UCAB Universidad Católica ANDRÉS BELLO
Escuela de Ingeniería Informática - Guayana

Algoritmos y Programación II

Guía de Problemas

1.- Escriba un algoritmo en pseudo código, para determinar el mayor de tres números.

2.- Escriba un algoritmo en pseudo código, para verificar si un número es par o impar.

3.- Escriba un algoritmo en pseudo código, para verificar si un número es primo.

4.- Dado un polinomio de la forma:

$aX^2 + bX + c$

Escriba un algoritmo en pseudo código, para calcular sus raíces.

5.- Dada una lista de elementos y las operaciones: primero, cola, insertar un elemento por el frente e insertar un al final: Escriba los algoritmos en pseudo código, para:

a.- Calcular la longitud de la lista

b.- Verificar si dos listas son iguales

c.- Concatenar dos listas

d.- Buscar un elemento de forma recursiva.

e.- Ordenar la Lista la lista de forma recursiva.

f.- Voltear (invertir) la lista de forma recursiva.

6.- Escriba un programa que C, que lea dos enteros y escriba su suma.

7.- Escriba una función en C, que dados un arreglo enteros retorne el mayor, menor y el promedio.

8.- Escriba una función en C, verifique la existencia de un elemento en un arreglo.

9.- Escriba una función en C, que calcule el número de ocurrencia de un elemento en un arreglo.

10.- Dado un arreglo en enteros realice una función en C, float relacionPI(int array[], int size), que calcule la división entre las suma de los valores pares y la suma de los valores impares.

11.- Dado un arreglo en enteros realice una función en C, float ratioEO(int array[], int size), que calcule la división entre las suma de los valores de los elementos pares y la suma de los valores de los elementos impares.

12.- Dado el juego de la vieja (tic-tac-toe) realice una función en C que verifique si una jugada a realizar resulta ganadora.

13.- Dado un tablero de Cuatro en Línea.

a.- Escriba un algoritmo que verifique si una jugada es ganadora.

b.- Escriba una función en C que implemente la parte a, NO puede utilizar variables globales.

14.- Escriba un programa que C, que dado un polinomio de las forma.

$a_o + a_1X + ... + a_{n-1}X^{n-1} + a_nX^n$

Evalúe el polinomio dado un valor de X, leyendo n y los coeficientes desde a_0 a a_n, calculando las potencias por multiplicaciones sucesivas y sin utilizar arreglos (en orden N).

15.- Escriba un programa que C, que dado un polinomio de las forma.

$a_nX^n + a_{n-1}X^{n-1} + \ldots + a_1X + a_o$

Evalúe el polinomio dado un valor de X, leyendo n y los coeficientes desde a_n a a_0, calculando las potencias por multiplicaciones sucesivas y sin utilizar arreglos (en orden N).

16.- Implemente en lenguaje C la función void sort(int Array[], size int) con cualquier algoritmo de ordenamiento no visto en clase. Explique como funciona.

17.- Se desea que usted desarrolle TAD para almacenar dos listas de elementos, donde cada una de las listas son de tipos de datos diferentes, con las siguientes premisas:

- La suma de las cantidades de ambos tipos de elementos, es siempre menor o igual a N.
- Los tipos de datos de los elementos difieren en sus atributos.
- Ambas listas se manejan bajo una política LIFO.

Se desea que usted:

a.- Diseñe una estructura de datos estática óptima para almacenar ambos grupos de elementos

b.- Asumiendo los atributos de los elementos del tipos1 son int campo1 y float campo2 y los de los elemento del tipo2 son char campo3 y char campo4, construya las funciones del TAD para insertar y remover elementos tipos1 e insertar y remover elementos tipos2.

18.- ¿Qué retornan las siguientes funciones? Y justifique su respuesta explicando cómo opera.

a.-

```
int misterio(int x, int y){
   int t;

   if(!y)
      return 1;
   t=misterio(x,y>>1);
   if(y&1)
      return t*t*x;
   return t*t;
}
```

b.-

```
unsigned long misterio(unsigned n){
```

```
    if(!n) return n;
    return misterio(n>>1)*10+(n&1);
}
```

c.-

```
void misterio(int *x, int *y){
    *x^=*y;
    *y^=*x;
    *x^=*y;
}
```

18.- Escriba en C, macro definiciones dado un número flotante, realice las siguientes funciones:

a.- Truncar

b.- Redondear

c.- Calcular la parte decimal

19.- Dada las declaraciones:

```
char d[3][] = {"uno","dos","tres"};
char *d[] =  {"uno","dos","tres"};
```

Cuales de las siguientes expresiones, son validas

a.- d[1][1] == 'o'
b.- d++
c.- &d
d.- *(d+3)

20.- Dada las siguientes declaraciones:

```
char m[][5] = {"uno",{'d','o','s','\0'},"tres"};
char *ap[] = {"uno","dos","tres"};
char **a = ap;
char ***x = &a;
```

Para la siguiente secuencia de instrucciones, complete el string del control (con %c o %s) e indique la salida respectiva considerando los efectos de las instrucciones previas, o indique que la instrucción contienen expresiones inválidas:

a.- `printf("%__ %__ %__",m[1][1],ap[1][1],a[1][1]);`
b.- `printf("%__ %__%__",a++[1][0],*(*(*x+1)+0),*(*m+5));`
c.- `printf("%__ %__",*a,m[1]);`
d.- `printf("%__ %__",*(m+2),*(ap+2));`
e.- `printf("%__ %__",**(m+1),**(ap+1));`

f.- `printf("%__ %__",*(*m+1),*(*ap+1));`

21.- Dado el siguiente programa en Lenguaje C.

```
#include <stdio.h>

int f0[]  = { 0, 1, 2, 3, 4},
    f1[]  = {10,11,12,13,14},
    f2[]  = {20,21,22,23,24},
    f3[]  = {30,31,32,33,34},
    f4[]  = {40,41,42,43,44},
    *fp[] = {f0,f1,f2,f3,f4},
    **mp = fp;

int ma[][5]={{ 0, 1, 2, 3,4},{10,11,12,13,14},
             {20,21,22,23,24},{30,31,32,33,34},
             {40,41,42,43,44}};

int main(){
   int i=4,j=4;

   printf("mp{%d}{%d}:%d\n",i,j,mp[i][j]);
   printf("ma[%d][%d]:%d\n",i,j,ma[i][j]);
   return 0;
}
```

Si el programa compila:

a.- Dibuje las estructuras de datos, partiendo de mp y ma.

b.- Escriba la salida.

En todos los casos justifique detalladamente sus respuestas.

22.- Cual es la salida, al ejecutar el siguiente programa

```
int main(){
    char c;

    for (c='a'; c<'g'; ++c){
        switch (c) {
            case 'a': c += 2;
            case 'c': c += 1;
            case 'g': ++c;
                      printf ("%c\n" , c-- );
            default: ++c;
        }
        printf ("*** %c\n" , c);
    }
```

```
    return 0;
}
```

23.- Cual es la salida del siguiente programa

```
#include <stdio.h>

int main(int argc,char **argv){
int i=5,j=10,*p=&i,**pp=&p;

     printf("valor de i:%d j:%d\n",i,j);
     printf("valor de *p:%d\n", *p);
     printf("valor de **pp:%d\n", **pp);
     p=&j;
     printf("valor de **pp:%d\n", **pp);
     return 0;
}
```

24.- Explique cual es el inconveniente del siguiente segmento de código y como se puede mejorar.

```
//   Mezcla arreglos de enteros positivos en orden
//   decreciente y terminados en 0. Asume espacio
reservado
void merge(const int *f1,const int *f2,int *d){
   while(*d++=*f1>*f2?*f1++:*f2++);
}
```

25.- ¿Qué problema tiene la siguiente función? Reescriba el código para arreglarlo.

```
char* WordInput(){
    char word[20];

    printf ("Type a word: ");
    scanf ("%s", word);
    return word;
}
```

26.- Sea s un apuntador a caracteres. Realice, en lenguaje C, las siguientes declaraciones.

a.- Un apuntador no constante a datos no constantes.

b.- Un apuntador no constante a datos constantes.

c.- Un apuntador constante a datos no constantes.

d.- Un apuntador constante a datos constantes.

27.- Escriba el valor de todas las variables que aparecen en cada instrucción

```
int main(){
      int x, y, z;
      float a, b;
      x = 3; y = 5.3; z = 2;    x ____ y ____ z ____
      a = 2.5; b = 3.14;        a ____ b ____
/*1*/ x += y + a * z-b;         x ____
/*2*/ a = b / (x%y + z)         a ____
      z += (x == y);            z ____
/*3*/ x == (a = b);             x ____
      return 0;
}
```

28.- Realice, en lenguaje C, una función que realice la conversión entre unidades.

```
//   convertir: retorna el resultado de convertir
//   el valor expresado en unidad1 a la unidad2
float convertir(float valor, int unidad1,int unidad2);
```

Por ejemplo:

```
#define MILIMETROS  1
#define CENTIMETROS 2
#define METROS      3
#define KILOMETROS  4
#define PULGADAS        5
#define PIES        6
#define YARDAS      7
#define MILLAS      8
…
float v_origen,v_destino;
…
v_destino = convertir(v_origen, METROS, PIES);
```

Adicionalmente, puede asumir que cuentas con definiciones con los factores de conversión:

```
#define METROS2CENTIMETROS 100.0
#define PULGADAS2CENTIMETROS 2.54
…
#define PIES2PULGADAS 12.0
```

29.- Según la siguiente definición:

```
Dado A un conjunto de elementos cualesquiera,
Sea L una lista de elementos de A
```

$L = L_1\ L_2\ \ldots\ L_n$ con $L_i \in A,\ 1 \leq i \leq n$

Se define:

$L^r = L_n\ L_{n-1}\ \ldots\ L_2\ L_1$ con $L_i \in A,\ 1 \leq i \leq n$

Como la reversa o inversa de L, donde $|L| = |L^r|$

Y dadas las siguientes declaraciones en lenguaje C:

```
typedef struct node {
      char name[20];
      int value;
      struct node *next;
} Node;

// add_front: añade el elemento apuntado por newp al
frente
// de una lista donde listp apunta al primer elemento
// retorna un apuntador al primer elemento de la lista
Node *add_front(Node *listp, Node *newp);

// add_end: añade un elemento apuntado por newp al final
// de una lista donde listp apunta al primer elemento
// retorna un apuntador al primer elemento de la lista
Node *add_end(Node *listp, Node *newp);

// reverse: invierte una lista donde listp apunta al
primer elemento
// retorna un apuntador al primer elemento de la lista
invertida
Node *reverse(Node *listp);
```

Se desea que usted implemente, en lenguaje C,

a.- La función reverse, versión iterativa

b.- La función reverse, versión recursiva

30.- Escriba un programa en C, que lea un número entero en notación decimal y lo escriba en notación octal y hexadecimal.

31.- Dado un arreglo de enteros realice una función en C, int contarElementos (int array[], int size, int elements[], int count[]), que cuente las ocurrencias de todos elementos en el arreglo y retorne el número de elementos distintos, realizarlo en $0(N^2)$.

32.- Explique el funcionamiento del Quicksort y escriba en lenguaje C una función que lo implemente.

33.- Dado un string con una secuencia de paréntesis (que abren y cierran en cualquier orden), escriba en C una función que verifiquen que están balanceado.

34.- Escriba en lenguaje C una función - char *dec2octhex(int, char *) - que dado un numero entero en base decimal, retorne un string con el numero en notación octal y hexadecimal.

35.- Escriba una función en C long potencia(int a, int n) que calcule a^n, sin iterar n veces.

36.- Implemente en lenguaje c, las funciones para manejo de string:

```
char *eliminar(char *cadena, char *subcadena)
char *reemplazar(char *cadena, char *subcadena,
                 char *subcadena1)
```

Que elimine todas las ocurrencias de subcadena en cadena y reemplace todas las ocurrencias de subcadena en cadena por subcadena1.

37.- Dada una lista simplemente enlazada con un apuntador a su primer elemento, escriba los algoritmos en pseudo código, para:

a.- Insertar un nuevo elemento por la cola.

b.- Eliminar la primera ocurrencia de un elemento.

c.- Voltear (invertir) la lista de forma recursiva.

38.- Para enteros muy grandes, de hasta 10000 dígitos. Se desea que usted defina el tipo EnteroGrande e implemente la función multiplicar enteros grandes en Lenguaje C.

39.- Escriba la funcion HasZeroDiagonal que, dada una matriz de NxN de enteros, retorne 1 si todos los elementos de la diagonal son 0 retorne 0 en caso contrario.

40.- Implemente la función char *trimAll(char *str) en C, que dado un string elimine los blancos al principio, al final y solo deje un blanco entre palabras.

41.- Dado el problema de las torres de Hanoi. Mover N disco desde A a C usando B, moviendo uno a ala vez y nunca colocar disco sobre uno más pequeño.

a.- Escriba un algoritmo iterativo que lo resuelva.

b.- Ejecutar el algoritmo anterior para N=4.

42.- La sucesión de Fibonacci, es la siguiente sucesión infinita de números naturales:

0, 1, 1, 2, 3, 5, 8, 13, …

La sucesión comienza con los números 0 y 1, y a partir de estos, cada elemento es la suma de los dos anteriores.

Se desea que realice en lenguaje C, funciones que calcule el n término de la sucesión de Fibonacci:

a.- De forma iterativa.

b.- De forma recursiva.

c.- Usando recursión de cola

43.- La estructura bicola o cola de doble entrada, es una extensión del TAD Cola, al que se le puede añadir a quitar elementos por cualquier extremo, los extremos de la bicola se pueden identificar con frente y fin (al igual que la cola) y sus operaciones básicas son:

CrearBicola, inicializa una bicola sin elementos

BicolaVacia, retorna true si la bicola es vacia

Ponerfrente, añade un elemento al frente

PonerFinal, añade un elemento al final

QuitarFrente, retira y retorna el elemento de frente de la bicola

QuitarFinal, retira y retorna el elemento del final de la bicola

Frente, retorna el elemento del extremo frente de la bicola

Final, retorna el elemento al extreno final de la bicola

Implemente el TAD Bicola usando arreglos circulares

44.- Utilizando solamente las operaciones del TAD Cola, escribir una función para realizar la copia de una cola especificada.

45.- Utilizando solamente las operaciones del TAD Pila, escribir una función para realizar la copia de una pila especificada.

46.- Modifique la implementación del TAD pila del tal forma que cuando se llena a pila, se amplié el tamaño del arreglo al doble de la capacidad actual.

47.- En la implementación de TAD conjunto con bits, escriba en lenguaje C y explique las operaciones de insertar, suprimir y miembro.

48.- Implementación TAD diccionario, con las operaciones insertar, buscar e eliminar, utilizando de Tablas de Hash con Dispersión Cerrada.

50.- Dada TAD ListasAnidadas, el cual puede contener como elementos caracteres o ListasAnidadas, se desea que usted

a.- Defina las estructuras de datos asociada al TAD

E implemente las siguientes funciones en C.

b.- Aplane la estructura de una lista.

Por ejemplo aplanar la ListaAnidada (a, (b, (c, d), e))

da como resultado (a, b, c, d, e)

c.- Empaquete elementos consecutivos duplicados de uma lista aplanada en sublistas.

Por ejemplo (a,a,a,a,b,c,c,a,a,d,e,e,e,e).

da como resultado ((a,a,a,a),(b),(c,c),(a,a),(d),(e,e,e,e))

d.- Elimine los consecutivos duplicados de una lista aplanada.

Por ejemplo (a,a,a,b,c,c,a,a,d,e,e,e,e)

da como resultado (a,b,c,a,d,e)

51.- Dado un arreglo de enteros en orden creciente, implemente en C la función int int b_binaria(int value, int array[], int size) que implementa la búsqueda binaria de forma iterativa y ejecútela en frió para buscar value=25 en Array = { 2, 4, 7, 10, 14, 16, 20, 24, 25, 30, 35 } .

52.- En la implementación en TAD conjunto utilizando los bits de un entero largo sin signo, se desea que usted escriba en lenguaje C las funciones para las operaciones Miembro(x,S), Inserta(x,S), Suprime(x,S) y Listar(S).

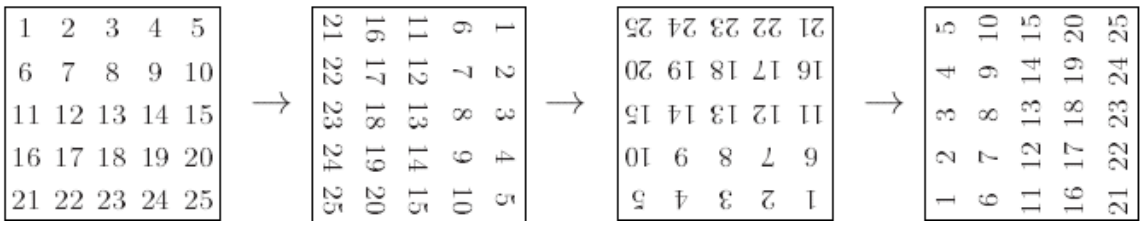
53.- Para la utilización de Nibles (cuatro bits o medio byte) en un arreglo de bytes sin signo, se deben escribir las siguientes funciones:

a.- Inicializar un arreglo de nibles

b.- Colocar en el i-ésimo elemento del arreglo de nible un valor

c.- Obtener el valor de i-ésimo elemento

54.- ¿Pueden las matrices dar volteretas?, Suponiendo que se tiene una matriz cuadrada de orden N, cuyos elementos son los números de 1 a N2, realice un función en lenguaje C, que dada la dimensión de la matriz n , imprima la matriz volteada un número v veces. Por ejemplo para una matriz de orden 5 tenemos:

