2 Tarea Programada

Parte A Listas:

Lista con elementos aleatorios:

En esta primera parte se insertó en la lista con nodo centinela un millón de elementos de manera aleatoria cuyo rango va de [0, 2*millones*).

Primero realizamos búsquedas en la lista seleccionando los elementos a buscar en el rango de [0, 2*millones*) y registrando el número de búsquedas realizadas fallidas o exitosas en un lapso de 10 segundos el cual dio como resultados**:**

Cuadro 1

Búsquedas exitosas en un tiempo de 10 segundos	**1177**
Búsquedas fallidas en un tiempo de 10 segundos	26
Búsquedas totales:	1203

Lista con elementos secuenciales:

Por otra parte, al insertar las llaves 0, 1, ..., $n-1$(n=1 millón), en ese orden. Y al realizar las búsquedas en el rango de [0, 2*millones*) y registrando el número de búsquedas realizadas fallidas y exitosas en un lapso de 10 segundos dio como resultado:

Cuadro 2:

Búsquedas exitosas en un tiempo de 10 segundos	**839**
Búsquedas fallidas en un tiempo de 10 segundos	768
Búsquedas totales:	1607

2 Tarea Programada

Cuadro 3:

Tiempos	***Búsquedas secuenciales***	***Búsquedas aleatorias***
Búsquedas exitosas en un tiempo de 10 segundos	839	1177
Búsquedas fallidas en un tiempo de 10 segundos	768	26
Búsquedas totales:	1607	1203

En la primera parte donde se insertaron y buscaron elementos de manera aleatoria se denota un sesgo muy grande entre las búsquedas fallidas y las exitosas, pero en la parte donde se insertaron elementos de manera secuencial podemos ver que el número de búsquedas fallidas y exitosas son muy similares esto se da debido a que en la parte de búsquedas secuenciales se buscan elementos en el rango de 0 a 2n y como se insertaron elementos de 0 a n-1 cabe una gran posibilidad que se generen números mayores a 1 millón y esto va a generar que el número de búsquedas fallidas incremente, por lo que el **cuadro 3** se evidencia que las fallidas en las búsquedas secuenciales es más del doble que las de las búsquedas aleatorias y por lo explicado anteriormente esto corresponde con los resultados esperados.

2 Tarea Programada

Parte B Árboles:

Árbol con elementos aleatorios:

En esta parte se insertó en el árbol un millón de elementos de manera aleatoria cuyo rango va de [0, 2*millones*).

Primero realizamos búsquedas en el árbol seleccionando los elementos a buscar en el rango de [0, 2*millones*) y registrando el número de búsquedas realizadas fallidas o exitosas en un lapso de 10 segundos el cual dio como resultados**:**

Cuadro 4:

Búsquedas exitosas en un tiempo de 10 segundos	**54381**
Búsquedas fallidas en un tiempo de 10 segundos	**3245295**
Búsquedas totales:	**3299676**

Cuadro 5:

Árbol con elementos secuenciales:

Por otra parte, al insertar las llaves 0, 1, ..., $n-1$(n=1 millón), en ese orden. Y al realizar las búsquedas en el rango de [0, 2*millones*) y registrando el número de búsquedas realizadas fallidas y exitosas en un lapso de 10 segundos dio como resultado:

Búsquedas exitosas en un tiempo de 10 segundos	**844**
Búsquedas fallidas en un tiempo de 10 segundos	**831**
Búsquedas totales:	**1675**

2 Tarea Programada

Cuadro 6:

Tiempos	***Búsquedas secuenciales***	***Búsquedas aleatorias***
Búsquedas exitosas en un tiempo de 10 segundos	844	54381
Búsquedas fallidas en un tiempo de 10 segundos	831	3245295
Búsquedas totales:	1675	3299676

En el cuadro 6 podemos denotar un gran sesgo entre las búsquedas aleatorias y secuenciales, esto lo podemos explicar ya que el árbol al insertarle elementos de manera secuencial se está incurriendo al peor caso de árboles de búsqueda binarios, ya que todos sus elementos quedarían en las ramas derechas por lo que realizar búsquedas en este peor caso pueden llegar a tardar demasiado, podemos ver que se evidencia lo esperado ya que buscar en el peor caso de árbol puede tardar mucho, en cambio en la inserción aleatoria esperaríamos que el árbol no quede en su peor forma, por lo que se da una mayor cantidad de búsquedas.

2 Tarea Programada

Cuadro 7

Numero de búsquedas	Listas	Árboles
Aleatorios	1203	3299676
Secuenciales	1607	1675

Del cuadro 7 podemos notar como el árbol de búsqueda binario fue sustancialmente mejor que las listas en las búsquedas aleatorias ya que realizando búsquedas de manera aleatoria produjo más del doble que la lista y esto se debe que en promedio al insertar de manera aleatoria elementos en un árbol se espera que no quede en su peor caso sino en su caso promedio.

También podemos denotar que en el caso de búsquedas secuenciales no se evidencia un gran sesgo entre árboles y listas y esto se debe a que el árbol al insertarle elementos de manera secuencial se está incurriendo al peor caso de árboles de búsqueda binarios, ya que todos sus elementos quedarían en las ramas derechas por lo que se comportaría como una lista por esto es que se producen una cantidad de búsquedas similares en ambas estructuras.

