APELLIDOS, NOMBRE____ NOTA________

Ejercicio_1. (2 puntos)

Dado el esquema del algoritmo de ordenación QuickSort:

```
QuickSort (A, izq, der)  /* Ordena un vector A desde izq hasta der */
    if (izq < der) {
        piv=mediana (izq, der)
        div =partition (A, piv, izq, der)
        /* El vector A[izq..der] se particiona en dos subvectores A[izq..div] y A[div+1..der],
        de forma que los elementos de A[izq..div] son menores o iguales que los de A[div+1..der]
        (según elemento pivote) */
        QuickSort (A, izq, div)
        QuickSort (A, div+1, der)
        }
```

Donde, con "***mediana***" se obtiene la mediana de los elementos del array A entre las posiciones izq y der (el elemento que ocuparía la posición central si estuvieran ordenados), y "partition" es el procedimiento de particionar pero usando ***piv*** como pivote, con lo que el problema se divide en dos subproblemas de igual tamaño. Si el tiempo de ejecución del procedimiento "***mediana***" es $t_{med}(n)=20n$, y el de "partition" es $t_{par}(n)=n$:

a. (0,75 puntos). Calcular la complejidad del algoritmo propuesto por el método de la **ecuación característica**.

b. (0,25 puntos). Calcular la complejidad del algoritmo propuesto por el Teorema maestro.

c. (0,5 puntos). Calcular la complejidad del algoritmo propuesto por expansión de recurrencia.

d. (0,5 puntos). Si el método de la `Burbuja` tiene un tiempo de ejecución de n^2, justificar para qué valores de la entrada es preferible esta versión del `QuickSort` al método de la `Burbuja`.

NOTAS:

- Suma de los valores de la progresión geométrica $\sum_{i=0}^{n} 2^i = 2^{n+1} - 1$
- El Teorema maestro aplicado a **T(n) = aT(n/b) + Θ (n^k long^p n)** es:

$$T(n) \in \begin{cases} O(n^{\log_b a}) & \text{si } a > b^k \\ O(n^k \cdot \log^{p+1} n) & \text{si } a = b^k \\ O(n^k \cdot \log^{p} n) & \text{si } a < b^k \end{cases}$$

Ejercicio_2. (3 puntos)

- Resolver el problema de la mochila para el caso en que no se permita partir los objetos (es decir, un objeto se coge entero o no se coge nada).
  - Problema de la mochila.
    - Tenemos:
      - **n** objetos, cada uno con un peso ($\mathbf{p_i}$) y un valor o beneficio ($\mathbf{b_i}$)
      - Una mochila en la que podemos meter objetos, con una capacidad de peso máximo **M**.
    - **Objetivo:** llenar la mochila con esos objetos, maximizando la suma de los beneficios (valores) transportados, y respetando la limitación de capacidad máxima **M**.
    - Se supondrá que los objetos **NO** se pueden partir en trozos.
- **Se pide**:

a. (1.5 puntos). Diseñar un algoritmo **voraz** para resolver el problema aunque no se garantice la solución óptima. Es necesario marcar en el código propuesto a que corresponde cada parte en el esquema general de un algoritmo voraz (criterio, candidatos, función……). Si hay más de un criterio posible elegir uno razonadamente y discutir los otros. Comprobar si el algoritmo garantiza la solución óptima en este caso (la demostración se puede hacer con un contraejemplo).

- Aplicar el algoritmo al caso: **n**= 3, **M**= 6, **p**= (2, 3, 4), **b**= (1, 2, 5)

b. (1.5 puntos). Resolver el problema mediante **programación dinámica**. Definir la ecuación recurrente, los casos base, las tablas y el algoritmo para rellenarlas y especificar cómo se **recompone la solución** final a partir de los valores de las tablas.

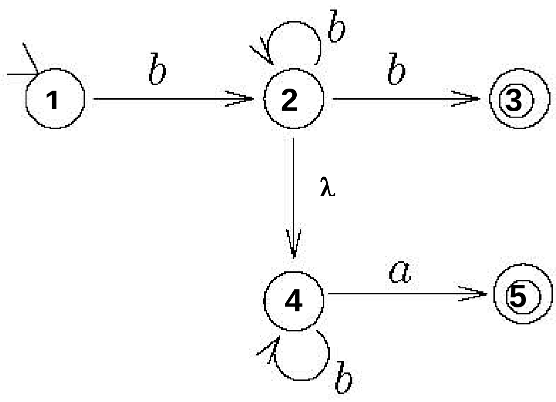
- Aplicar el algoritmo al caso: **n**= 3, **M**= 6, **p**= (2, 3, 4), **b**= (1, 2, 5)

- **Nota**: una posible ecuación recurrente es:

$$\text{Mochila}(k, m) = \begin{cases} 0 & \text{Si } \mathbf{k}=0 \text{ ó } \mathbf{m}=0 \\ -\infty & \text{Si } \mathbf{k}<0 \text{ ó } \mathbf{m}<0 \\ \max\{\text{Mochila}(k-1, m),\ b_k + \text{Mochila}(k-1, m-p_k)\} & \end{cases}$$

Ejercicio_3. (2 puntos)

- Dado el AFND definido en el grafo:



➢ **Se pide**:

a. (0.25 puntos). Si son aceptadas o no por el autómata las siguientes cadenas:

1. f'(1, ***ba***)
2. f'(1,***ab***)
3. f'(1,***bb***)
4. f'(1,***b***)
5. f'(1,***bba***)

b. (0,5 puntos). El AFD equivalente

c. (0,5 puntos). El AFD mínimo

d. (0,25 puntos). Corroborar el resultado obtenido para las palabras del apartado **a.** con el AFD obtenido en el apartado **c.**

e. (0,5 puntos). Obtener una expresión regular equivalente al AFD obtenido en el apartado **c.**

APELLIDOS, NOMBRE____ NOTA________

Ejercicio_4. (3 puntos)

Considérese la siguiente gramática:

$$
\begin{aligned}
S &\rightarrow (\ L\ ) \\
  &\mid a \\
L &\rightarrow L\ \%\ S \\
  &\mid S
\end{aligned}
$$

- **a.** (0,25 puntos). Comprobar si es LL(1) mediante el cálculo de los conjuntos Primero y Siguiente.
- **b.** (0.25 puntos). Con la gramática equivalente LL(1), especificar un autómata con pila que acepte el mismo lenguaje por pila vacía.
- **c.** (0.5 puntos). Analizar por el autómata del apartado **b.** anterior, teniendo en cuenta el principio de preanálisis (lectura de un símbolo de la entrada con anticipación) la entrada "**(a%(a%a))** ".
- **d.** (0,75 puntos) Con la gramática equivalente LL(1), construir la tabla de análisis LL(1) y especificar el pseudocódigo de análisis sintáctico tabular.
- **e.** (0,75 puntos) Construir la traza correspondiente al reconocimiento de la frase: "**(a%(a%a))** " según el pseudocódigo especificado en el apartado **d.** anterior.
- **f.** (0,5 puntos) Especificar el pseudocódigo de análisis sintáctico dirigido por la sintaxis para la gramática obtenida LL(1).