Método de ingeniería laboratorio n°3

Fase 1: Identificación de necesidades y síntomas

-La solución al problema debe garantizar efectividad y rapidez en la búsqueda.
-La simulación debe mostrar gráficamente los precios de una acción y el mercado de divisas.
-Se debe buscar una data set que proporcione la información necesaria para el previo análisis.

Definición del problema:
-Una compañía desarrolladora de software de mediana escala, pero de mucho futuro, se le ha solicitado la implementación de la aplicación de la BVC, para el manejo de información de gran tamaño.

Requerimientos funcionales:

-R1. Consultar el precio más alto. El programa debe consultar el precio más alto de una acción o un mercado de divisas en un rango de tiempo. Es decir, el precio más alto de la misma, recibiendo como entrada una fecha inicial y una fecha final.
-R2. Consultar el precio más bajo. El programa debe consultar el precio más bajo de una acción o un mercado de divisas en un rango de tiempo. Es decir, el precio más alto de la misma recibiendo como entrada una fecha inicial y una fecha final.
-R3.Consultar mayor crecimiento. El programa debe consultar el periodo de tiempo donde una acción / mercado de divisas tuvo su mayor crecimiento.
-R4. Mostrar gráfica.El programa debe mostrar una gráfica del estado de los precios de una acción / mercado de divisas. En la gráfica debe ser posible agregar hasta un máximo de 3 acciones / mercados de divisas en donde cada indicador de gráfica deberá tener un color diferente.
-R5. Consultar qué divisas superan un valor. Consultar cuáles acciones / Mercado de divisas superan un valor en un rango de tiempo.
-R6. Consultar las 3 acciones con mayor crecimiento.El programa debe consultar cuales son las 3 acciones / Mercados que presentaron mayor crecimiento en un rango de tiempo.
-R7 Medir tiempo de diferencia. Se debe especificar el tiempo que tarda en consultar el mercado de bitCoins en diferencia a los otros.

Fase 2: Recopilación de información:

Divisa: Denominamos divisa a toda moneda extranjera, es decir, a las monedas oficiales distintas de la moneda legal en el propio país.Mientras que la moneda local es la moneda de referencia de un país, la moneda local y oficial de un territorio. Se considera divisa a todas aquellas monedas distintas de las del país de origen.

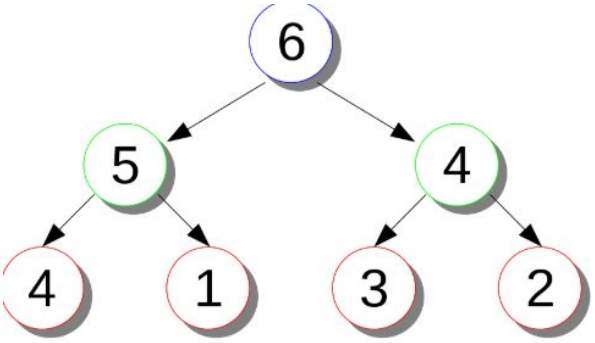
Acción: Una acción es un activo financiero que representa una parte alícuota del capital social de una Sociedad Anónima (S.A). Mediante la compra de una acción nos convertimos en dueños de esta sociedad. Al adquirir acciones se reciben unos derechos sobre la empresa y se obtiene la categoría de socio. De esta forma el comprador de la acción se convierte en dueño de la empresa en una proporción acorde a las acciones que hayamos comprado.

Recopilación de dataSet: Para la recolección del dataSet, se pensó en la alternativa de usar los archivos suministrados por la página https://es.investing.com/ pues esta página nos proporciona una cantidad de mercados que podríamos usar para el laboratorio.Por otra parte, el laboratorio nos ofrece una serie de archivos que también podrían ser útiles en la elaboración del mismo.

Fase 3: Búsqueda de soluciones creativas.

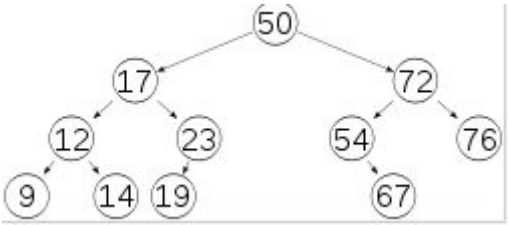
Alternativa 1:

Como grupo se pensó usar un heap para controlar e ingresar los datos recibidos por los archivos planos, teniendo en cuenta que los objetos en un heap siempre se crean en el espacio de almacenamiento dinámico y las referencias a estos objetos se almacenan en la memoria de la pila. Estos objetos tienen acceso global y se puede acceder desde cualquier lugar de la aplicación.

