Examen de Teoría de la Programación

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

Final Febrero – Curso 2009-2010 26 de enero de 2010

DNI _____ Nombre _____ Apellidos _____

Titulación: ☐ Gestión ☐ Sistemas

3. (1 punto) Sea el algoritmo de ordenación *quicksort*, cuyo código se muestra a continuación:

```
public void quicksort(int[] a, int izq, int der) {
    int i = izq;
    int j = der;
    int pivote = a[ (izq + der) / 2];
    do {
      while (a[i] < pivote) {
        i++;
      }
      while (a[j] > pivote) {
        j--;
      }
      if (i <= j) {
        int aux = a[i];
        a[i] = a[j];
        a[j] = aux;
        i++;
        j--;
      }
    } while (i <= j);
    if (izq < j) {
      quicksort(a, izq, j);
    }
    if (i < der) {
      quicksort(a, i, der);
    }
  }
```

La complejidad de este algoritmo está en función de la elección del *pivote*. En este caso, como se aprecia, se ha escogido el elemento central del vector como *pivote*. Con este planteamiento el caso peor se daría cuando los elementos son todos iguales, o cuando el vector está inicialmente ordenado en orden creciente o decreciente. Calcular y justificar el caso mejor y peor de la complejidad de este algoritmo basándose en las tablas vistas en Divide y Vencerás.

4. (1,5 puntos) Antonio se pasa todo el día escuchando música y quiere cargar en el MP3 de su coche, que tiene una capacidad limitada, la mejor selección posible de canciones. Tiene que realizar este recopilatorio de las 2000 canciones que tiene disponibles, a las cuales pacientemente les ha ido asignado una puntuación en función de sus gustos y además tiene almacenado su tamaño. En el MP3 no caben todas las canciones, para crear el mejor recopilatorio debe maximizar la puntuación total de las canciones que contiene, teniendo en cuenta que los ficheros de canciones no se pueden partir. Como Antonio tenía que estudiar para diversos exámenes de la convocatoria de febrero, lo ha dejado para última hora y debe utilizar un algoritmo que calcule las canciones que debe incluir de forma inmediata.

a) **Justificar por qué debe aplicar un algoritmo voraz para resolver este problema.**
b) **Explicar el heurístico más adecuado para este caso.**
c) **¿Permite este heurístico obtener la solución óptima en cualquier caso? Demostrar la respuesta.**

5. (1,75 puntos) El problema de la asignación de tareas consiste en asignar *n* tareas a *n* agentes, de tal forma que cada agente realiza una única tarea y cada tarea sólo puede ser realizada por un único agente. A cada agente le cuesta realizar unas tareas más que otras, así que cada pareja *(agente, tarea)* tendremos asociado un coste. Esto se representa en una matriz. El objetivo del problema es buscar la asignación que minimice el coste total de realizar todas las tareas.

a) **Completar el código Java para dar solución a este problema con la técnica de Backtracking (escribirlo en los recuadros preparados a tal efecto).**

```
public class Main
{
    private int costes[][],  // almacena lo que le cuesta a cada agente realizar una tarea
            almacenEstados[],// estado actual del problema
            mejorSolucion[]; // contiene la mejor solución en cada momento
    private boolean tareasAsignadas[];// tareas ya asignadas (las que están a true)

    private int n, // Tamaño del problema.
            coste, // Almacena el coste acumulado en el cálculo de una solución.
            mejorCoste; // El mejor coste hasta el momento de una solución.

    […]

    public void ensayar(____________________)
    {
        for (__________________________________________________)
        {
            if (______________________________________)
            {
                almacenEstados[agente] = tarea;
                coste += costes[tarea][agente];
                tareasAsignadas[tarea] = true;


                if (____________________________)      // no es solución
                {
                    ensayar(____________________);
                }
                else        // es una solución posible
                    if (coste < mejorCoste)
                    {
                        ______________________________________________
                        ______________________________________________
                        ______________________________________________
                    }

                ______________________________________________
                ______________________________________________
                ______________________________________________
            }
        }
    } // Fin método
```

6. (1,5 puntos) Sea el problema de la asignación de tareas a agentes visto en el ejercicio 5. Se dispone de una matriz de costes de ejecución de una tarea j por un agente i.
Dados 4 agentes: a..d y 4 tareas: 1..4. Y la siguiente matriz de costes:

	1	2	3	4
A	18	39	24	15
B	23	28	25	18
C	17	29	26	17
D	22	30	25	20

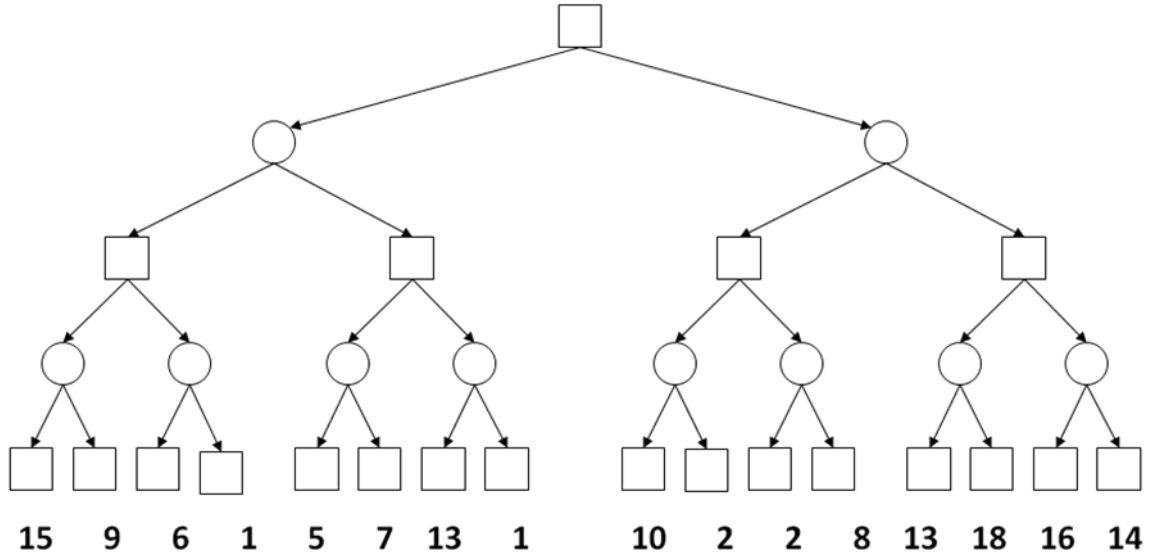
Nos planteamos resolver el problema mediante la técnica de ramificación y poda.

a) **(0,25 puntos) Explicar cómo se calcula el heurístico de ramificación para el estado donde asignamos la tarea 3 al agente b, cuando ya tenemos asignada la tarea 1 al agente a.**
b) **(0,25 puntos) Explicar cómo se calcula la cota inicial de poda y razonar cuándo se produce el cambio de esta cota.**
c) **(0,5 punto) Representar el árbol de estados después de haber expandido los primeros 2 estados (incluyendo el estado inicial)**
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).**

DNI _____ Nombre _____ Apellidos _____

Titulación: ☐ Gestión ☐ Sistemas

7. (1,25 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.**

Notas: El jugador que realiza el primer movimiento en el árbol es MAX. Los nodos del árbol se desarrollan de izquierda a derecha.

Ejemplo de indicaciones: