COMPUTACIÓN 1
Instituto de Computación

Examen - Diciembre de 2009 - 2ª parte

- Duración del examen: 3 Hs.
- No se podrá utilizar ningún tipo de material (apuntes, libro, calculadora, etc). Apague su teléfono celular.
- **Sólo** se contestarán preguntas sobre interpretación de la letra hasta 20 minutos antes de la finalización del mismo.
- Las partes no legibles del examen se considerarán no escritas
- En la primer hoja a entregar ponga con letra clara, en el ángulo superior derecho, su nombre, número de cédula de identidad y cantidad de hojas -en ese orden-; las demás hojas es suficiente con nombre, número de cédula y número de página.

Problema 1	22 ptos	

Dadas dos matrices $\mathbf{A=[[a_{ij}]]_{m\times n}}$ y $\mathbf{B=[[b_{ij}]]_{p\times q}}$, se define el producto de Kronecker entre las matrices A y B (lo notamos $A \otimes B$) de la siguiente manera:

$$A \otimes B = \begin{bmatrix} a_{11}\times B & a_{12}\times B & \cdots & a_{1n}\times B \\ a_{21}\times B & a_{22}\times B & \cdots & a_{2n}\times B \\ \vdots & \vdots & \ddots & \cdots \\ a_{m1}\times B & a_{m2}\times B & \cdots & a_{mn}\times B \end{bmatrix} = \left[\begin{array}{ccc:c:ccc} a_{11}b_{11} & \cdots & a_{11}b_{1q} & & a_{1n}b_{11} & \cdots & a_{1n}b_{1q} \\ \vdots & \ddots & \vdots & \cdots & \vdots & \ddots & \vdots \\ a_{11}b_{p1} & \cdots & a_{11}b_{pq} & & a_{1n}b_{p1} & \cdots & a_{1n}b_{pq} \\ \hdashline & \vdots & & \ddots & & \vdots & \\ \hdashline a_{m1}b_{11} & \cdots & a_{m1}b_{1q} & & a_{mn}b_{11} & \cdots & a_{mn}b_{1q} \\ \vdots & \ddots & \vdots & \cdots & \vdots & \ddots & \vdots \\ a_{m1}b_{p1} & \cdots & a_{m1}b_{pq} & & a_{mn}b_{p1} & \cdots & a_{mn}b_{pq} \end{array}\right]$$

Notar que el tamaño de la matriz $A \otimes B$ es $\mathbf{(m*p)\times(n*q)}$.

Se pide escriba en *Matlab* una función iterativa denominada **`kronecker`** que tome como entrada dos matrices A y B y devuelva el producto de Kronecker de las mismas.

Ejemplo:

$$\begin{bmatrix} 4 & 1 \\ 3 & 6 \end{bmatrix} \otimes \begin{bmatrix} 1 & 3 \\ -1 & 2 \end{bmatrix} = \begin{bmatrix} 4\times\begin{bmatrix} 1 & 3 \\ -1 & 2 \end{bmatrix} & 1\times\begin{bmatrix} 1 & 3 \\ -1 & 2 \end{bmatrix} \\ 3\times\begin{bmatrix} 1 & 3 \\ -1 & 2 \end{bmatrix} & 6\times\begin{bmatrix} 1 & 3 \\ -1 & 2 \end{bmatrix} \end{bmatrix} = \left[\begin{array}{cc:cc} 4\times 1 & 4\times 3 & 1\times 1 & 1\times 3 \\ 4\times(-1) & 4\times 2 & 1\times(-1) & 1\times 2 \\ \hdashline 3\times 1 & 3\times 3 & 6\times 1 & 6\times 3 \\ 3\times(-1) & 3\times 2 & 6\times(-1) & 6\times 2 \end{array}\right] =$$

$$= \left[\begin{array}{cc:cc} 4 & 12 & 1 & 3 \\ -4 & 8 & -1 & 2 \\ \hdashline 3 & 9 & 6 & 18 \\ -3 & 6 & -6 & 12 \end{array}\right]$$

Nota: Para la resolución de este ejercicio no se permite el uso de la función `kron` de *Matlab*, ni ninguna otra función que resuelva de manera trivial el problema.

Problema 2	18 Ptos(6, 6, 6)	

a) Implementar una función recursiva en MatLab que dado un vector y dos números (un mínimo y un máximo) retorna la cantidad de elementos intermedios del vector (elementos que son mayores o iguales que el mínimo y son menores o iguales que el máximo).

b) Implementar una función recursiva en MatLab que dada una matriz y dos números (un mínimo y un máximo) retorna un vector que indica para cada COLUMNA de la matriz cuantos elementos intermedios tiene (elementos mayores o iguales que el mínimo y menores o iguales que el máximo).

c) Implementar una función iterativa en MatLab que dada una matriz y dos números (un mínimo y un máximo) retorna los indices de las COLUMNAS que contienen la menor y la mayor cantidad de elementos intermedios (elementos mayores o iguales que el mínimo y menores o iguales que el máximo).

Problema 3	30 ptos(6, 6, 6, 12)	

Se desea implementar la función **opRara** la cual dado dos números naturales opera con los mismos a nivel de su representación en binario puro y retorna el resultado de la operación en representación decimal. La operativa de la función **opRara** es la siguiente: dados dos números naturales (en representación de base 10) se llevan a su representación en base 2. Luego de obtenidas las representaciones binarias de los parámetros de entrada, se realiza un XOR (O-exclusivo) bit a bit**.** Finalmente, el resultado de la función **opRara** se obtiene pasando el número obtenido en binario a su representación decimal (base 10).

Ejemplos:

opRara(10, 6) = 12
10 → 1010
6 → 0110
1100 → 12

opRara(10, 5) = 15
10 → 1010
5 → 0101
1111 → 15

opRara(10, 10) = 0
10 → 1010
10 → 1010
0000 → 0

Recordemos la tabla de verdad del O-exclusivo (XOR):

^	**0**	**1**
0	0	1
1	1	0

Para implementar la función **opRara** se deben de seguir los siguientes pasos:

a) Implementar en MatLab de forma recursiva la función **numToBin** que dado un número natural en base decimal retorna un vector de unos y ceros que representa el número en base binaria.

Ejemplos: numToBin(10) = [1, 0, 1, 0]
numToBin(6) = [1, 1, 0]
numToBin(5) = [1, 0, 1]

b) Implementar de forma iterativa la función **binToNum** que dado un vector de unos y ceros que representa un número en binario puro, retorna el número en su representación en base decimal (base 10).

Ejemplos: binToNum([1, 0, 1, 0]) = 10
binToNum([0, 0, 1, 1, 0]) = 6

c) Implementar la función **largoN** que dado un vector de unos y ceros que representa un número en binario y un número n (natural y mayor que 0), retorna un vector de unos y ceros el cual es el vector de entrada con largo n. Notar que si n es mayor que el largo de vector de entrada se agregan ceros a la izquierda, si n es menor que el largo del vector de entrada se pierden los números más significativos (los números de más a la izquierda).

Ejemplo: largoN([1, 0, 1, 0] , 10) = [0, 0, 0, 0, 0, 0, 1, 0, 1, 0]
largoN([1, 0, 0, 0, 0, 0, 0, 0, 0, 0] , 4) = [0, 0, 0, 0]
largoN([1, 0, 0, 0, 0, 0, 0, 1, 1, 0] , 4) = [0, 1, 1, 0]

d) Implementar la función **opRara**

Nota: Recordar que en MatLab la función ***xor*** implementa el operador lógico O-exclusivo.
