Departamento de Computación
FCEFQyN, Universidad Nacional de Río Cuarto
Asignatura: Diseño de Algoritmos / Algoritmos II
Primer Cuatrimestre de 2016

Examen Parcial No. 1

Este examen debe ser resuelto en forma individual. No olvide poner su nombre y número de documento en el encabezado de la resolución.
Este examen consta de seis ejercicios. **Sólo cuatro de ellos deben resolverse**. No resuelva más de cuatro ejercicios, pues los ejercicios sobrantes no serán considerados.
Cada ejercicio tiene un valor de 2,5 puntos. La nota mínima de aprobación es de cinco puntos.

Ej. 1. El juego *MasterMind* es un juego de ingenio que involucra dos jugadores, cada uno de los cuales escribe un código secreto que el otro jugador debe intentar adivinar. Cada código es un número de 4 dígitos, sin repetidos. Para que un jugador pueda adivinar qué número esconde el otro jugador, propone una clave, a la cual otro jugador responde cuántas coincidencias tiene con su número secreto. Se necesita desarrollar una función de apoyo a este juego que, dada una clave propuesta y un número de coincidencias (las coincidencias son aquellos dígitos de la clave propuesta que coinciden con la clave secreta, en valor y posición), retorne todas las soluciones candidatas. Por ejemplo, si la clave propuesta es $[1,2,3,4]$, y el número de coincidencias es 4, el conjunto de soluciones candidatas es $\{[1,2,3,4]\}$; si en cambio la clave propuesta es $[1,2,3,4]$, y el número de coincidencias es 3, las soluciones candidatas son las siguientes:

$$\{[0,2,3,4],[5,2,3,4],[6,2,3,4],[7,2,3,4],[8,2,3,4],[9,2,3,4],[1,0,3,4],[1,5,3,4],[1,6,3,4],[1,7,3,4],[1,8,3,4],\ldots\}.$$

Implemente, usando Java o Haskell, este algoritmo.

Ej. 2. Como parte del entrenamiento de un deportista, el mismo debe seleccionar rutinas de ejercitación y alimentos de manera tal de compensar, con los alimentos, las calorías consumidas en los ejercicios. Suponiendo que todos los ejercicios y opciones de alimentación se encuentran cargados en un conjunto de números enteros S, donde cada número representa o bien las calorías consumidas por un ejercicio (valores negativos) o las calorías adquiridas por un alimento (valores positivos), se desea determinar si existe una rutina de ejercicios y alimenticia que mantenga las calorías en cero. Más precisamente, si podemos tomar un subconjunto no vacío $R \subseteq S$, tal que la suma de los valores (calorías) de R sea cero.

Diseñe un algoritmo que resuelva este problema, e implemente el mismo en Java o Haskell.

Ej. 3. El problema de calcular el número mínimo de saltos para alcanzar una posición en un arreglo consiste en lo siguiente. Dado un arreglo de enteros, en el cual cada elemento representa el número máximo de pasos que se puede realizar hacia adelante desde esa posición, encontrar el número mínimo de saltos para llegar a la posición destino desde una posición origen (menor o igual a la primera). El siguiente programa resuelve este problema de forma recursiva:

```
int minJumps(int arr[], int l, int h) {
    if (h == l)
        return 0;
    if (arr[l] == 0)
        return INT_MAX;
    int min = INT_MAX;
    for (int i = l+1; i <= h && i <= l + arr[l]; i++) {
        int jumps = minJumps(arr, i, h);
        if (jumps != INT_MAX && jumps + 1 < min)
            min = jumps + 1;
    }
    return min;
}
```

Esta solución es sumamente ineficiente. Mejore esta solución de manera tal que diferentes subproblemas comunes que surjan en la resolución del mismo no sean computados repetidamente.

Ej. 4. Como parte de un sistema de monitoreo de una aplicación concurrente, se mantienen *logs* locales para los eventos de cada proceso, y un *log* global para la aplicación concurrente completa. Se desea determinar mediante análisis de los logs, si no hubo problemas de sincronización durante el almacenamiento de los logs. Para esto, se debe determinar si el log global del sistema es un *interleaving* (intercalado) de los logs locales de los procesos.

Escriba un programa en Java o Haskell que resuelva este problema, suponiendo sólo 2 procesos, es decir 3 logs: 1 local para cada proceso, y uno global. Por simplicidad, puede suponer que los eventos son caracteres, i.e., que los logs son strings. Por ejemplo, si los logs locales son $[a, a, b, c]$ y $[c, b, d, d]$ y el log global es $[c, a, a, b, b, d, d, c]$ la respuesta es *true*, mientras que si el log global es $[c, d, a, a, b, b, d, c]$, la respuesta es *false*.

Es su solución polinomial? En caso de no serlo explique cómo podrá mejorarla en cuanto a tiempo de ejecución, y por qué.

Ej. 5. Un árbol binario se denomina *lleno* si todos sus niveles están ocupados con nodos, es decir, si su altura es h, entonces cada nivel $0 \leq i \leq h$ tiene exactamente 2^i nodos. Escriba un programa en Java o Haskell que determine si un árbol binario está lleno. Cuál es el tiempo de ejecución de su programa? Justifique.

Ej. 6. Dado un árbol binario con valores enteros en los nodos, y un valor k, se necesita determinar si existe un camino desde la raíz hasta alguna hoja tal que la suma de los valores del camino sea igual a k. Diseñe e implemente en Java o Haskell un algoritmo que resuelva este problema. Cuál es el tiempo de ejecución de su programa? Justifique.