TRANSMISIÓN DE DATOS 2005/2006		
Examen Prácticas		**6 de Septiembre de 2006**
		Calificación
Apellidos, Nombre		
DNI		

Normas

- Se dispone de un máximo de hora y media (1h 30m) para la realización del examen.
- El examen es individual.
- Se podrá hacer uso del código de las prácticas realizadas durante el curso.
- Se puntuará la corrección de los resultados, pero también la "calidad" del código (legibilidad, comentarios, uso de MATLAB, etc.).
- Se entregará esta memoria, así como un fichero .zip (por el método de entrega de prácticas habitual indicando el grupo A) con el siguiente nombre: ***"examenTxDatosSeptiembre06_PrimerApellido_SegundoApellido_Nombre.zip".***
- El fichero .zip entregado deberá contener todos los ficheros MATLAB necesarios para ejecutar el código escrito en el examen.
- La hora límite para entrega de los ficheros será las 16:35**.**
  A partir de dichas horas se penalizara con un punto por cada minuto de retraso.
- Este examen consta de tres ejercicios. Es **imprescindible obtener al menos 1,5 puntos en cada ejercicio** para que se evalúe el resto del examen.

1. Ejercicio 1: Codificación Huffman (3 puntos)

Considerar una fuente con cuatro posibles símbolos 1, 2, 3, 4 en la que dichos símbolos se codifican con dos bits por símbolo de la siguiente forma:

Símbolo	*Codificación*
1	00
2	01
3	10
4	11

De tal forma que el mensaje :

[1 2 3 2 1 2 1 4 4 3 2 2]

Se codificaría como la secuencia de bits:

[00 01 10 01 00 01 00 11 11 10 01 01]

1.1 Codificación Huffman

Escriba el código de una función MATLAB (guardar como `codificaHuffman.m`) que reciba un mensaje como una secuencia de cualquier longitud compuesta de los cuatro posibles símbolos 1, 2, 3, 4 (Nota: Considerar la secuencia de símbolos como un vector MATLAB usual). La función deberá

comprobar que la secuencia contiene al menos una muestra de cada uno de los cuatro posibles símbolos y en caso contrario dar un mensaje de error.

A partir de la secuencia recibida se pide que la función calcule y muestre:

- El mensaje codificado como secuencia de bits siguiendo la correspondencia expuesta anteriormente (Nota: La codificación puede realizarse bien como un vector secuencia de unos y ceros o bien como un vector de parejas de elementos 'xx' entrecomilladas, siendo x 0 ó 1) .
- La probabilidad de aparición de cada uno de los cuatro posibles símbolos calculada a partir del mensaje recibido por la función.
- El código Huffman correspondiente a cada símbolo basándose en la probabilidad de aparición de cada símbolo calculada anteriormente.
- El mensaje codificado mediante la aplicación de los códigos Huffman correspondientes a cada símbolo calculados anteriormente.
- Calcule los códigos Huffman y probabilidades para una extensión de fuente de orden 2.

Dado el mensaje compuesto por 20 símbolos M = [1 3 4 4 2 1 2 2 1 3 1 1 2 1 2 2 1 3 1 3] calcule:

1.1.1 El mensaje M codificado sin aplicación de la codificación Huffman:

1.1.2 La probabilidad de aparición de cada uno de los cuatro posibles símbolos en el mensaje M:

Símbolo	*Probabilidad*
1	
2	
3	
4	

1.1.3 El código Huffman asignado a cada uno de los 4 posibles símbolos para el mensaje M:

Símbolo	*Código Huffman*
1	
2	
3	
4	

1.1.4 El mensaje M codificado con aplicación de la codificación Huffman:

1.1.5 Los códigos Huffman correspondientes a una extensión de fuente de orden 2 y sus probabilidades asociadas (utilice las casillas que considere necesarias):

Probabilidad	*Código*	*Probabilidad*	*Código*

1.1.6 Comente los resultados obtenidos al codificar el mensaje M con y sin codificacion Huffman.

2. Ejercicio 2: Cuantificación (3 puntos)

2.1 Cuantificación Uniforme

Escriba el código de un cuantificador uniforme simétrico, sin nivel de reconstrucción igual a 0, de 6 bits, que será guardado en `PCMUniforme.m`. Se usará a_0=-$V_{sobrecarga}$=-2 Voltios.

