TAREA PROGRAMADA II

REPORTE

LUIS EDUARDO ROJAS CARRILLO – B86875

I INTRODUCCIÓN:

En el presente trabajo se desarrollarán estructuras de datos utilizadas para el guardado y manejo de información. Esto para poder realizar un estudio sobre el tiempo de ejecución, funcionamiento y eficacia de estas estructuras de datos. Para este trabajo se plantean cuatro estructuras de datos, las cuales son: lista enlazada con nodo centinela y árbol de búsqueda binaria, árboles rojinegros y tablas de dispersión. El objetivo de esta tarea programada es ejecutar dichas estructuras de datos y compararlas, estudiando la eficacia según la cantidad de datos a guardar y su tiempo de ejecución.

II METODOLOGÍA:

Para cumplir con el objetivo se realizó el diseño y ejecución de las cuatro estructuras de datos mencionadas anteriormente para estudiar su funcionamiento. Posteriormente los algoritmos se pusieron a prueba ingresando nodos y buscándolos en un rango de tiempo. La primer forma de ingresar los nodos fue de manera aleatoria, el objetivo fue cuantificar cuantas búsquedas distintas de manera aleatoria se podrían realizar en un lapso de diez segundos, este mismo proceso se repitió ingresando nodos de manera ordenada.

A continuación se presentan los cuadros con los datos de las pruebas según las estructuras de datos y orden de los nodos ingresados.

Cuadro I

LISTA ENLAZADA CON NODO CENTINELA

Nodos ingresados:	1 000 000.
Valor de los nodos ingresados:	Aleatorio entre 0 y 2 000 000.
Tiempo de búsquedas:	10 seg.
Valor de los nodos de búsqueda:	Aleatorio entre 0 y 2 000 000.
Búsquedas realizas en promedio:	1807.

En el cuadro I se observan los datos de la estructura de datos LISTA ENLAZADA CON NODO CENTINELA, ingresando los nodos con valores aleatorios.

Cuadro II

LISTA ENLAZADA CON NODO CENTINELA

Nodos ingresados:	1 000 000.
Valor de los nodos ingresados:	Ordenado de 0 a 1 000 000
Tiempo de búsquedas:	10 seg.
Valor de los nodos de búsqueda:	Aleatorio entre 0 y 2 000 000
Búsquedas realizas en promedio:	1820.

En el cuadro II se observan los datos de la estructura de datos LISTA ENLAZADA CON NODO CENTINELA, ingresando los nodos con valores ordenados.

Cuadro III

ARBOL DE BÚSQUEDA BINARIA

Nodos ingresados:	1 000 000.
Valor de los nodos ingresados:	Ordenado de 0 a 1 000 000.
Tiempo de búsquedas:	10 seg.
Valor de los nodos de búsqueda:	Aleatorio entre 0 y 2 000 000.
Búsquedas realizas en promedio:	1889.

En el cuadro III se observan los datos de la estructura de datos ARBOL DE BUSQUEDA BINARIA, ingresando los nodos con valores ordenados y utilizando la búsqueda iterativa.

Cuadro IV

ARBOL DE BÚSQUEDA BINARIA

Nodos ingresados:	1 000 000.
Valor de los nodos ingresados:	Aleatorio de 0 a 1 000 000.
Tiempo de búsquedas:	10 seg.
Valor de los nodos de búsqueda:	Aleatorio entre 0 y 2 000 000.
Búsquedas realizas en promedio:	8 617 906.

En el cuadro IV se observan los datos de la estructura de datos ARBOL DE BUSQUEDA BINARIA, ingresando los nodos con valores aleatorios y utilizando la búsqueda iterativa.

Cuadro V

ARBOL DE BÚSQUEDA BINARIA

Nodos ingresados:	1 000 000.
Valor de los nodos ingresados:	Aleatorio de 0 a 1 000 000.
Tiempo de búsquedas:	10 seg.
Valor de los nodos de búsqueda:	Aleatorio entre 0 y 2 000 000.
Búsquedas realizasen promedio:	7 949 087

En el cuadro V se observan los datos de la estructura de datos ARBOL DE BUSQUEDA BINARIA, ingresando los nodos con valores aleatorios y utilizando la búsqueda recursiva.

