Guía de ejercicios prácticos

A continuación se plantean una serie de problemas, que se deberán resolver utilizar el TDA árbol binario de búsqueda AVL, salvo que el ejercicio pida utilizar otro tipo particular de árbol.

1. Desarrollar un algoritmo que permita cargar 1000 número enteros –generados de manera aleatoria– que resuelva las siguientes actividades:
    a. realizar los barridos preorden, inorden, postorden y por nivel sobre el árbol generado;
    b. determinar si un número está cargado en el árbol o no;
    c. eliminar tres valores del árbol;
    d. determinar la altura del subárbol izquierdo y del subárbol derecho;
    e. determinar la cantidad de ocurrencias de un elemento en el árbol;
    f. contar cuántos números pares e impares hay en el árbol.
2. (NO)Implementar una función que permita cargar una expresión matemática en un árbol binario (no balanceado), y resuelva lo siguiente:
    a. determinar cuál de los barridos muestra la expresión en el orden correcto;
    b. resolver la expresión matemática y muestre el resultado.
3. (NO)Desarrollar un algoritmo que permita cargar el índice del libro Ingeniería de Software de Ian Summerville de manera automática desde un archivo de texto, transformando el árbol n-ario del índice en un árbol binario no balanceado mediante el uso de la transformada de Knuth, para resolver las siguientes actividades:
    a. listar el índice en su orden original;
    b. mostrar la parte del índice correspondiente al subtitulo “Diseño de software de tiempo real”;
    c. deberá almacenar además del texto de índice la página del libro donde está dicho tema;
    d. determinar cuántos capítulos tiene;
    e. determinar todos los temas que contengan las palabras modelo y métrica.
4. Implementar un algoritmo que contemple dos funciones, la primera que devuelva el hijo derecho de un nodo y la segunda que devuelva el hijo izquierdo.
5. Dado un árbol con los nombre de los superhéroes y villanos de la saga *Marvel Cinematic Universe* (MCU), desarrollar un algoritmo que contemple lo siguiente:

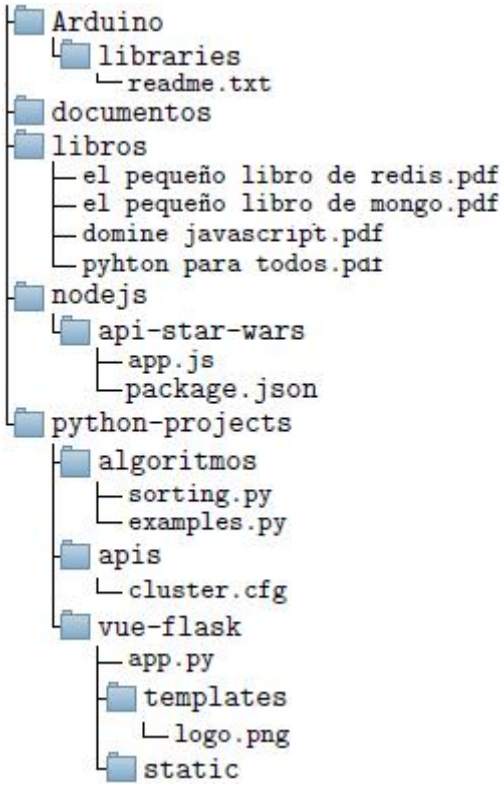
a. además del nombre del superhéroe, en cada nodo del árbol se almacenará un campo booleano que indica si es un héroe o un villano, *True* y *False* respectivamente;
b. listar los villanos ordenados alfabéticamente;
c. mostrar todos los superhéroes que empiezan con *C;*
d. determinar cuántos superhéroes hay el árbol;
e. Doctor Strange en realidad está mal cargado. Utilice una búsqueda por proximidad para encontrarlo en el árbol y modificar su nombre;
f. listar los superhéroes ordenados de manera descendente;
g. generar un bosque a partir de este árbol, un árbol debe contener a los superhéroes y otro a los villanos, luego resolver las siguiente tareas:
    i. determinar cuántos nodos tiene cada árbol;
    ii. realizar un barrido ordenado alfabéticamente de cada árbol.

6. Dado un archivo con todos los Jedi, de los que se cuenta con: nombre, especie, año de nacimiento, color de sable de luz, ranking (Jedi Master, Jedi Knight, Padawan) y maestro, los últimos tres campos pueden tener más de un valor. Escribir las funciones necesarias para resolver las siguientes consignas:
    a. crear tres árboles de acceso a los datos: por nombre, ranking y especie;
    b. realizar un barrido inorden del árbol por nombre y ranking;
    c. realizar un barrido por nivel de los árboles por ranking y especie;
    d. mostrar toda la información de Yoda y Luke Skywalker;
    e. mostrar todos los Jedi con ranking “Jedi Master”;
    f. listar todos los Jedi que utilizaron sabe de luz color verde;
    g. listar todos los Jedi cuyos maestros estan en el archivo;
    h. mostrar todos los Jedi de especie “Togruta” o “Cerean”;
    i. listar los Jedi que comienzan con la letra *A* y los que contienen un “-” en su nombre.
7. (no)Partiendo del árbol n-ario del directorio, que se observa en la siguiente figura, implementar los algoritmos necesarios para poder transformarlo a un árbol binario no balanceado –utilizando la transformada de Knuth–. Tener en cuenta que los archivos serán nodos hojas –es decir que estos no pueden tener hijos– y además resolver las siguientes actividades:
    a. el nodo deberá tener un campo que indique si es un directorio o un archivo;
    b. realizar un barrido inorden del árbol;

c. listar el contenido de la carpeta /Imágenes;

d. contar cuantos archivos hay en cada carpeta;

e. mostrar todos los archivos



8. Desarrollar un algoritmo que implemente dos funciones, una para obtener el mínimo nodo del árbol y la segunda para obtener el máximo.

9. Poe Dameron, líder del escuadrón negro de la Resistencia, tiene dificultades para transmitir los mensajes a la base de la Resistencia, dado que los mismos son muy largos y los satélites espías de la Primera Orden los intercepta, en un lapso muy corto desde que se transmiten. Por lo cual, nos solicita desarrollar un algoritmo que permita comprimir los mensajes para enviarlos más rápido y no puedan ser interceptados. Contemplando los siguientes requerimientos, implementar un algoritmo que los resuelva:

   a. crear un árbol de Huffman a partir de la siguiente tabla:

Símbolo	Frecuencia
A	0.2
F	0.17
1	0.13
3	0.21
0	0.05

M	0.09
T	0.15

b. desarrollar las funcionas para comprimir y descomprimir un mensaje.

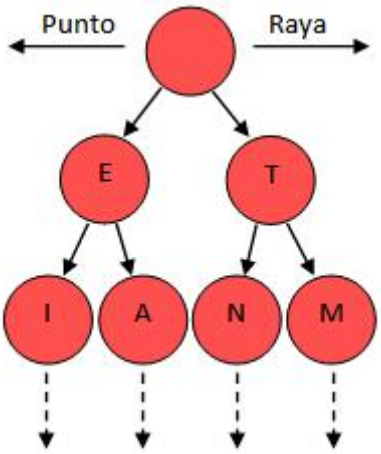
10. Desarrollar dos algoritmos, el primero que permita calcular en el número de nodos de un nivel del árbol –a partir de un nivel ingresado–. La segunda que cuente los nodos que hay en dicho nivel –dado que podría no estar completo–, para responder las siguientes actividades:
    a. determinar si el nivel del árbol está completo;
    b. ¿cuántos nodos faltan para completar dicho nivel?
11. Escribir un algoritmo que permita resolver las siguientes actividades:
    a. contar el número de nodos del árbol;
    b. determinar el número de hojas del árbol;
    c. mostrar la información de los nodos hojas;
    d. determinar el padre de un nodo;
    e. determinar la altura de un árbol.
12. Generar un árbol binario que tenga nueve niveles, luego diseñar los algoritmos necesarios para resolver las siguientes actividades:
    a. generar un bosque cortando los tres primeros niveles del árbol;
    b. contar cuántos nodos tiene cada árbol del bosque;
    c. realizar un barrido preorden de cada árbol del bosque;
    d. determinar cuál es el árbol con mayor cantidad de nodos;
    e. indicar que árboles del bosque están completos.
13. Nick Fury, líder de la agencia S.H.I.E.L.D., tiene la difícil tarea de decidir qué vengador asignará a cada nueva misión –por ahora considere que solo se asignará un superhéroe por cada misión–. Por lo que nos solicita desarrollar un árbol de decisión para resolver esta tarea con los siguientes requerimientos:
    a. cada nodo hoja del árbol debe ser un superhéroe y en cada nodo intermedio inclusive el raíz debe haber una pregunta;
    b. si la respuesta es sí, se debe desplazar hacia el subárbol izquierdo, si es no al subárbol derecho;
    c. desarrollar una función que determine el superhéroes para una misión;
    d. los Guardianas de la Galaxia son ideales para misiones intergalácticas en equipo;

e. Ant-Man es excelente en misiones de recuperación donde sea necesario no se detectado;
f. para misiones de destrucción Hulk es una excelente opción;
g. el Capitán América es un supersoldado de ética incorruptible ideal para comandar misiones de defensa y de recuperación;
h. Capitana Marvel es muy poderosa y puede viajar por las distintas galaxias;
i. Spider-Man es muy hábil y puede ser útil para varias misiones;
j. para misiones de recuperación donde requiera infiltrarse con personas del lugar, Black Widow es la indicada;
k. Iron Man es un líder para planear misiones de defensa, además es un genio y domina el manejo de tecnología avanzada, cuenta con un traje muy poderoso;
l. cuando se requiere elegir cuál será la próxima acción a tomar y moverse rápidamente de un lugar a otro, Doctor Strange es la opción más lógica;
m. Thor tiene el poder para destruir ejércitos completos;
n. no se debe utilizar árbol balanceado.

14. Desarrollar un algoritmo que permita decodificar mensajes en código morse, para ello deberá resolver las siguientes consignas:
    a. generar un árbol que contenga todo el alfabeto y los dígitos del 0 al 9, cuyos códigos morse están en la siguiente figura:

A ● –	G – – ●	M – –	S ● ● ●	Y – ● – –	5 ● ● ● ● ●
B – ● ● ●	H ● ● ● ●	N – ●	T –	Z – – ● ●	6 – ● ● ● ●
C – ● – ●	I ● ●	O – – –	U ● ● –	1 ● – – – –	7 – – ● ● ●
D – ● ●	J ● – – –	P ● – – ●	V ● ● ● –	2 ● ● – – –	8 – – – ● ●
E ●	K – ● –	Q – – ● –	W ● – –	3 ● ● ● – –	9 – – – – ●
F ● ● – ●	L ● – ● ●	R ● – ●	X – ● ● –	4 ● ● ● ● –	0 – – – – –

    b. cada nodo del árbol contendrá una letra o un digito, el cual se debe construir de manera manual (en un árbol que no sea auto balanceado), cuya raíz es vacía y, a partir de esta, la izquierda significa punto y la derecha guion –y se cargaran según su codificación morse, como se observa en la siguiente figura:



c. el espacio separa cada letra de una palabra y la “/” separa cada palabra;

d. descifrar los siguientes mensajes:

Mensaje 1 (Dr. Abraham Erskine): .--. .- ... . / .-.. --- / --.- ..- . / .--. .- ... . / -- .- .- -. .- / .--. .-. --- -- - .- -- . / .- .-.. --. --- / --.- ..- . / ... . --. ..- .. .-. / ... .. . -. -.. --- / ..- ... - . -.. / -. --- / ..- -. / ... --- .-.. -.. .- -.. --- / .--. . .-. ..-. . -.-. - --- --..-- / ... .. -. --- / ..- -. / -... ..- . -. / .... --- -- -... .-. . .-.-.