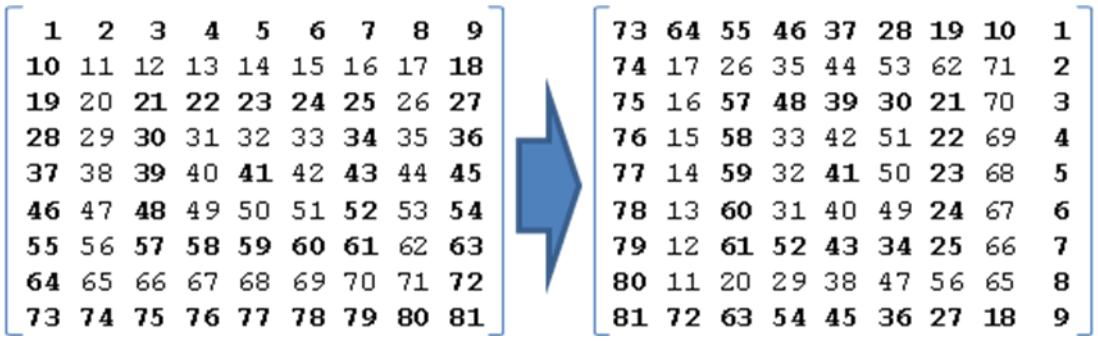


Matriz inicial Una voltereta Dos volteretas Tres volteretas

55.- ¿Podemos agitar una matriz?, Suponiendo que se tiene una matriz cuadrada de orden n. Realice una función en lenguaje C que agite la

matriz, es decir, si vemos la matriz como anillos concéntricos, el más externo debe rotar un cuarto de vuelta en sentido de las agujas de reloj (SAR), el que le sigue rotar un cuarto de vuelta en contra las agujas del reloj (CAR), el siguiente un cuarto de vuelta en SAR y así sucesivamente hasta llegar al centro.

Por ejemplo para una matriz de orden 9, los anillos en negritas rotan en SAR y los otros en CAR, como se puede verificar a continuación:



Se debe verificar que después de cuatro agitadas, queda la matriz original.

56.- Dada la siguiente declaración para un nodo de una lista doblemente enlazada:

```
typedef struct nodo{
     int elemento;
     struct nodo *ant_sig;
} nodo_t;
```

Apoyándose en un diagrama, explique como puede realizar el doble enlace utilizando un solo campo.

57.- Dada una lista de enteros doblemente enlazada, escriba una función en C, que elimine los consecutivos duplicados.

Por ejemplo (1,1,1,2,3,3,1,1,4,5,5,5,5)

da como resultado (1,2,3,1,4,5)

58.- Dado un string con una secuencia de paréntesis, llaves y corchetes que abren y cierran en cualquier orden, escriba en C una función que verifiquen que estén balanceados.

59.- En representaciones por filas de matrices dispersas o esparcidas, explique el método para solventar la dificultad de acceso eficiente por columnas de la representación, que se puede utilizar en la multiplicación de matrices. Escriba el algoritmo en pseudo-código.

60.- Escriba una función en lenguaje C, que dada una matriz la almacene en formato Compressed Row Storage (CRS).

61.- Escriba una función en lenguaje C, que multiplique de forma eficiente matrices almacenadas en formato Compressed Row Storage (CRS).

62.- Defina montículo binario, explique brevemente su representación utilizando un arreglo, las operaciones de inserción y borrado, y las ventajas de su uso en la implementación de TAD cola de prioridad.

63.- El código Gray, nombrado así en honor del investigador Frank Gray, es un sistema de numeración binario en el que dos valores sucesivos difieren solamente en uno de sus dígitos. Escriba un programa en C, que genere el código de Gray para n-bits.

Por ejemplo para 3 bits:

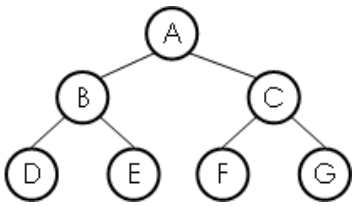
```
000
001
011
010
110
111
101
100
```

64.- En sistemas de computación, Binary-Coded Decimal (BCD) o Decimal codificado en binario es un estándar para representar números decimales en el sistema binario, en donde cada dígito decimal es

codificado con una secuencia de 4 bits. Escriba una función que codifique en BCD un entero positivo dado.

65.- Escriba un algoritmo que dado un árbol binario, calcule su altura.

66.- Dado el siguiente árbol:



Muestre los resultados de los recorridos en Preorden, Inorden y Postorden.

67.- Dada la siguiente declaración para un nodo de un árbol binario de busqueda:

```
Typedef struct nodo{
     Int elemento;
     Struct nodo *izq,*der;
} nodo_t;
```

Implemente en C las funciones **int menor**(nodo_t *arbolbp, **int** *elemento) e **int mayor**(nodo_t *arbolbp, **int** *elemento), que buscan el menor y el mayor elemento en un árbol binario y lo retorna en la variable elemento. La función debe retornar 0 si el apuntador al árbol es NULL y 1 en caso contrario.

68.- Dada la siguiente declaración para un nodo de un árbol AVL:

```
typedef struct AVLnodo{
     int elemento,altura;
     struct AVLnodo *izq,*der;
} *AVLp;
```

Implemente en C la función **static AVLp rotar_derecha**(AVLp p) que realiza la operación de rotación simple a la derecha dado un pivote. La función debe retornar el nuevo subárbol, así como realizar todas las verificaciones y actualizaciones necesarias. Así mismo en la respuesta se deben implementar todas las funciones auxiliares.

69.- Se desea que usted implemente una función en C, que reciba una lista de enteros y un apuntador a una función, que retorna un entero y tiene como parámetro un entero, retorne una lista que es el producto de aplicar la función pasada como parámetro a todos los elementos de la lista. Por ejemplo, dada la lista

(2, 4, 6, 8)

Y la función cuadrado(X)

El resultado es:

(4, 16, 36, 64)

70.- Se desea que usted implemente una función en C, que reciba una lista de enteros, un acumulador del tipo flotante y un apuntador a una función, que retorna un entero y tiene como parámetro un entero, retorne el resultado de acumular todos los elementos de la lista según la función. Por ejemplo, dada la lista

(1, 2, 3, 4, 5)

Y la función productor(A,X)

El resultado es:

A = 120

71.- Dada una expresión en notación polaca inversa, escriba un algoritmo en pseudo-código que la evalúe. Por ejemplo 2 1 + 5 3 - * da como resultado 6.

72.- Escriba en Lenguaje C la función long potencia(int x, int n) que calcula x^n con orden menor a n^2. Adicionalmente explique como opera y que orden tiene.

73.- Implemente el TAD diccionario, utilizando tabla de Hash con Dispersión Coalescente

74.- Escriba en Lenguaje C una función que imprima en orden las hojas de un árbol.

75.- Escriba en lenguaje C una función que dado dos nodo en un árbol binario de búsqueda verifique si el primero es descendiente del otro en un árbol binario.

76.- Escriba un algoritmo en pseudo código, que dada una lista ordenada de enteros, determine la secuencia de inserción en un árbol binario de búsqueda, de tal forma de al insertar el ultimo elemento se obtenga un árbol balanceado sin necesidad de rotaciones.

77.- Escriba las funciones que recorran un árbol binario en preorden, inorden y postorden sin utilizar de recursión.

78.- Escriba en lenguaje C una función que dado un árbol binario imprima en orden todos los nodos de un nivel.

79.- Implemente en lenguaje C una función que dados dos árboles binario de búsqueda verifiquen que son iguales.

80.- Implemente en lenguaje C una función que dados dos árboles binario de búsqueda verifiquen que contienen los mismos elementos.

81.- Escriba un programa en C, que dado un archivo, especificado en la línea de comando, imprima el número de caracteres, de palabras y de líneas del mismo.

82.- Implemente para el TAD Árbol AVL, la operación de insertar.

83.- Dado un árbol, escriba una función que calcule su número de nodos.

84.- Dado un nodo de un árbol binario, escriba una función en C que calcule su altura y su profundidad.

85.- Escriba programas en C, que implemente los comando cp, cat y sort de Linux, no es necesario implementar ninguna opción.