Cuadro VI

ARBOL ROJINEGRO

Nodos ingresados:	1 000 000.
Valor de los nodos ingresados:	Ordenado de 0 a 1 000 000.
Tiempo de búsquedas:	10 seg.
Valor de los nodos de búsqueda:	Aleatorio entre 0 y 2 000 000.
Búsquedas realizas en promedio:	22 664 994

En el cuadro VI se observan los datos de la estructura de datos ARBOL ROJINEGRO ingresando los nodos con valores ordenados y utilizando la búsqueda recursiva.

Cuadro VIII

ARBOL ROJINEGRO

Nodos ingresados:	1 000 000.
Valor de los nodos ingresados:	Aleatorio de 0 a 1 000 000.
Tiempo de búsquedas:	10 seg.
Valor de los nodos de búsqueda:	Aleatorio entre 0 y 2 000 000.
Búsquedas realizas en promedio:	14 528 994.

En el cuadro VIII se observan los datos de la estructura de datos ARBOL ROJINEGRO ingresando los nodos con valores aleatorios y utilizando la búsqueda iterativa.

Cuadro VII

ARBOL ROJINEGRO

Nodos ingresados:	1 000 000.
Valor de los nodos ingresados:	Aleatorio de 0 a 1 000 000.
Tiempo de búsquedas:	10 seg.
Valor de los nodos de búsqueda:	Aleatorio entre 0 y 2 000 000.
Búsquedas realizas en promedio:	24 137 381

En el cuadro VII se observan los datos de la estructura de datos ARBOL ROJINEGRO ingresando los nodos con valores aleatorios y utilizando la búsqueda recursiva.

Cuadro IX

ARBOL ROJINEGRO

Nodos ingresados:	1 000 000.
Valor de los nodos ingresados:	Ordenado de 0 a 1 000 000.
Tiempo de búsquedas:	10 seg.
Valor de los nodos de búsqueda:	Aleatorio entre 0 y 2 000 000.
Búsquedas realizas en promedio:	25 217 533.

En el cuadro IX se observan los datos de la estructura de datos ARBOL ROJINEGRO ingresando los nodos con valores ordenados y utilizando la búsqueda iterativa.

Cuadro X

TABLA DE DISPERSIÓN

Nodos ingresados:	1 000 000.
Valor de los nodos ingresados:	Aleatorio de 0 a 1 000 000.
Tiempo de búsquedas:	10 seg.
Valor de los nodos de búsqueda:	Aleatorio entre 0 y 2 000 000.
Búsquedas realizas en promedio:	19 305 999

En el cuadro X se observan los datos de la estructura de datos TABLA DE DISPERSIÓN ingresando los nodos con valores aleatorios y utilizando la búsqueda iterativa.

Cuadro XI

TABLA DE DISPERSIÓN

Nodos ingresados:	1 000 000.
Valor de los nodos ingresados:	Ordenado de 0 a 1 000 000.
Tiempo de búsquedas:	10 seg.
Valor de los nodos de búsqueda:	Aleatorio entre 0 y 2 000 000.
Búsquedas realizas en promedio:	19 905 809

En el cuadro X se observan los datos de la estructura de datos TABLA DE DISPERSIÓN ingresando los nodos con valores ordenados y utilizando la búsqueda iterativa.

III RESULTADOS:

-Comparación de lista enlazada ordenada con lista enlazada desordenada:

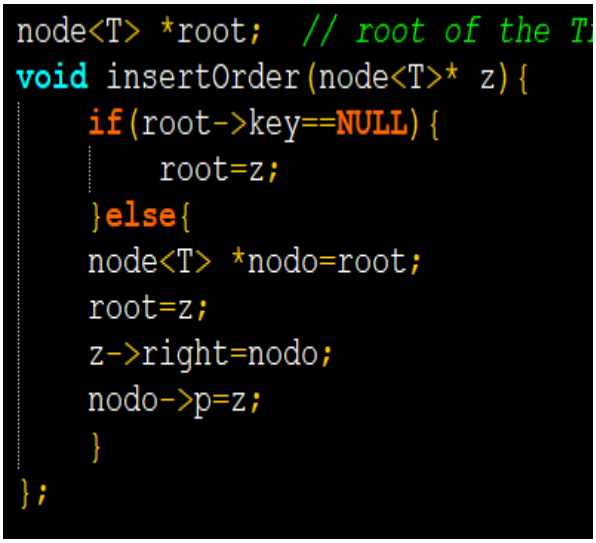
Como podemos apreciar en la lista enlazada con centinela el promedio de búsquedas según el orden de los nodos se mantiene muy parecido. Cabe resaltar que los números ingresados a buscar son en un intervalo entre cero y dos millones, esto va a crear que gran cantidad de datos jamás se encuentren y por lo tanto tengan que recorrer la totalidad de la lista. Por otro lado si el elemento se encuentra en la lista el promedio de nodos a recorrer es el mismo, sea que esté ordenada o no, ya que el nodo a buscar es aleatorio. Por lo tanto los datos obtenidos son los esperados en estas pruebas para la lista enlazada con nodo centinela.

-Comparación de lista enlazada con árbol de búsqueda binaria:

Es importante señalar que si se ingresan todos los nodos de manera ordenada a un árbol de búsqueda binaria, este sólo va a llenar sus nodos en una rama, por lo tanto se convierte en un estilo de lista. Es por esta razón que el árbol puede durar un tiempo considerable ingresando un millón de datos ordenados. Para solucionar un caso como este, se plantea un insertar en el árbol binario de manera que siempre se agregue en la raíz, de esta manera no hay que recorrer todo el árbol.

Imagen I

INSERTAR DE MANERA SEQUENCIAL



En la imagen I, se muestra el fragmento del código para ingresar los nodos al árbol de manera secuencial sin tener que recorrer todo el árbol.

Volviendo con la comparación, en el caso de ingresar los nodos de manera desordenada en el árbol binario, permite una mejor distribución de sus nodos, y por lo tanto la búsqueda de datos se realiza de una manera más eficiente. Podemos notar como sobrepasa por mucho la cantidad de búsquedas en el periodo de diez segundos a la lista enlaza.

Por último, en el caso de ingresar los datos de manera secuencial al árbol, esto nos va a generar que sólo una rama del árbol se desarrolle, como ya se mencionó, podemos notar que en promedio de búsquedas entre la lista enlazada con nodo centinela de manera secuencial y el árbol de búsqueda binaria de manera secuencial es prácticamente el mismo.

-Comparación árbol rojinegro con la lista enlazada, árbol de búsqueda binaria y tabla de dispersión:

Como podemos ver en el cuadro Vlll y lX la cantidad de búsquedas realizadas por el árbol rojinegro es elevado. La cantidad de búsquedas media del árbol rojinegro supera los veinte millones de búsquedas por lo que supera o triplica estructuras de datos como la lista enlazada y árbol de búsqueda binaria. En el caso del ingreso de los datos de forma ordenada el árbol rojinegro triplica las búsquedas del árbol binario, y en el caso del ingreso de datos de forma desordenada este lo triplica.

Esta cantidad de búsquedas y la comparación es la esperada, ya que en el caso del árbol rojinegro en ningún momento su altura va a ser diferente en ninguna de sus ramas, lo que le permite realizar búsquedas de una forma más ordenada y eficaz.

-Comparación tabla de dispersión con la lista enlazada, árbol de búsqueda binaria y árbol rojinegro:

La tabla de dispersión es la mejor estructura en el aspecto del ingreso de datos en orden o desorden, ya que como podemos ver en los cuadros X y Xl la cantidad de búsquedas es prácticamente igual.

