ESTRUCTURA DE DATOS 1
Código ST0245

Laboratorio Nro. 4
Tablas de Hash y Árboles

Resumen de Ejercicios a Resolver

1.1 Implementen un algoritmo para identificar las abejas robóticas que se encuentren a 100 metros o menos de distancia de otra abeja.
github

2.1 Resuelvan el problema usando árboles binarios
github

3.1 Expliquen qué estructura de datos utilizaron para calcular las colisiones entre abejas, por qué la eligieron y qué complejidad tiene el algoritmo que, utilizando dicha estructura de datos, calcula las colisiones.
Se implementa el octree por que es de los algoritmos mas eficientes en cuanto a la deteccion de colisiones en un plano tridimensional.
O(n)

3.4 Calculen la complejidad del ejercicio 2.1.

$T(n) = 2T(n/2) + C$; $O(n)$

4. Simulacro parcial

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas

4. Simulacro de p arcial en informe PDF

En los videojuegos, las tablas de hash se usan para guardar la información de las armas; por ejemplo, su ataque, su velocidad y su duración. Considera la siguiente función de hash para cadenas de caracteres. La constante `TABLE_SIZE` representa el tamaño máximo del arreglo con el que se representa internamente la tabla de hash.

```
private int funcionHash(String k){
  return (int(k.charAt(0))) % TABLE_SIZE;
```

¿Qué problema presenta esta función hash? Las cadenas...

(a) que terminan con la misma letra colisionan
(b) que inician con la misma letra colisionan
(c) cuya suma de los caracteres es la misma colisionan
(d) que tienen los mismos caracteres colisionan

¿Cuál es la complejidad asintótica, en el peor de los casos, de `funcionHash(k)`? Donde *n* es la longitud de la cadena *k*.

(a) $O(n)$

(b) $O(n^2)$

(c) $O(n.\log n)$

(d) $O(1)$

4.2 Según Andrés Mejía –egresado de Eafit; reconocido por haber trabajo en *Google* y *Facebook*; y, actualmente, ingeniero de software en *Riot Games*–, el principal reto que uno afronta al entrar a trabajar a una gran compañía –como *Riot Games*– es cómo entender el código que han hecho otras personas, entender su complejidad asintótica para el peor de los casos y poder realizar mejoras. El siguiente algoritmo es de vital importancia para empresas –como *Oracle* y *Microsoft*– porque se utiliza dentro de los compiladores de lenguajes orientados a objetos como Java y C#. Por si fuera poco, este ejercicio es común en entrevistas para grandes empresas, según el portal L*eetCode*. Imagina que llegas nuevo a una de estas empresas, y debes entender y modificar este código. Lo único que sabemos es que `left_mis` es el resultado de `mistery` del árbol izquierdo y `right_mis` es el resultado de `mistery` del árbol derecho.

Si trabajas en Java, revisa este código:

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas

```
public class BinaryTree {
 private Node root;
 Node mistery(int n1, int n2) {
  return mistery(root, n1, n2);
 }

 Node mistery(Node node, int n1, int n2) {
  if (node == null) return null;

  if (node.data == n1 || node.data == n2)
   return node;

  Node left_mis = mistery(node.left, n1, n2);
  Node right_mis =mistery(node.right, n1, n2);

  if (left_mis!=null && right_mis!=null)
   return node;

  return (left_mis != null) ? left_mis :
      right_mis;
}}
```

Si trabajas en Python, revisa este código:

D. Mauricio Toro Bermúdez
cente | Escuela de Ingeniería | Informática y Sistemas
rreo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627 Tel: (+57) (4)
1 95 00 Ext. 9473

```
def mistery(root, n1, n2):
 if root is None:
  return None

 if root.key == n1 or root.key == n2:
  return root

 left_mis = mistery(root.left, n1, n2)
 right_mis = mistery(root.right, n1, n2)

 if left_mis and right_mis:
  return root

 return left_mis if left_mis is not None else
     right_mis
```

1. ¿Qué retorna la función `mistery`?

Se le asignan dos valores, n1 y n2 (valores a buscar), el primer return, es la condición de parada, si al hacer la recursión se sale del ciclo y no encontró nada retorna el None, el segundo return, verifica si el valor en el nodo es igual a n1 o n2, en caso de que lo sea, retorna el nodo, el tercer return, retorna el nodo por izquiera y por derecha en caso de haber encontrado los dos valores n1 y n2 en ambos lados, y por último, como ya definimos que, si el nodo es None es porque ya se salio del arbol, entonces, en caso de estar en el lado izquierdo retorna el nodo izquierdo, o si está al lado derecho, entonces retorna el nodo derecho

D. Mauricio Toro Bermúdez
cente | Escuela de Ingeniería | Informática y Sistemas
rreo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627 Tel: (+57) (4)
1 95 00 Ext. 9473

2. ¿Cuál es la complejidad **asintótica**, en el peor de los casos, del algoritmo anterior o la ecuación de recurrencia para el peor caso?

O(n)

El siguiente problema es muy común en entrevistas de *Goldman Sachs* según el portal *Geeks for Geeks*. Dados dos arreglos *arr1* y *arr2*, de igual tamaño, la tarea es encontrar si los dos arreglos son iguales o no. Se dice que dos arreglos son iguales si ambos contienen el mismo conjunto de elementos, aunque el orden (o permutación) de los elementos puede ser diferente. Si hay repeticiones, entonces las ocurrencias de los elementos repetidos también deben ser iguales para que dos arreglos sean iguales. Las aplicaciones de este problema –en el sector bancario y financiero– son muy amplias, por la gran cantidad de transacciones.

Si trabajas en Java, revisa este código:

D. Mauricio Toro Bermúdez
cente | Escuela de Ingeniería | Informática y Sistemas
rreo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627 Tel: (+57) (4)
1 95 00 Ext. 9473

```
import java.util.*;
static boolean areEqual(int arr1[], int arr2
    []) {
 int n = arr1.length;
 int m = arr2.length;
 if (n != m)
  return false;
 Map<Integer, Integer> map = new HashMap<
     Integer, Integer>();
 int count = 0;
 for (int i = 0; i < n; i++) {
  if (map.get(arr1[i]) == null)
   map.put(arr1[i], 1);
  else {
   count = map.get(arr1[i]);
   count++;
   map.put(arr1[i], count);
  }
 }
 for (int i = 0; i < n; i++) {
  if (!map.containsKey(arr2[i]))
   return false;
  if (map.get(arr2[i]) == 0)
   return false;
  count = map.get(arr2[i]);
  --count;
  map.put(arr2[i], count);
 }
 ..............
}
```

D. Mauricio Toro Bermúdez
cente | Escuela de Ingeniería | Informática y Sistemas
rreo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627 Tel: (+57) (4)
1 95 00 Ext. 9473

Si trabajas en Python, revisa este código:

```
from collections import defaultdict
def areEqual(arr1, arr2, n, m):
 if (n != m):
  return False;
 count = defaultdict(int)
 for i in arr1:
  count[i] += 1
 for i in arr2:
  if (count[i] == 0):
   return False
  else:
   count[i] -= 1
 ..............
```

1. Completa la última línea..return True...
2. ¿Cuál es la complejidad asintótica, en el peor de los casos, de areEqual? Donde *n* es el número de elementos del primer arreglo y *m* el número de elementos del segundo arreglo. Es O(n)

4.4 Consideren la siguiente definición de árbol binario:

```
 class Node {
public Node left;
public Node right;
public String data;
public Node(String d){
```

D. Mauricio Toro Bermúdez
cente | Escuela de Ingeniería | Informática y Sistemas
rreo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627 Tel: (+57) (4)
1 95 00 Ext. 9473

```
data = d;
}
}
```

El siguiente algoritmo imprime todos los valores de un árbol en pre orden.

```
01        private void printAUX(Node node) {
02        if (node != null) {
03        System.out.println(node.data);
04        printAUX(node.left);
05        printAUX(node.right);
06        }
07        }
08        public boolean print() {
09        printAUX(root);
10        }
```

4.4.1 ¿Cuál es la ecuación de recurrencia que describe el número de instrucciones que ejecuta el algoritmo *print* en el peor de los casos?

La variable *n* representa el número de elementos del árbol.

