***Resumen* – En este trabajo se realiza un análisis de tiempos de búsqueda, en este caso, se busca comparar que tan eficientes son las búsquedas en 4 estructuras de datos en concreto, *Lista Enlazada con nodo centinela*, *Árbol binario de búsqueda, Árbol Rojinegro* y *Hash table*.**

I. Introducción

En este trabajo se realizó un análisis de tiempos de búsqueda entre las estructuras de datos *Lista Enlazada con nodo centinela*, *Árbol binario de búsqueda, Árbol Rojinegro* y *Hash table* los cuales fueron implementados mediante un programa en el lenguaje de programación C++, se les tomará la cantidad de elementos que pueden encontrar en 1 segundo y cuantos no encuentran en 1 segundo.

II. Metodología

Para lograr lo propuesto, se utilizo el lenguaje de programación C++ para recrear las estructuras de datos propuestas en el libro *Introduction to Algorithms* de Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest y Clifford Stein, para obtener cuantos elementos se pueden encontrar en el lapso de 1 segundo y cuantos elementos no se pueden encontrar en el mismo lapso de tiempo.

Cuadro I
TIEMPO DE EJECUCIÓN DE LOS ALGORIMTOS

			Elementos Encontrados				Elementos No Encontrados			
			corridas				corridas			
Estructura de datos	Tam(millones)	Tiempo(s)	1	2	3	Prom.	1	2	3	Prom.
Lista Enlazada										
Search (aleatorio)	1	1	108	103	139	116.7	145	157	169	157.0
Search (secuencial)	1	1	155	167	149	157	124	118	144	128.7
		1								
Arbol Binario										
Search(aleatorio)	1	1	53500	49376	50096	50990.7	66082	64326	57565	62657.7
Search(secuencial	1	1	152	135	143	143.3	205508	220987	199726	208740.3
IterativeSearc(aleatorio)	1	1	52784	52423	49700	51635.7	59637	63069	58351	60352.3
IterativeSearch(aleatorio)	1	1	222	234	253	236.3	223903	228099	209031	220344.3
ArbolRN										
Search(aleatorio)	1	1	327865	321334	334961	328053.3	507239	494489	514521	505416.3
Search(secuencial	1	1	413411	407774	404703	408629.3	414877	408068	403931	408958.7
IterativeSearc(aleatorio)	1	1	326753	340182	339932	335622.3	504370	524276	523259	517301.7
IterativeSearch(aleatorio)	1	1	456252	448147	401548	435315.7	455642	448925	403420	435995.7
Hash	1									
Search(aleatorio)	1	1	666945	644950	675903	662599.3	1027016	991334	1043198	1020516.0
Search(secuencial	1	1	638663	762712	753162	718179	636492	761918	754958	717789.3

III. Resultados

La cantidad de elementos encontrados y no encontrados en un lapso de 1 segundo figuran en el cuadro 1, las estructuras de datos utilizaron el método *Search*() para encontrar tantos elementos aleatorios les sea posible en dicho lapso, en el caso del *árbol binario de búsqueda* y el *árbol rojinegro* también se consideró la versión Recursiva, independientemente de si se utilizó una inserción con números aleatorios o secuenciales, la *tabla hash* se muestra muy superior en cuanto a cantidades de búsquedas en un tiempo dado, en donde se pueden buscar casi 4580 veces más elementos en un mismo lapso de tiempo que con una lista enlazada, 13 veces más elementos que un árbol de búsqueda binaria y casi 2 veces más elementos que un árbol rojinegro.

Por otro lado, es interesante observar que la lista enlazada y la tabla hash se comportan mejor con elementos ordenados que con elementos desordenados, caso totalmente contrario con el

árbol binario y en cierta medida con el árbol rojinegro, por lo cual se obtuvo el resultado esperado.

IV. Conclusiones

A partir de los resultados obtenidos, se concluye que la *tabla hash* es muy superior en cuanto a búsquedas que una *lista enlazada con nodo centinela, un árbol binario de búsqueda* y es algo superior a un *árbol rojinegro*, por lo tanto, tanto teóricamente como prácticamente, la tabla hash obedece a su tipo O(1), el *árbol binario* se comportara como una *lista enlazada* cuando se le insertan elementos secuenciales, y el árbol rojinegro se comporta bastante bien sin importan si se le insertan elementos aleatorios o secuenciales, ya que este se mantiene balanceado.