El programa dibujará la representación temporal y el histograma de las siguientes señales:

- Señal Original.
- "Señal" Cuantificada.
- Señal Reconstruida.
- Señal de Error.

El programa deberá representar por pantalla la potencia de error total calculada mediante:

```
Perror = sum(error.^2)
```

El programa deberá permitir ver el código que se asignaría (esto es, al menos el valor decimal del intervalo de cuantificación).

2.1.1 Dibuje e indique los valores de decisión y reconstrucción del cuantificador uniforme simétrico, con nivel de reconstrucción igual a 0, de 6 bits, y valor de sobrecarga V.

2.1.2 Calcule los códigos, valores de reconstrucción y error de las siguientes muestras:

- *V1 = 1.2 voltios*

Palabra cuantificada	
Valor de reconstrucción	
Error de cuantificación	

- *V2 = 0.8 voltios*

Palabra cuantificada	
Valor de reconstrucción	
Error de cuantificación	

- *V3 = 2.5 voltios*

Palabra cuantificada	
Valor de reconstrucción	
Error de cuantificación	

2.1.3 Una vez desarrollado el codificador, ejecute el programa sobre el fichero de audio `sample_audio.mat` y dibuje (mediante el programa MATLAB) las gráficas con sus correspondientes etiquetas e indicando los valores de los ejes de:

- Señal original.
- "Señal cuantificada".
- Señal reconstruida.
- Señal de error.

2.1.4 Calcule la potencia de error y el valor máximo del error:

Potencia de error	
Error máximo	

2.1.5 Comente los resultados:

3. Ejercicio 3: Códigos Lineales (4 puntos)

Generar el código lineal C(6,3) que incluye como palabras código las siguientes: {(100100),(010111),(111001)}, así como sus matrices generatriz (G) y de chequeo de paridad (H), y usarlo para codificar una secuencia y calcular síndromes. Todo el código se guardará en `CodigoLineal63.m`

3.1 Desarrollo de generador de códigos lineales

Desarrollar un código MATLAB para generar una matriz generatriz sistemática del código lineal anterior.

3.1.1 Escribir la matriz generatriz generada del código lineal:

3.1.2 Escribir los mensajes del código (6,3) y sus correspondientes palabras código:

Mensaje	Palabra Código

3.2 Codificación Lineal

Sea la secuencia de 12 bits:

1	1	0	1	0	1	0	0	1	1	1	1

Desarrollar el código MATLAB para codificarla con el código anterior.

3.2.1 Escriba el resultado del programa MATLAB al codificar la secuencia con el código anterior:

3.3 Desarrollo de detector de errores

Desarrollar un código MATLAB para generar la matriz de chequeo de paridad H correspondiente a la matriz G calculada anteriormente.

3.3.1 Escribir la matriz H del código (6,3):

3.4 Cálculo de síndrome

Desarrollar un código MATLAB para calcular el síndrome. Calcule el síndrome de las siguientes palabras código recibidas {(001100), (010111),(101110)}.

3.4.1 Escriba los síndromes obtenidos para las palabras anteriores:

Palabra recibida	Síndrome
001100	
010111	
101110	

3.5 Detección de errores

Siendo la secuencia enviada la correspondiente al punto 3.2.1 se recibe la siguiente secuencia:

1	1	0	0	1	0	1	0	1	1	0	0	0	0	1	0	1	1	1	1	1	0	0	0

Desarrollar un código MATLAB para calcular la secuencia de error que da lugar a la secuencia anterior.

3.5.1 Escriba la secuencia de error calculada:

3.5.2 Calcule el síndrome de las secuencias de código recibidas:

Palabra recibida	Síndrome

3.5.3 Comente los resultados: