# ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y COMPUTACIÓN

## ESTRUCTURAS DE DATOS

TERCERA EVALUACIÓN – I TÉRMINO 2011-2012

Nombre:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Matrícula: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Paralelo:\_\_\_\_\_\_\_

**Tema 1 (15 puntos)**

Considere los siguientes casos de estudio, donde se debe analizar las necesidades del sistema para proponer el uso de un TDA en específico. En cada caso, indique explícitamente mediante código JAVA que TDA utilizará y justifique su respuesta.

1. Una universidad desea generar unos talleres especiales con una cantidad fija de estudiantes (30). A los participantes se les asigna una identificación únicamente para el taller. correspondiente al número de la lista de estudiantes. Defina:
	1. El TDA estudiante que contenga las calificaciones ponderadas que se obtendrán en el taller. Agregue todos los campos que considere necesarios.
	2. La forma en la que almacenará en el sistema al conjunto de estudiantes, en un TDA Taller.
2. Similar al caso anterior, proponga las 2 TDA pero con los siguientes nuevos requisitos:
	1. Se debe de poder buscar al estudiante por su número de matrícula (previamente asignado y aleatorio). Esta operación debe de ser altamente eficiente y de tiempo constante.
	2. El tamaño del TDA puede variar entre 20 y 30 estudiantes.

**Tema 2 (10 puntos)**

En la tabla proporcionada a continuación se encuentran las frecuencias aproximadas de las 9 letras más frecuentes en el idioma castellano. Cree un árbol de huffman considerando que:

* En el árbol binario, la rama de la izquierda se codifica con 0 y la de la derecha con 1.
* Se debe poner siempre a la izquierda al elemento con menor frecuencia, el momento de unir dos símbolos. Si coinciden en frecuencia, se ordena alfabéticamente.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| E | 15 |  | N | 7 |
| A | 14 |  | I | 6 |
| O | 11 |  | D | 5 |
| S | 10 |  | T | 4 |
| R | 8 |  |  |  |

Decodifique la siguiente cadena:

**011011000011010111111 0111111110100100111**

**Tema 3 (15 puntos)**

Un grafo es completo si cada uno de los nodos es adyacente a los demás nodos del grafo. Implementar la función **boolean esCompleto(AdjacencyListGraph g)** que recibe un grafo e indica si es completo o no.

**Tema 4 (5 puntos)**

|  |  |
| --- | --- |
| 0 |  |
| 1 | -> CMB -> HSB |
| 2 |  |
| 4 | -> ADF -> EZF -> GJF |
| 5 | -> DFG |
| 6 |  |
| 7 | -> BMI -> FBI |
| 8 |  |
| 9 |  |

Considera la tabla de hash abierto presentada, implementada con un arreglo de listas:

Si se desea construir un hash cerrado con una función de redispersión lineal (es decir que incremente en 1 la clave en cada colisión) con los mismos datos.

1. ¿Tendría una solución única?
2. Justifique su respuesta indicando que factores afectan la posición final de los datos en el TDA.

**Tema 5 (5 puntos)**

Cual es la salida de el siguiente bloque de código para n=50:

Stack stack = new Stack();

while (n >= 1) {

stack.push(n % 2);

n = n / 2;

}

while(!stack.isEmpty())

 System.out.println(stack.pop());