De acuerdo a lo anterior, elijan la respuesta que consideren acertada:

a) $T(n)=T(n-1)+C$, que es O(n)
b) $T(n)=2.T(n-1)+C$, que es $O(2^n)$
c) $T(n)=2.T(n/2)+C$, que es O(n)
d) $T(n)=T(n/2)+C$, que es $O(\log_2 n)$
e) $T(n)=T(n+1)+C$, que es infinito

4.4.2 ¿Cuál ecuación de recurrencia que describe el número de instrucciones que ejecuta el algoritmo *print* en el peor de los casos?

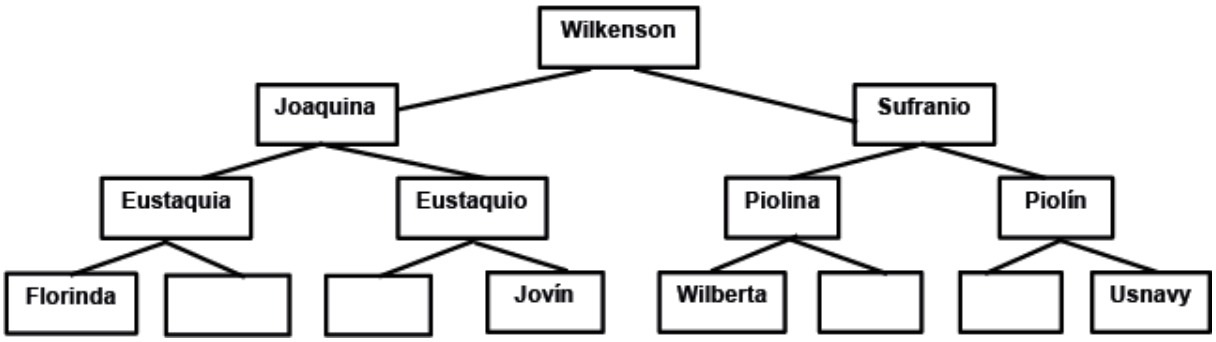
D. Mauricio Toro Bermúdez
cente | Escuela de Ingeniería | Informática y Sistemas
rreo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627 Tel: (+57) (4)
1 95 00 Ext. 9473

Elijan la respuesta que consideren acertada:

a) $O(n)$
b) $O(n^2)$
c) $(log\ n)$
d) $O(n.m)$
e) $O(1)$

4.4.3 ¿Cuál es la salida del algoritmo *print* para el siguiente árbol?

Tengan en cuenta que la raíz es Wilkenson. Después de hacer la rotación, usted entenderá que realmente las mujeres van a la izquierda y los hombres a la derecha.
Además, en un árbol genealógico, para algunas personas, no se conoce la mamá o el papá.



Elijan la respuesta que consideren acertada:

a) Wilkenson, Sufranio, Piolín, Usnavy, Piolina, Wilberta, Joaquina, Eustaquio, Florinda, Eustaquia, Yovín

D. Mauricio Toro Bermúdez
cente | Escuela de Ingeniería | Informática y Sistemas
rreo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627 Tel: (+57) (4)
1 95 00 Ext. 9473

b) Sufranio, Piolina, Wilberta, Piolín, Usnavy, Joaquina, Estaquia, Florinda, Wilkenson, Yovín, Eustaquio

c) Wilkenson, Yovín, Eustaquio, Sufranio, Piolina, Wilberta, Piolín, Usnavy, Joaquina, Estaquia, Florinda

d) Wilkenson, Joaquina, Eustaquia, Florinda, Eustaquio, Jovín, Sufranio, Piolina, Wilberta, Piolín, Usnavy

e) Sufranio, Piolina, Wilberta, Piolín, Usnavy, Florinda, Wilkenson, Yovín, Eustaquio, Joaquina, Estaquia

4.4.4 ¿Qué modificación hay que hacer algoritmo *print* para que arroje la siguiente respuesta para el árbol anterior?:

Usnavy, Piolín, Wilberta, Piolina, Sufranio, Florinda, Eustaquio, Yovín, Eustaquia, Joaquina, Wilkenson.

Tengan en cuenta que la raíz es Wilkenson. Como Wilkenson es la raíz, para poder entender el árbol, recomendamos rotarlo 180 grados. Después de hacer la rotación, usted entenderá que realmente las mujeres van a la izquierda y los hombres a la derecha

Además, en un árbol genealógico, para algunas personas, no se conoce la mamá o el papá.

Elijan la respuesta que consideren acertada:

a) Cambiar el orden de las líneas 03, 04 y 05 por 05, 04, 03
b) Cambiar el orden de las líneas 03, 04 y 05 por 04, 05, 03
c) Cambiar el orden de las líneas 03, 04 y 05 por 03, 05, 04
d) Cambiar el orden de las líneas 03, 04 por 06, 07
e) Intercambiar la línea 03 y la línea 04

D. Mauricio Toro Bermúdez
cente | Escuela de Ingeniería | Informática y Sistemas
rreo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627 Tel: (+57) (4)
1 95 00 Ext. 9473

4.5 Luis escribió un programa para insertar un número en un árbol binario de búsqueda. En dicho árbol, él quiere tener los números menores o iguales a la raíz a la derecha y los mayores a la raíz a la izquierda.

Ayúdele a completar su código. El algoritmo recibe la raíz de un árbol `p` y un número a insertar `toInsert`, y retorna la raíz del árbol con el elemento insertado donde corresponde.

```
01   private Node insert(Node p,int toInsert){
02   if (p == null)
03   return new Node(toInsert);
04   if (...........)
05     return p;
06   if (...........)
07   p.left = insert(p.left, toInsert);
08   else
09   p.right = insert(p.right, toInsert);
10   return p;
11   }
```

Completen los espacios en blanco, así:

a) Completen, por favor, la línea 4 con la condición que corresponde
toInsert != null

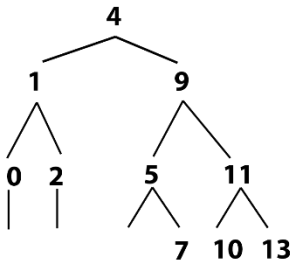
b) Completen, por favor, la línea 6 con la condición que corresponde
toInsert <= p

D. Mauricio Toro Bermúdez
cente | Escuela de Ingeniería | Informática y Sistemas
rreo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627 Tel: (+57) (4)
1 95 00 Ext. 9473

4.7 PDF Los siguientes algoritmos son el recorrido en in orden y pos orden. En la vida real, estos recorridos son de utilidad para hacer procesamiento del lenguaje natural.

```
void InOrden(Node node){
 if (node != null){
   InOrden(node.left);
   System.out.println(node.data);
   InOrden(node.right);
} }
void PosOrden(Node node){
if (node != null){
PosOrden(node.left);
   PosOrden(node.right);
   System.out.println(node.data);
} }
```

Consideren el recorrido in orden y pos orden de un **Árbol binario de búsqueda** con los siguientes elementos 4, 9, 1, 5, 7, 11, 13, 2, 0, 10. Ahora, la impresión del recorrido inorden nos devolverá los elementos, y, la impresión del recorrido pos-orden devolverá.



4.7.1¿Cuál es la impresión pos-orden?

Elijan la respuesta que consideren acertada:

D. Mauricio Toro Bermúdez
cente | Escuela de Ingeniería | Informática y Sistemas
rreo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627 Tel: (+57) (4)
1 95 00 Ext. 9473

a) 0, 2, 1, 7, 5, 10, 13, 11, 9, 4

b) 0, 1, 2, 4, 5, 7, 9, 10, 11, 13

c) 0, 2, 1, 7, 10, 5, 13, 11, 9, 4

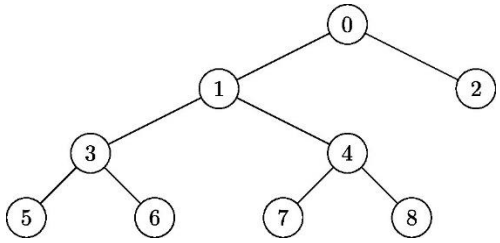
d) 4, 1, 0, 2, 9, 5, 7, 11, 13, 10

4.7.2 Supongan que hacemos la comparación elemento a elemento del recorrido en inorden y el recorrido en pos-orden. ¿Cuántos elementos aparecen en la misma posición en ambos recorridos? Como uno ejemplo, si un recorrido es 1,3,4,2 y el otro es 2,3,4,1, la respuesta es 2 porque el 3 y el 4 aparecen en la misma posición en ambos.