Mensaje 2 (Rocket Raccoon): -. --- ... --- - .-. --- ... / ... --- -- --- ... / .-.. --- ... / -- .- .-.. -.. .. - --- ... / --. ..- .- .-. -.. .. .- -. . ... / -.. . / .-.. .- / --. .- .-.. .- -..- .. .- .-.-.

Mensaje 3 (Natasha Romanoff): -.-- --- / ... --- .-.. --- / .- -.-. - ..- --- / -.-. --- -- --- / ... .. / . -. / ...- . .-. -.. .- -.. / .-.. --- / ... ..- .--. .. . .-. .- / - --- -.. --- .-.-.

Mensaje 4 (Tony Stark): -.-. .... .. -.-. --- ... / . ... - --- -.-- / .-.. .-.. . ...- .- -. -.. --- / .-.. .- / ..-. .. . ... - .- / .... .- -.-. .. .- / ..- ... - . -.. . ... .-.-.

Mensaje 5 (Steve Rogers): .--. --- -.. .-. .. .- / .... .- -.-. . .-. / . ... - --- / - --- -.. --- / . .-.. / -.. .. .- .-.-.

15. Desarrollar un algoritmo que permita implementar un árbol como índice para hacer consultas a un archivo que contiene personajes de la saga de Star Wars, de los cuales se sabe su nombre, altura y peso. Además deberá contemplar los siguientes requerimientos:

    a. en el árbol se almacenara solo el nombre del personaje, además de la posición en la que se encuentra en el archivo (*nrr*);

    b. se debe poder cargar un nuevo personaje, modificarlo (cualquiera de sus campos) y darlo de baja;

    c. mostrar toda la información de Yoda y Boba Fett;

    d. mostrar un listado ordenado alfabéticamente de los personajes que miden más de 1 metro;

e. mostrar un listado ordenado alfabéticamente de los personajes que pesan menos de 75 kilos;
f. deberá utilizar el TDA archivo desarrollado en el capítulo V;

16. Una empresa de nano satélites dedicada al monitoreo de lotes campo destinados al agro, tiene problemas para la transmisión de los datos recolectados, dado que la ventana de tiempo que dispone para enviar los datos antes de una nueva medición es muy corta, por lo que nos solicita desarrollar un algoritmo que permita comprimir la información para poder enviarla más rápido, para lo cual se debe tener en cuenta los siguientes requerimientos:
    a. la información transmitida por el nano satélite son estado del tiempo, humedad del suelo, y tres dígitos que identifican el lote al cual pertenecen los datos;
    b. desarrollar un árbol de Huffman que permita comprimir la información para transmitirla, la frecuencia de la información transmitida se observa en la siguiente tabla:

Variable	**Símbolo**	**Frecuencia**
Estado del clima	Despejado	0.22
	Nublado	0.15
	Lluvia	0.03
Humedad del suelo	Baja	0.26
	Alta	0.14
Código de identificación 1	1	0.05
	2	0.01
Código de identificación 2	3	0.035
	5	0.06
Código de identificación 3	7	0.02
	8	0.025

    c. comprimir un mensaje y descomprimirlo, para ver si no se pierde información durante el proceso de codificación, la trama enviada por el nano satélite tiene el siguiente formato (estado del clima-humedad del suelo-cod1-cod2-cod3), por ejemplo la siguiente trama es válida “NubladoBaja157”, –los guiones son a fines de comprender como está formada la trama pero no forman parte de la misma–;
    d. determinar la diferencia en tamaño de memoria utilizada por la trama original y la trama comprimida –puede utilizar la función getsizeof() de la librería sys–.

17. Se tiene un archivo con los Pokémons de las 8 generaciones cargados de manera desordenada (890 en total) de los cuales se conoce su nombre, número, tipo/tipos, debilidad frente a tipo/tipos, para el cual debemos construir tres árboles para acceder de manera eficiente a los datos almacenados en el archivo, contemplando lo siguiente:
    a. los índices de cada uno de los árboles deben ser nombre, número y tipo;
    b. mostrar todos los datos de un Pokémon a partir de su número y nombre –para este último, la búsqueda debe ser por proximidad, es decir si busco "bul" se deben mostrar todos los Pokémons cuyos nombres comiencen o contengan dichos caracteres–;
    c. mostrar todos los nombres de todos los Pokémons de un determinado tipo agua, fuego, planta y eléctrico;
    d. realizar un listado en orden ascendente por número y nombre de Pokémon, y además un listado por nivel por nombre;
    e. mostrar todos los Pokémons que son débiles frente a Jolteon, Lycanroc y Tyrantrum;
    f. mostrar todos los tipos de Pokémons y cuántos hay de cada tipo.
18. La armería de la base Starkiller, central de la primera orden, almacena los registros de los reportes de fallos en armas de las tropas de sus principales generales –Kylo Ren, general Hux y capitana Phasma–. Para dicha labor, se solicita desarrollar un algoritmo que permita resolver las siguientes tareas:
    a. se debe registrar el nombre del general a cargo de la misión, fecha de la misión –a los fines del ejercicio considere como máximo 20 fechas de misiones–, código de blaster generado de manera aleatoria –de 8 dígitos y no puede estar repetido–, estado del blaster (si falló o no) y el tipo de soldado que portaba el blaster –Imperial Stromtrooper, Imperial Scout Trooper, Imperial Death Trooper, Sith Trooper o First Order Stromtrooper–;
    b. debe generar y cargar al menos 10 000 registros;
    c. determinar el total de armas que fallaron por general;
    d. indicar la cantidad y tipo de soldado de las misiones de Kylo Ren;
    e. determinar cuántos Sith Troopers salieron en misiones y a cuantos les fallaron los blasters;
    f. listar los códigos de los blasters de las misiones de una determinada fecha, indicando además el porcentaje de armas que fallaron;

g. mostrar los datos del blaster código “75961380” si fue utilizado en alguna misión.

19. Desarrollar los algoritmos necesarios que permitan almacenar libros, de los cuales se conoce su título, ISBN, autores, editorial y cantidad páginas en un archivo, contemplando los siguientes requerimientos y tareas:
    a. utilizar el TDA archivo desarrollado en el capítulo V;
    b. deberá cargar al menos 100 libros;
    c. implementar tres árboles para manejar los índices de acceso, estos serán por título, ISBN y autores. En cada nodo del árbol se almacenará el campo clave correspondiente y la posición en el archivo donde está el resto de la información;
    d. las búsquedas deberán ser de la siguiente manera:
        i. por exactitud en el árbol de ISBN,
        ii. que esté contenido en el árbol de autores –es decir si son más de un autor y busco por uno debería encontrarlo–,
        iii. por proximidad en el inicio del nombre en el árbol de título –si busco “Alg” debería encontrar todos los libros cuyo nombres comienzan así–

    Ahora resuelva las siguientes consultas mostrando toda la información de los libros correspondiente:
    a. los libros de los autores Tanenbaum, Connolly, Rowling, Riordan, Morgan Kass;
    b. mostrar los libros de “minería de datos”, “algoritmos” y “bases de datos”;
    c. mostrar los libros de más de 873 páginas;
    d. mostrar los datos del libro ISBN 9789504967453;
    e. mostrar el autor del libro “los 100”.
20. Implementar un algoritmo que permita generar un árbol de decisión meteorológico para la predicción del estado del tiempo basado en las reglas de la siguiente figura, considerando los siguientes requerimientos:
    a. los 5 posibles estados del tiempo son despejado, parcialmente nublado, mayormente nublado, nublado y lluvia;
    b. los nodos hojas del árbol representan los estados del tiempo que predice el árbol –en la figura están con negrita–;

c. en cada nodo deberá almacenar el nombre de la variable o campo que se utilizará en ese nodo para la decisión y el valor umbral: se avanza hacia la izquierda si el valor es menor o igual y se avanza a la derecha cuando el valor es mayor;

d. dado un nuevo registro con datos meteorológicos del cual se conoce temperatura, presión, humedad, visibilidad, velocidad del viento, se debe predecir el estado del tiempo de manera automática.

```
visibilidad <= 15
|  humedad <= 70
|  |  viento <= 8.7
|  |  |  viento <= 5: Despejado
|  |  |  viento > 5: Nublado
|  |  viento > 8.7: Parcialmente nublado
|  humedad > 70
|  |  visibilidad <= 8
|  |  |  presión <= 1013
|  |  |  |  humedad <= 96: Nublado
|  |  |  |  humedad > 96: Mayormente nublado
|  |  |  presión > 1013
|  |  |  |  viento <= 7.2
|  |  |  |  |  presión <= 1018
|  |  |  |  |  |  visibilidad <= 1: Lluvia
|  |  |  |  |  |  visibilidad > 1: Mayormente nublado
|  |  |  |  |  presión > 1018: Nublado
|  |  |  |  viento > 7.2: Nublado
|  |  visibilidad > 8
|  |  |  humedad <= 92
|  |  |  |  visibilidad <= 12: Despejado
|  |  |  |  visibilidad > 12: Mayormente nublado
|  |  |  humedad > 92
|  |  |  |  viento <= 12.2: Lluvia
|  |  |  |  viento > 12.2: Nublado
visibilidad > 15: Despejado
```

21. Parta de la base del árbol genealógico "*greek-gods*" (n-arios) que se observa en el siguiente link: https://drive.google.com/file/d/13lMB6A2k4zO2zq-wuTgdxZdgwUKLl4Tr/view?usp=sharing, y utilice la transformada de Knuth para convertirlo en un árbol binario que permita realizar las siguiente actividades (no se deben utilizar árboles balanceados):

    a. la raíz del árbol debe ser Urano;

    b. además del nombre de los dioses, deberá cargar una breve descripción de quien es o lo que representa, no más de 20 palabras;

    c. listar el árbol por niveles, es decir, mostrando primero los hermanos. Para esto desarrolle una función barrido llamada "hermanos(raíz)" que devuelva

todos los hijos derechos de un determinado nodo de un árbol general transformado a binario;

d. solo se representarán las relaciones padre-hijo, a excepción de los dioses que en la imagen no tengan padre, –en este caso se deberá cargar la relación madre-hijo–, en los demás la madre será almacenada en un campo del nodo;
e. dado el nombre de un dios mostrar los hijos de este;
f. dado el nombre de un dios mostrar su nombre, padre, madre, hermanos y sus hijos;
g. realizar un barrido inorden, preorden y por nivel de dicho árbol;
h. realizar un barrido inorden mostrando el nombre de cada dios y el de su madre;
i. mostrar todos los ancestros de un determinado dios;
j. generar un bosque eliminando el nodo Uranos:
    i. determinar cuántos árboles forman dicho bosque,
    ii. realizar un barrido inorden de cada árbol del bosque,
    iii. determinar cuántos nodos hay en cada árbol y cuál es el nombre del dios del nodo raíz del árbol más grande;
k. mostrar todos los hijos de Tea.

22. Implementar un algoritmo que permita generar un árbol con los datos de la siguiente tabla y resuelva las siguientes consultas:
    a. listado inorden de las criaturas y quienes la derrotaron;
    b. se debe permitir cargar una breve descripción sobre cada criatura;
    c. mostrar toda la información de la criatura Talos;
    d. determinar los 3 héroes o dioses que derrotaron mayor cantidad de criaturas;
    e. listar las criaturas derrotadas por Heracles;
    f. listar las criaturas que no han sido derrotadas;
    g. además cada nodo debe tener un campo “capturada” que almacenará el nombre del héroe o dios que la capturo;
    h. modifique los nodos de las criaturas Cerbero, Toro de Creta, Cierva Cerinea y Jabalí de Erimanto indicando que Heracles las atrapo;
    i. se debe permitir búsquedas por coincidencia;
    j. eliminar al Basilisco y a las Sirenas;

k. modificar el nodo que contiene a las Aves del Estínfalo, agregando que Heracles derroto a varias;
l. modifique el nombre de la criatura Ladón por Dragón Ladón;
m. realizar un listado por nivel del árbol;
n. muestre las criaturas capturadas por Heracles.

Criaturas	Derrotado por	Criaturas	Derrotado por
Ceto	-	Cerda de Cromión	Teseo
Tifón	Zeus	Ortro	Heracles
Equidna	Argos Panoptes	Toro de Creta	Teseo
Dino	-	Jabalí de Calidón	Atalanta
Pefredo	-	Carcinos	-
Enio	-	Gerión	Heracles
Escila	-	Cloto	-
Caribdis	-	Láquesis	-
Euríale	-	Átropos	-
Esteno	-	Minotauro de Creta	Teseo
Medusa	Perseo	Harpías	-
Ladón	Heracles	Argos Panoptes	Hermes
Águila del Cáucaso	-	Aves del Estínfalo	-
Quimera	Belerofonte	Talos	Medea
Hidra de Lerna	Heracles	Sirenas	-
León de Nemea	Heracles	Pitón	Apolo
Esfinge	Edipo	Cierva de Cerinea	-
Dragón de la Cólquida	-	Basilisco	-
Cerbero	-	Jabalí de Erimanto	-

23. Desarrollar los algoritmos necesarios para generar un árbol de Huffman a partir de la siguiente tabla –para lo cual deberá calcular primero las frecuencias de cada carácter a partir de la cantidad de apariciones del mismo–, para resolver las siguientes actividades:
    a. la generación del árbol debe hacerse desde los caracteres de menor frecuencia hasta los de mayor, en el caso de que dos caracteres tengan la misma frecuencia, primero se toma el que este primero en el alfabeto, el carácter “espacio” y “coma” se consideraran anteúltimo y último respectivamente en el orden alfabético;
    b. descomprimir los siguientes mensajes –cuyo árbol ha sido construido de la misma manera que el ejemplo visto anteriormente:
        i. Mensaje 1:
        “1000101110101100001011101000111000001101100000011110011110100101100001101001110011010001011101011111110100001

11100111111001111010001100011000000101101011111011111101110101101110011101101111001111111001010010100101000001011010110001011001101000111001001011000011001000110101101011111111111011011101110010000100101011000111111100010000111011001100101101000110111110101101000110111000000011100100101010001111110000110010110101110011001111010001100011000000101101011111001110011

ii. Mensaje 2:

"01101010110111001010001111010111001101110101101101000010001110101001011110100111111101110010100011110101110011011101010110100001000111010100101111010011111111011100101000111101011100110111010110000110001001101000111001001000110001011001100111001001000011110111101"

c. finalmente, calcule el espacio de memoria requerido por el mensaje original y el comprimido.

Carácter	Cantidad	Frecuencia
A	11	
B	2	
C	4	
D	3	
E	14	
G	3	
I	6	
L	6	
M	3	
N	6	
O	7	
P	4	
Q	1	
R	10	
S	4	
T	3	
U	4	
V	2	
' ' espacio	17	
, coma	2	

24. A partir del organigrama de un empresa de desarrollo que se presenta en la siguiente figura (árbol n-ario) implementar las funciones necesarias para resolver los siguientes requerimientos:

a. utilizar la transformada de Knuth para convertirlo en un árbol binario;

b. realizar un barrido inorden y por nivel de dicho árbol;

c. agregue una persona a cada puesto de la empresa, salvo a los puesto de desarrollador, tester y soporte al cliente, en estos cargue cinco personas;
d. mostrar todos los empleados dependiente del líder de proyecto;
e. mostrar todo los empleados dependientes directamente del director de proyectos;
f. mostrar todos los empleados del nivel tres del organigrama (recuerde que la raíz es el nivel uno).