Examen de Teoría de la Programación

E. U. ING. TEC. EN INFORMÁTICA DE OVIEDO

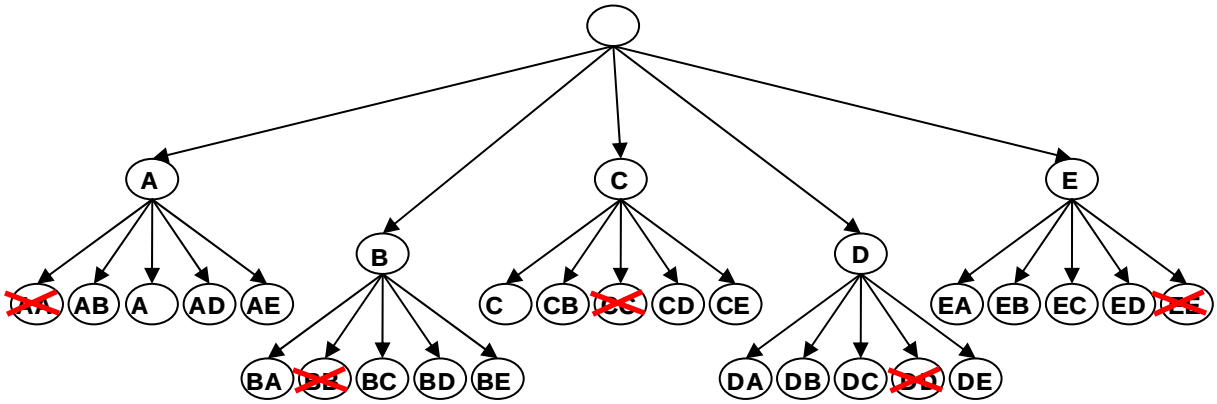
Final Septiembre – Curso 2007-2008 10 de septiembre de 2008

DNI _____ Nombre _____ Apellidos _____

Titulación: ☐ Gestión ☐ Sistemas

3. (2 puntos) Los Juegos Olímpicos de Pekín 2008 en vez de tener una única mascota han tenido 5. Los organizadores querían disponer de todas las combinaciones de *m* mascotas sin repetir ninguna, siendo *m<=5*. El problema se ha resuelto utilizando la técnica de Backtracking.

a) (0,5 puntos) Dibujar el árbol completo que genera la técnica de backtracking para este problema, cuando queremos formar combinaciones de dos elementos (m= 2).



b) (1,5 puntos) Completar el código Java para dar solución a este problema (escribirlo en los recuadros preparados a tal efecto).

```
public class Mascotas {
    private char[] mascotas;  // conjunto mascotas=5 identificadas por su inicial
    private int m;        // tamaño m de las combinaciones de mascotas
    private boolean[] estaElegida;        // array que marca las mascotas ya usadas
    private char[] solucion;              // array para almacenar la solución

    public Mascotas(char[] mascotas, int m) {
         this.mascotas= mascotas;
         this.m= m;
         estaElegida = new boolean[mascotas.length];
         solucion = new char[m];
    }

    public void combinarMascotas(________int posicion________){
         for(_____int i=0; i<mascotas.length; i++_____{
            if(__________!estaElegida[i]___________){
                estaElegida[i]=true;
                solucion[posicion]=mascotas[i];
                if(_____posicion<m-1_______)
                    combinarMascotas(________posicion+1__________);
                else {
                    imprimirArray(solucion);
                }
                estaElegida[i]=false;
            }
        }
    }

    private void imprimirArray(char[] array){ … }
```

```
    public static void main(String[] args) {
        char[] mascotas= null;
        int m= 0;

        // Instanciamos el array de mascotas con las cinco correspondientes
        …
        // Lectura del tamaño m de las combinaciones de letras
        …
        Mascotas masc = new Mascotas(mascotas,m);
        masc.combinarMascotas(_________0_________);
    }
}
```

4. (1,5 puntos) Las nuevas instalaciones olímpicas de Pekín 2008 necesitan un nuevo tendido que les proporcione la energía suficiente para su funcionamiento. Para optimizar la longitud de éste tendido los diseñadores chinos han desarrollado una aplicación con el algoritmo de Kruskal. El cual, dado un grafo conexo no dirigido, donde cada arista posee una longitud no negativa, permite calcular el árbol de recubrimiento mínimo (o árbol abarcador). Este árbol está formado por un subconjunto de las aristas del grafo dado, tal que utilizando este subconjunto todos los nodos queden conectados y además la suma de las longitudes de las aristas de este subconjunto es mínima.
En el algoritmo de Kruskal se elige una arista del grafo de entre las de menor peso, se añaden paulatinamente las aristas con menor peso siempre que estas no formen un ciclo con las otras ya incorporadas, el proceso termina cuando se han seleccionado n-1 aristas.

a) Explicar qué características del algoritmo de Kruskal permiten encuadrarlo dentro de los algoritmos voraces.