Muchas de las estructuras de datos mencionadas anteriormente presentaban problemas en este aspecto, sin embargo esta estructura es muy estable y representa esta la principal diferencia con las demás

estructuras. En el caso del ingreso de datos desordenados la cantidad de búsquedas es menor a la del árbol rojinegro, pero en el caso del ingreso de los datos de forma ordenada es menor.

IV CONCLUSIONES:

Como conclusión se puede notar que un buen orden y desarrollo de los nodos en un árbol de búsqueda binaria mejora en una gran cantidad la eficacia de este, y de lo contrario su eficacia es casi nula.

Es importante mencionar que para la búsqueda de gran cantidad de elementos el árbol de búsqueda binaria presenta una mejor eficacia en comparación con la lista, pero en el caso de que la cantidad de datos sea relativamente pequeña la lista enlaza es una buena opción.

Con respecto a la tabla de dispersión y árbol rojinegro, este presenta una clara ventaja sobre las anteriores estructuras de datos. Principalmente la tabla de dispersión presenta una ventaja en el ingreso de datos ordenados, lo que puede servir como estructura segura para evitar casos extraños. En el caso de los árboles rojinegros son bastantes eficientes, y más que la tabla de dispersión, pero estos pueden fallar un poco o realizar más trabajo cuando los datos se ingresan de forma ordenada. Por lo tanto en conclusión y como mi opinión la estructura más segura y estable es la tabla de dispersión.

Imagen II

PRUEBAS EN EL ÁRBOL BINARIO BALANCEADO

```
for(int i=0; i<1000000; i++){
    int key=((rand()*rand())%2000001);
    nodo= new node<int>(key);
    arbol.treeInsert(nodo);
}

cont=0;
clock_t end=(clock() +(10*CLOCKS_PER_SEC));
while(clock()<end){
    int key=((rand()*rand())%2000001);
    arbol.iterativeTreeSearch(key);
    cont++;
}
cout<<"CANTIDAD DE BÚSQUEDAS:"<<cont<<endl;
```

Imagen III

PRUEBAS EN LISTA ENLAZADA CON NODO CENTINELA

```
for(int i=0; i<1000000; i++){
    int key=((rand()*rand())%2000001);
    nodo=new llnode<int>(key);
    lista.listInsert(nodo);
}

cont=0;
clock_t end=(clock() +(10*CLOCKS_PER_SEC));
while(clock()<end){
    int key=((rand()*rand())%2000001);
    lista.listSearch(key);
    cont++;
}
cout<<"CANTIDAD DE BÚSQUEDAS:"<<cont<<endl;
```

Imagen IV

PRUEBAS EN EL ÁRBOL ROJINEGRO

```
for(int i=0; i<1000000; i++){
    int key=((rand()*rand())%1000000);
    nodo= new rbnode<int>(key);
    arbol->treeInsert(nodo);
}

cont=0;
clock_t end=(clock() +(10*CLOCKS_PER_SEC));
while(clock()<end){
    int key=((rand()*rand())%2000001);
    arbol->treeSearch(key);
    cont++;
}
cout<<"ARBOL DESORDENADO:"<<cont<<endl;
```

Imagen V

PRUEBAS EN TABLA DE DISPERSIÓN

```
for(int i=0; i<1000000; i++){
    tabla->insert(i);
}

cont=0;
clock_t end=(clock() +(10*CLOCKS_PER_SEC));
while(clock()<end){
    int key=((rand()*rand())%2000001);
    tabla->search(key);
    cont++;
}
cout<<"TABLA ORDENADA: "<<cont<<endl;
```

V REFERENCIAS:

-Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C. Introduction to Algorithms. The MIT Press, 2001.

-Hernández, Z.J. y otros: Fundamentos de Estructuras de Datos. Soluciones en Ada, Java y C++, Thomson, 2005.

-Weiss, M.A.: Data Structures and Algorithm Analysis in C++, 4th Edition,Pearson/Addison Wesley, 2014.