Universidad Rey Juan Carlos

Prueba de Diseño de Algoritmos – Curso 2014-2015 – 15 de junio de 2015

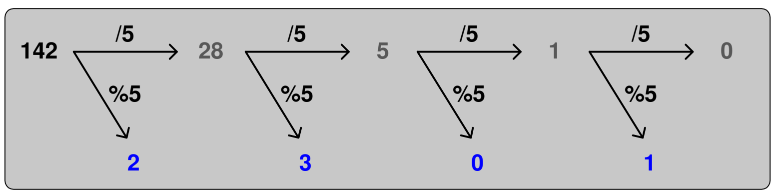
Diseño y Análisis de Algoritmos – Grado en Ingeniería Informática

Valor: 35 % de la nota final. Duración: **2 horas**

Se compensa el examen con 4 puntos de los 10 posibles

Ejercicio 1 [3 puntos]

Un mismo número se puede representar en diferentes "bases". Nosotros solemos usar números decimales (es decir, en base 10), donde los dígitos representan unidades, decenas, centenas, etc. Por ejemplo, el número 1473 se puede descomponer en $1 \cdot 10^3 + 4 \cdot 10^2 + 7 \cdot 10^1 + 3 \cdot 10^0$. De esta manera, los dígitos del número se multiplican por potencias de 10. Ahora bien, podemos representar ese número (1473) en otra base, obteniendo otros dígitos multiplicados por potencias de esa base. Por ejemplo, en base 5, $1473 = 2 \cdot 5^4 + 1 \cdot 5^3 + 3 \cdot 5^2 + 4 \cdot 5^1 + 3 \cdot 5^0$. En ese caso podemos escribir: $1473_{10} = 21343_5$, donde el subíndice denota la base.



La figura de arriba ilustra los pasos de un algoritmo que transformaría 142_{10} a 1032_5, donde '/' es división entera y '%' es el resto de la división entera. Se pide implementar una función recursiva que implemente el algoritmo, el cual calcula el cambio de base de un cierto número decimal n (entero), a su representación en una base entera $b \in [2, 9]$. Se trata de calcular el nuevo entero (no se pide imprimirlo por pantalla). Es decir, la función devolvería el entero 1032 para las entradas $n = 142$ y $b = 5$.

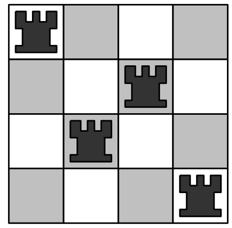
Ejercicio 2 [3.5 puntos]

Se pide implementar una función recursiva que calcule el producto de dos matrices $A \cdot B$ utilizando la técnica de **divide y vencerás**. La descomposición debe realizarse partiendo el problema en los siguientes bloques:

$$A \cdot B = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \cdot \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} = \begin{pmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{pmatrix}$$

Se deben implementar las funciones auxiliares que se empleen (por ejemplo, la suma de matrices). Las matrices no tienen por qué ser cuadradas.

Ejercicio 3 [3.5 puntos]



Uno de los problemas más conocidos que se resuelven mediante la técnica de *backtracking* es el de las n reinas de ajedrez. En este ejercicio se pide resolver el problema de las n torres, también usando *backtracking*. Las torres se pueden mover en horizontal y vertical, con lo cual se trata de ubicar n torres en un tablero cuadrado de $n \times n$ casillas, de manera que solo haya una en cada fila y en cada columna. En concreto se pide escribir un programa que calcule todas las posibles soluciones y las imprima por pantalla.