Elijan la respuesta que consideren acertada:

a) 7
b) 2
c) 5
d) 8

Nota: Consideren el siguiente árbol para los ejercicios 8,9 y 10



D. Mauricio Toro Bermúdez
cente | Escuela de Ingeniería | Informática y Sistemas
rreo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627 Tel: (+57) (4)
1 95 00 Ext. 9473

Elijan la respuesta que consideren acertada:

a) $O(n^3)$
b) $O(n^2)$
c) O(log n)
d) O(n)

4.8 Sean *A* y *B* las salidas de los recorridos pre-orden y pos-orden del árbol binario anterior, respectivamente. Determinen el número de elementos para los cuales se cumple que para $1 \leq i \leq 8$.

Elijan la respuesta que consideren acertada:

a) *3*
b) *2*
c) *4*
d) *0*

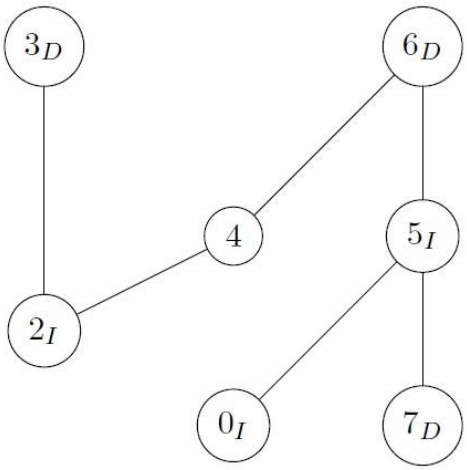
4.9 ¿Cuál es la salida del recorrido in-orden del árbol binario anterior?

Elijan la respuesta que consideren acertada:

a) 5, 3, 6, 1, 7, 4, 8, 0, 2
b) 0, 1, 3, 5, 6, 4, 7, 8, 2
c) 5, 6, 3, 7, 8, 4, 1, 2, 0
d) 5, 6, 3, 1, 7, 4, 8, 0, 2

D. Mauricio Toro Bermúdez
cente | Escuela de Ingeniería | Informática y Sistemas
rreo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627 Tel: (+57) (4)
1 95 00 Ext. 9473

4.10 PDF El sub-índice D indica que el nodo es un hijo derecho con respecto a su padre. El sub-índice I indica que es un hijo izquierdo con respecto a su padre



4.10.1 ¿Cuál es la salida del recorrido en *in-orden* del árbol anterior, tomando el nodo 4 como el nodo raíz?

Elijan la respuesta que consideren acertada:

a) 3, 2, 0, 7, 5, 6, 4

b) 2, 3, 4, 0, 5, 7, 6

c) 4, 2, 3, 6, 0, 5, 7

d) 4, 7, 3, 6, 2, 0, 5

4.10.2 Asuma que el nodo 4 es la raíz del árbol binario anterior. Asuma que la salida del recorrido *pre-orden* es $\{a_1,a_2,a_3,...,a_n\}$ y la salida del recorrido *in-orden* es $\{b_1,b_2,b_3,...,b_n\}$.

¿Cuál es el primer valor de *i* para el cual $a_i=b_i$? Note que la *i* empieza en 1.

D. Mauricio Toro Bermúdez
cente | Escuela de Ingeniería | Informática y Sistemas
rreo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627 Tel: (+57) (4)
1 95 00 Ext. 9473

a) *5*

b) 4

c) *3*

Por ejemplo, asuman que las salidas son *a={4,6,5,0,7,2,3} y b={6,4,7,0,5,2,3}*. La respuesta sería 4 porque $a_4 = b_4 = 0$ *e i=4* es el primer valor para el que $a_i=b_i$.

4.10.3 ¿Es un ***árbol binario de búsqueda*** el árbol anterior?

Elijan la respuesta que consideren acertada:

a) *Si*

b) No

D. Mauricio Toro Bermúdez
cente | Escuela de Ingeniería | Informática y Sistemas
rreo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627 Tel: (+57) (4)
1 95 00 Ext. 9473

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas