árbol de cubrimiento mínimo de un grafo conexo no dirigido.

Ejercicio 4 (10 puntos)

1. Indique porqué es aplicable el principio de optimalidad al problema de hallar los costos de los caminos de mínimo costo entre todo par de vértices de un grafo que tiene costos positivos asociados a sus aristas. Escriba la recurrencia del algoritmo de Floyd para resolver el problema, definiendo completamente todas las estructuras que intervienen en la misma.
2. Explicar la ejecución paso a paso del algoritmo de Floyd para el grafo de la figura.



Problemas

Problema 1 (30 puntos)

Un camionero quiere hacer al menos C cantidad de dinero. Hay n pedidos disponibles en la empresa entre los que elegir para lograr su objetivo. A su vez cada pedido tiene una fecha de inicio, una fecha de fin y otorga una cierta cantidad de dinero. El camionero quiere ganar el dinero haciendo la menor cantidad posible de pedidos. Además no quiere que haya superposición en el tiempo entre ninguna de los pedidos que elija (si la fecha de fin de una coincide con la de inicio de otro también se considera que se superponen). Los pedidos se identifican con enteros del 1 al n.

Se plantea resolver el problema por Backtracking. Considere los siguientes datos:

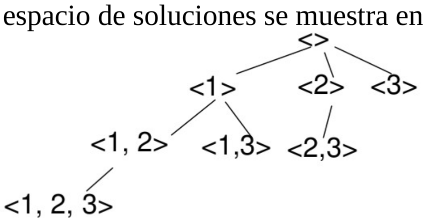
- Los datos están disponibles en el vector Pedidos, cuyo índice está entre 1 y n.
- Pedidos[i].Inicio, Pedidos[i].Fin y Pedidos[i].Dinero indican la fecha de inicio, la fecha de fin y el dinero del pedido i respectivamente.
- Los pedidos están ordenadas en el vector de manera creciente según la fecha de inicio (o sea, Pedidos[i].Inicio ≤ Pedidos[i+1].Inicio).

i. Formalizar el problema en términos de Backtracking. Describa formalmente y con lenguaje natural, la forma de la tupla solución (la forma de la tupla solución debe ser de longitud variable e indicar los pedidos elegidos de manera creciente por la fecha de inicio), las restricciones explícitas e implícitas, la función objetivo y los predicados de poda.

ii. Considere los siguientes pedidos, con una meta de C = 10:

Pedido	Inicio	Fin	Dinero
1	1	4	6
2	2	5	10
3	6	9	6

una posible organización del espacio de soluciones se muestra en el siguiente árbol de soluciones:



¿En qué nodos del árbol de soluciones dado se cumplen las restricciones explícitas? ¿En qué nodos del árbol de soluciones dado se cumplen las restricciones implícitas? ¿Qué nodos del árbol de soluciones dado representan soluciones óptimas?

Problema 2 (30 puntos)

Sea $G=$ $(V‹A)$ un grafo con ciclos, no dirigido, conexo, sin lazos ni aristas múltiples. Cada arista $(v_i,v_j)\in A$ tiene asociado un costo positivo $Costo(v_i,v_j)$. Se denota con $n=$ ♣V♣ la cantidad de vértices de G.

1. Escriba el seudocódigo de $Ciclo(G)$ que devuelve un ciclo (como lista de vértices) de G. Es decir, devuelve una lista $(v_1,v_2,$⊳⊳⊳‹$v_h,v_{h+1})$, con $v_1 = v_{h+1}$ y $(v_i,v_{i+1})\in A$ para $i\in[1..h]$.
   Si hay más de un ciclo devuelve cualquiera de ellos. Los vértices son de tipo $Vertice$, que es el subrango $[1..n]$.
2. Muestre el funcionamiento del algoritmo en el siguiente grafo: $A=$ ¶$(1,2),(2,3),(3,$‹$4),(3,5),(5,$‹$6),(6,$‹$2),(6,7)$♢.
3. Escriba el seudocódigo de MasCostosa(C) que, dada la lista de vértices C que forman un camino no vacío de G, devuelve la arista de mayor costo en C.
4. Asuma que la cantidad de aristas de G es igual a la cantidad de vértices. Escriba el seudocódigo de $ACCM(G)$, que devuelve un árbol de cubrimiento de costo mínimo de G. No puede utilizar un algoritmo que use la propiedad MST vista en el curso.
5. Fundamente en lenguaje natural por qué su algoritmo devuelve un ACCM.