Examen - 17 de Diciembre de 2010 - 2ª parte

- Duración del examen: 2:30 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 30 minutos antes de la finalización del mismo.
- Escriba las hojas de un solo lado
- 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, salón en el cual desarrolló la prueba, 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.
- Al entregar su prueba recuerde firmar la planilla correspondiente

Problema 1	31 ptos (10,2,12,7)	

La elevación de una zona se puede representar mediante una matriz, donde cada entrada de la matriz representa la altura o elevación promedio de determinada sub-zona. Las coordenadas (o posición) de una sub-zona se pueden especificar mediante un vector con dos componentes, uno que indica la fila y otro que indica la columna en la matriz.

Si Mapa = $\begin{matrix} 1 & 2 & 1 & 2 \\ 3 & 4 & 3 & 2 \\ 3 & 4 & 0 & 2 \end{matrix}$ la altura en la coordenada [2,2] es 4.

a) Escriba una ***función*** **iterativa** ***`mas_bajo`*** en *Matlab* que tome como entrada una matriz que representa la elevación de una zona y devuelva la coordenada (posición de la matriz) donde se encuentra la sub-zona con menor altura.

b) Especifique el **cabezal** para una función que reciba una matriz que representa la elevación de una zona y una coordenada, y devuelva la coordenada de la sub-zona vecina con la menor elevación y que sea menor que la altura de la coordenada dada (en la matriz de ejemplo, las sub-zonas vecinas a [2,2] son las sub-zonas [1,2], [2,1],[2,3] y [3,2] por lo que la sub-zona vecina a [2,2] con la menor elevación es la [1,2], con altura 2). Si ninguno de los vecinos tiene menos altura que la coordenada original la función devuelve el punto ingresado. **No se debe implementar la función.**

c) Escriba una ***función*** **recursiva** ***`bajo_rapido`*** en *Matlab* que dadas una matriz que representa la elevación de una zona, y una coordenada, devuelva el *camino descendente* hasta una sub-zona que no tenga vecinas de menor altura. El camino descendente se deberá representar como una matriz de Nx2, donde la primera fila contiene la coordenada de origen (recibida como parámetro), la segunda contiene la sub-zona vecina a la primera con la menor elevación, y así sucesivamente hasta llegar a la N-ésima posición con la coordenada que no tenga vecinas de menor altura.

Bajo_rapido(Mapa, [2,2]) = [2,2 ; 1,2 ; 1,3]
Bajo_rapido(Mapa, [1,1]) = [1,1]

d) Escriba una ***función*** que dada una matriz que representa la elevación de una zona y la coordenada de una sub-zona, verifica si siguiendo la trayectoria de descenso descrita en la *parte c* a partir de la coordenada se llega a la sub-zona con menor altura. Devuelve 1 si se llega, 0 en caso contrario.

Notas: Para la parte c se puede considerar como implementada la función de la parte b.
Para la parte d puede (y se aconseja) utilizar las partes a y c.

a)
```
function coord= mas_bajo(M)
[n,m]=size(M);
coord=[1,1];
minimo=M(1,1);
for i=1:n
    for j=1:m
        if minimo > M(i,j)
            coord=[i,j];
            minimo=M(i,j);
        end
    end
end
```

b)
```
function coordMenor=menor_elevacion(M,coord)
```

c)
```
function camino= Bajo_rapido(M,coord)
coordMenor=menor_elevacion(M,coord)
if coordMenor(1)==coord(1) & coordMenor(2)==coord(2)
    camino=[coord];
else
    camino=[coord; Bajo_rapido(M,coordMenor];
end
```

d)
```
function y=Verificar(M,coord)
coordMinima=mas_bajo(M)
camino=bajo_rapido(M,coord)
if coordMinima(1)==coord(1) & coordMinima(2)==coord(2)
    y=1;
else
    y=0;
end
```

Problema 2	20 ptos (10,10)	

Se desea implementar la ***función sumaCol***. Dicha función recibe como parámetro de entrada una matriz *M* y retorna un vector *v* que almacena en la i-ésima posición la suma de los elementos de la columna *i* de la matriz *M*. Para ello, se pide:

a) Implementar la ***función sumaCol*** utilizando una estrategia **NO recursiva.**

b) Implementar la ***función sumaCol*** utilizando una estrategia **recursiva.**

a)
```
function V = sumaCol(M)
[m,n] = size(M);
V = zeros(1,n);
for i=1:m
    for j=1:n
        V(j) = V(j) + M(i,j);
    end
end
```

b)
```
function V = sumaColrec(M)
[m,n] = size(M);
if (m == 1)
    V = M;
else
    V = sumaCol(M(2:m,:));
    for i=1:n
        V(i) = V(i) + M(1,i);
    end
end
```

Problema 3	25 ptos (10,8,7)	

Un ensayo que se realiza con un esclerómetro permite determinar la uniformidad de un hormigón o comparar la calidad de un hormigón con otro de referencia. El ensayo consiste en medir varias veces el rebote de una masa al chocar contra la superficie del hormigón.

Parte a) Realizar una función **iterativa** en ***Matlab*** llamada **promedio** que dado un vector con los valores medidos de los rebotes obtenidos, devuelva el promedio.

Parte b) Realizar una función **recursiva** en ***Matlab*** llamada **vectAcep** que dado un vector con los valores medidos de los rebotes y el valor del promedio de los rebotes, devuelva un vector que contenga los valores de los rebotes cuya medida difiere hasta en 5 unidades del promedio.

Parte c) El resultado de un ensayo será exitoso cuando al descartar las medidas de los rebotes que estén en un entorno superior a 5 unidades del promedio aritmético, se tengan por lo menos 5 medidas. En caso contrario, se considera que el ensayo no fue exitoso. Se pide realizar una función **iterativa** en ***MatLab*** llamada ***esclTest*** que dados los valores medidos de los rebotes en forma de vector, devuelva como resultado 1 si el ensayo es exitoso y además devuelva un vector con los valores aceptados. En caso contrario, la función debe devolver el valor 0 y un vector vacío.

El siguiente es un ejemplo de la ejecución de la función esclTest:

```
>> [aceptado, valores]= esclTest([2, 4, 5, 6, 4, 12, 1, 6])
    aceptado = 1
    valores =[2, 4, 5, 6, 4, 1, 6]
```

a)

```
function P=promedio(v)
n=length(v);
S=0;
for i=1:n
    S=v(i)+S;
end
P=S/n;
```

b)

```
function vectAceptado= vectAcep(v,P)
n=length(v);
if n==0
    vectAceptado=[];
else
    if v(1)>P-5 & v(1)<P+5
        vectAceptado=[v(1),vectAcep(v(2:n),P)];
    else
        vectAceptado=vectAcep(v(2:n),P);
    end
end
```

c)

```
function [exito, vectorAceptado]=esclTest(v)
P=promedio(v);
vectorAceptado=vectAcep(v,P);
n=length(vectorAceptado);
if n>5
    exito=1;
else
    exito=0;
    vectorAceptado=[];
end
```