Parte C:

Árboles rojinegros con elementos aleatorios:

En esta parte se insertó en el árbol rojinegro un millón de elementos de manera aleatoria cuyo rango va de [0*, 2millones*).

Primero realizamos búsquedas en el árbol seleccionando los elementos a buscar en el rango de [0*, 2millones*) y registrando el número de búsquedas realizadas fallidas o exitosas en un lapso de 10 segundos el cual dio como resultados**:**

2 Tarea Programada

Cuadro 8:

Búsquedas exitosas en un tiempo de 10 segundos	**6924246**
Búsquedas fallidas en un tiempo de 10 segundos	**115766**
Búsquedas totales:	**7040012**

Árboles rojinegros con elementos secuenciales:

Al insertar las llaves 0, 1, ..., $n-1$(n=1 millón), en ese orden. Y al realizar las búsquedas en el rango de [0, 2*millones*) y registrando el número de búsquedas realizadas fallidas y exitosas en un lapso de 10 segundos dio como resultado:

Cuadro 9:

Búsquedas exitosas en un tiempo de 10 segundos	**7863448**
Búsquedas fallidas en un tiempo de 10 segundos	**7864638**
Búsquedas totales:	**15728086**

2 Tarea Programada

Cuadro 10:

Tiempos	***Búsquedas secuenciales***	***Búsquedas aleatorias***
Búsquedas exitosas en un tiempo de 10 segundos	7863448	6924246
Búsquedas fallidas en un tiempo de 10 segundos	**7864638**	**115766**
Búsquedas totales:	**15728086**	**7040012**

Del **cuadro 10** podemos notar como el árbol rojinegro fue por mucho mejor que las estructuras listas y ABB en las búsquedas aleatorias ya que realizando búsquedas de manera aleatoria produjo más del doble que la lista y el ABB.

También podemos denotar que en el caso de búsquedas aleatorias se evidencia un gran sesgo entre árboles rojinegros con respecto a listas y ABB.

También podemos denotar que en el caso de búsquedas secuenciales se evidencia un gran sesgo entre árboles rojinegros con respecto a listas y ABB esto se debe a que al árbol y a la lista al insertarle elementos de manera secuencial se da el peor caso de estas estructuras ya que si yo busco el último elemento tendré que recorrer toda la estructura para encontrarlo, por otro lado, el árbol rojinegro esta balanceado por lo que su tiempo de búsqueda es muy rápido**(O(logn)).** por lo que podemos concluir que los resultados si corresponden a lo esperado.

2 Tarea Programada

Parte D:

Tablas Hash con elementos aleatorios:

En esta parte se insertó en la tabla hash un millón de elementos de manera aleatoria cuyo rango va de [0, 2*millones*).

Primero realizamos búsquedas en el árbol seleccionando los elementos a buscar en el rango de [0, 2*millones*) y registrando el número de búsquedas realizadas fallidas o exitosas en un lapso de 10 segundos el cual dio como resultados**:**

Cuadro 11

Búsquedas exitosas en un tiempo de 10 segundos	**882515**
Búsquedas fallidas en un tiempo de 10 segundos	**52976392**
Búsquedas totales:	**53858907**

Tablas H con elementos secuenciales:

Al insertar las llaves 0, 1, ..., $n-1$(n=1 millón) en la tabla, en ese orden. Y al realizar las búsquedas en el rango de [0, 2*millones*) y registrando el número de búsquedas realizadas fallidas y exitosas en un lapso de 10 segundos dio como resultado:

Cuadro 12

Búsquedas exitosas en un tiempo de 10 segundos	**28065327**
Búsquedas fallidas en un tiempo de 10 segundos	**28038777**
Búsquedas totales:	**56104104**

2 Tarea Programada

Cuadro 13

Tiempos	***Búsquedas secuenciales***	***Búsquedas aleatorias***
Búsquedas exitosas en un tiempo de 10 segundos	28065327	882515
Búsquedas fallidas en un tiempo de 10 segundos	**28038777**	**52976392**
Búsquedas totales:	**56104104**	**53858907**

Podemos notar del **cuadro 13** que en el caso de búsquedas aleatorias se evidencia una gran diferencia entre tablas hash con respecto a las otras estructuras esto se debe a que al árbol y a la lista, si yo busco el último elemento tendré que recorrer toda la estructura para encontrarlo, por otro lado, el árbol rojinegro esta balanceado por lo que su tiempo de búsqueda es muy rápido**(O(logn))** pero esto comparado con el tiempo de ejecución promedio de las tablas hash **θ(1)** hace que se produzca este gran sesgo**.** Y como esperábamos que al insertar en el hash elementos de manera aleatoria no quede en su peor caso **θ(n)** sino en el promedio **θ(1)** y como se esperábamos el hash realizo más búsquedas.

Por otro lado, al insertar de manera secuencial estamos en el mejor caso de la tabla hash **θ(1)**, por lo que era de esperar que esta sobrepasara en búsquedas a las otras estructuras. Por lo anterior concluimos que los resultados corresponden a lo esperado.