Se basa en una función heurística que guía al algoritmo en la selección del nodo adecuado. Esto evita recorrer un gran número de estados en el árbol.
Se elige el siguiente estado con información local: la arista más corta que conecta el grafo. Esta decisión no se revoca nunca, no hay vuelta atrás.

b) Escribir pseudocódigo para la función heurística.

Buscar la arista de menor peso que no forme ciclos con las ya seleccionadas.

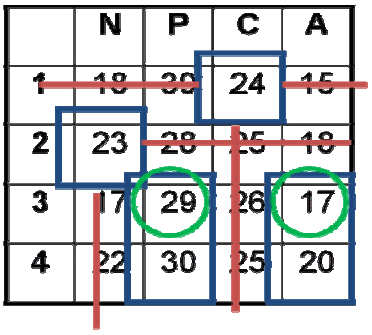
c) Escribir el código Java del método y su llamada que permita obtener la solución a este problema mediante el algoritmo descrito. Para simplificar se puede suponer la existencia de funciones auxiliares para trabajar con los nodos del grafo describiendo brevemente (una línea) las operaciones de cada una.

```
public void kruskal(Grafo grafo)
{
        int i= 0;
        int n= grafo.numNodos();
        Arista arbolRecubrimiento[]= new Arista[numNodos];

        // mientras no hayamos seleccionado n-1 aristas
        while (i<n)
        {
                // Heurístico:
                // Buscar la arista de menor peso
                Arista arista= grafo.getAristaMasCorta();
                if (!formaCiclo(arista,arbolRecubrimiento))
                {
                        // Marcar la arista como utilizada
                        arbolRecubrimiento[i]= arista;

                        i++;
                }
        }
}
```

5. (2 puntos) El cuadriatlón es una variante de triatlón en la que se combinan Natación(N)-Piragua(P)-Ciclismo(C)-Atletismo(A). Ante la proximidad de los juegos olímpicos de Pekín 2008 (aunque ésta no es una prueba olímpica) se ha propuesto, a modo de entrenamiento para los deportistas, hacer una competición. Se ha decidido que la prueba sea por relevos y cada participante realizará un deporte y le dará el relevo a su compañero. El seleccionador dispone de cuatro deportistas y de sus tiempos para cada una de las pruebas. Se quiere diseñar el algoritmo que mediante la técnica de ramificación y poda permita decidir al seleccionador que deportista realizará cada prueba.
Deportista 1 (18-39-24-15). Deportista 2 (23-28-25-18). Deportista 3 (17-29-26-17). Deportista 4 (22-30-25-20). Los tiempos están en el mismo orden en el que se realizan los deportes.

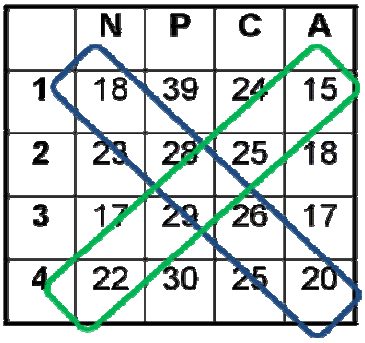
a) (0,25 puntos) Explicar cómo se calcula el heurístico de ramificación. Y aplicarlo al estado en el que asignamos al deportista 1 a ciclismo y al deportista 2 a natación (y quedan dos deportistas por asignar).

El cálculo del heurístico de ramificación consiste en: La suma de lo ya asignado en cada estado más el caso mejor (mínimo de columnas) de lo que queda por asignar.
Al estado 1→C, 2→N, le corresponde un coste de 24+23+29+17=93.

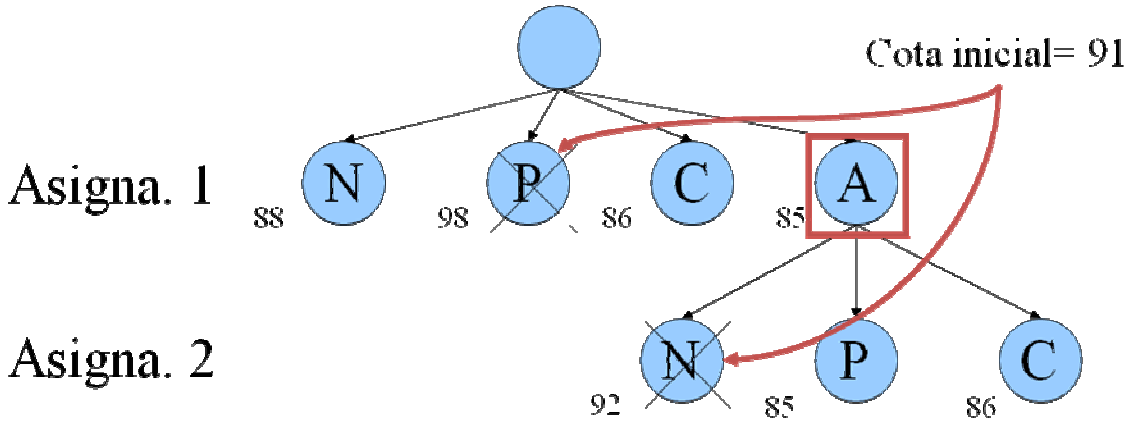


b) (0,25 puntos) Explicar cómo se calcula la cota inicial de poda y razonar cuándo se produce el cambio de esta cota.

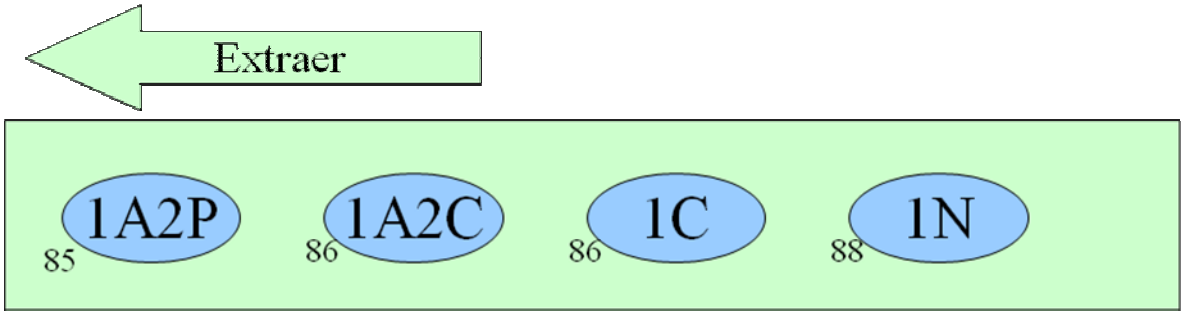
Inicialmente, es la menor de la sumas de las dos diagonales de la matriz de costes.
Valor cota inicial: min(91,92)=91.
Cuando en el desarrollo de los estados del árbol, lleguemos a una solución válida cuyo valor sea menor que el de la cota actual, se cambiará la cota a este nuevo valor.



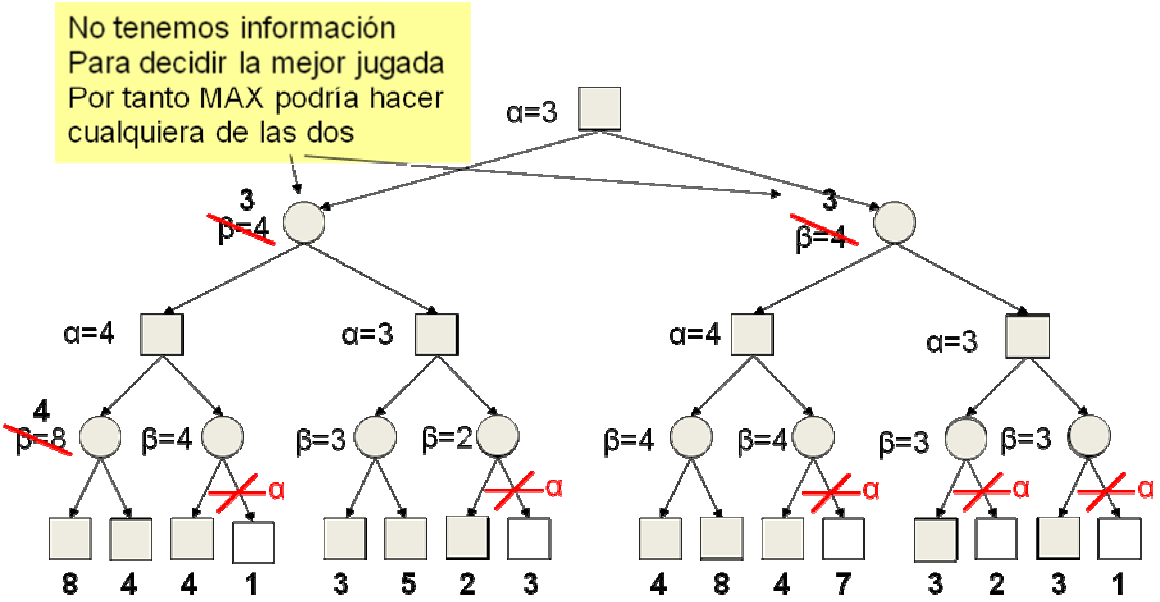
c) (1 punto) Representar el árbol de estados después de haber expandido dos estados del árbol.



d) (0,5 puntos) Representar de forma ordenada los estados que quedan en la cola de prioridad en la situación descrita en el punto c).



6. (1,5 puntos) Desarrollar la poda α-β para conocer que jugada debe realizar el jugador MAX, sobre el siguiente árbol:



a) Sombrear los nodos que haya que desarrollar
b) Escribir las cotas α y β,
c) Marcar los cortes e indicar si son de tipo α o β,
d) Por último, indicar que jugada debe elegir MAX para situarse en la mejor posición posible.