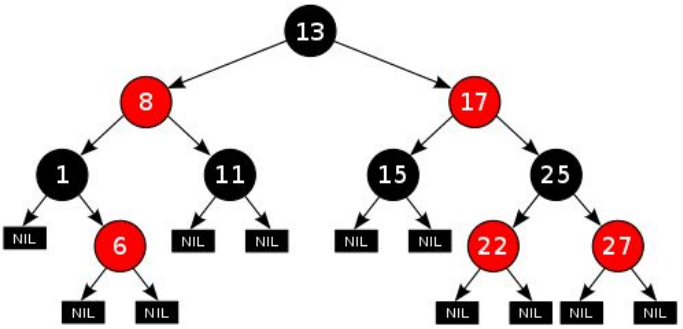


6	5	4	4	1	3	2

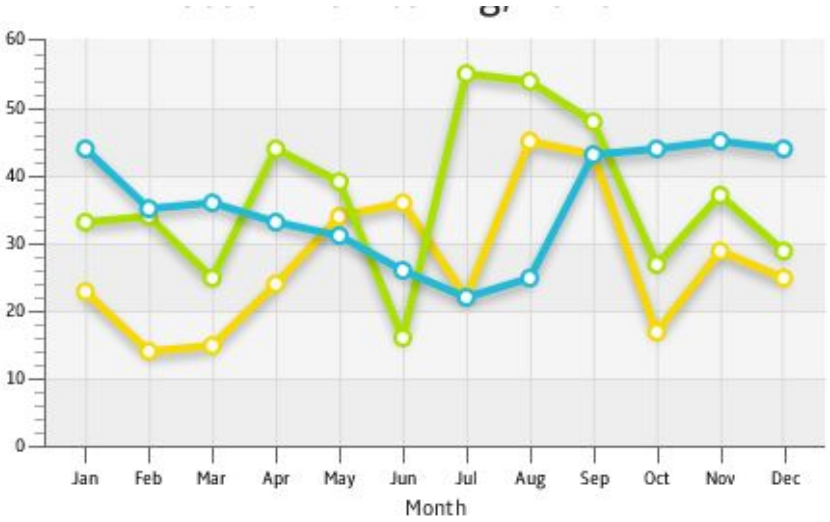
Para el manejo de divisas se usará un Árbol Binario AVL , Los árboles AVL están siempre equilibrados de tal modo que para todos los nodos, la altura de la rama izquierda no difiere en más de una unidad de la altura de la rama derecha o viceversa. Gracias a esta forma de equilibrio (o balanceo), la complejidad de una búsqueda en uno de estos árboles se mantiene siempre en orden de complejidad O(log n). Como para la inserción, como para la búsqueda de datos.



Para el manejo de acciones, se usará un Árbol RojiNegro ya que en los árboles rojo-negro es un tipo especial de árbol binario usado en informática para organizar información compuesta por datos comparables (por ejemplo, números).



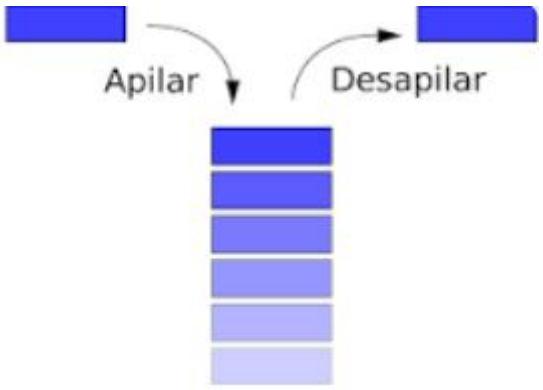
Para graficar los datos, se usará en esta alternativa la clase de JavaFx “LineChart”, esta clase, nos facilita métodos para agregar una gran cantidad de gráficas en un solo plano



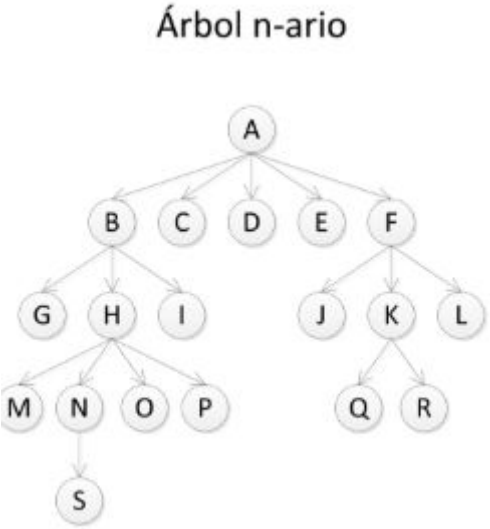
Para esta alternativa, los archivos planos a usar serán:
https://drive.google.com/drive/folders/12r8W6imtnkp38HnjrVY2s-sL7Jd8We9s

Alternativa 2:

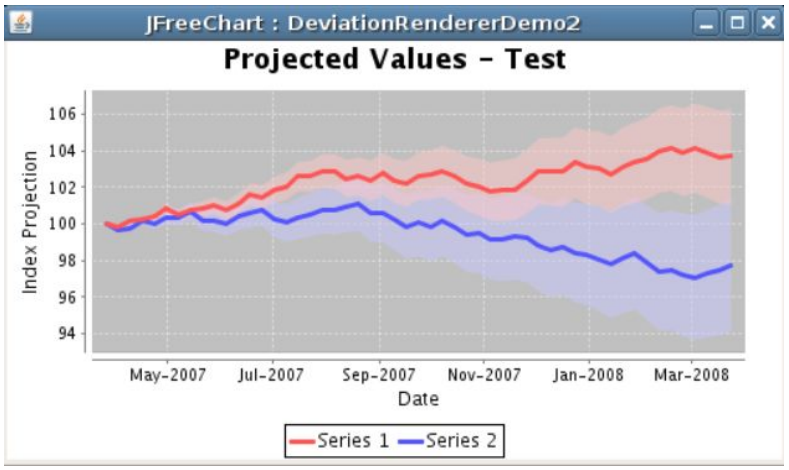
Para almacenar los datos de los dataSets se usará una pila, lo haremos para separar las divisas y las acciones en dos pilas diferentes.



Luego para procesar la información usaremos Árboles n-arios,corresponde a la generalización de un árbol binario de cuyos nodos pueden desprenderse múltiples árboles binarios facilitando el almacenamiento y procesamiento de grandes datos, a causa de que cada nodo, puede tener una gran cantidad de datos.



Para la gráfica se usará la libreria JfreeChart esta permite crear distintas graficas y formas que pueden ayudar a la interpretación



Para los DataSets en esta alternativa se usarán los datos dados por la página https://es.investing.com/, investing tiene los datos de una gran cantidad de mercados, que pueden ser útiles.

Fase 4: Transición de ideas a diseños preliminares.

Revisión de las alternativas

Alternativa 1:
-Los montículos son de un tamaño fijo, esto puede llegar a tener problemas si se desean agregar una cantidad excesiva de datos.

Alternativa 2:
-Los dataSets dados por Investing.com tienen información que no puede llegar a ser útil.
-La búsqueda de Árboles n-arios puede llegar a tener una complejidad de O(n)

Fase 5: Evaluación de la mejor solución:

Criterios:

- Criterio A. Eficiencia. Se prefiere una solución con mejor eficiencia que las otras consideradas. La
  eficiencia puede ser:
  [4] Constante
  [3] Mayor a constante
  [2] Logarítmica
  [1] Lineal

- Criterio B. Compatibilidad. Se prefiere una solución que sea compatible con todos los sistemas a evaluar.

[3] Compatible con todos los sistemas
[2] Compatible con algunos sistemas
[1] No compatible.

- Criterio C. Eficiencia en la búsqueda. Se prefiere una solución que sea eficiente en el momento de buscar y seleccionar datos.La eficiencia puede ser.
  [4] Constante
  [3] Mayor a constante
  [2] Logarítmica
  [1] Lineal

- Criterio D. Información DataSets. Se prefiere una solución que no contenga información irrelevante. La información puede ser.
  [3] Relevante
  [2]Poco relevante
  [1]Irrelevante.

- Criterio E. Claridad en la gráfica. Se prefiere una solución que muestre la información de manera clara y que permita tener más de una gráfica.
  [2]Muestra todas las gráficas ingresadas
  [1]Solo muestra una gráfica.

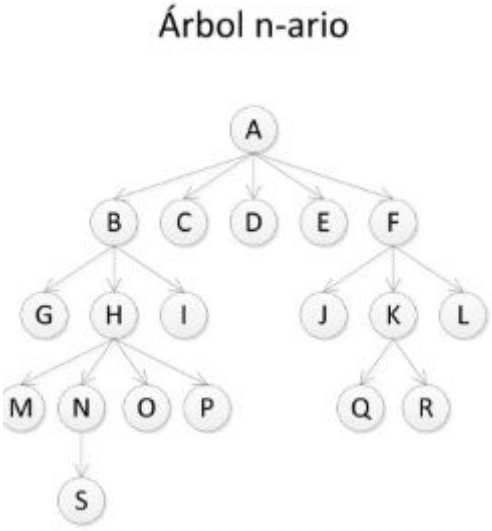
Evaluación:

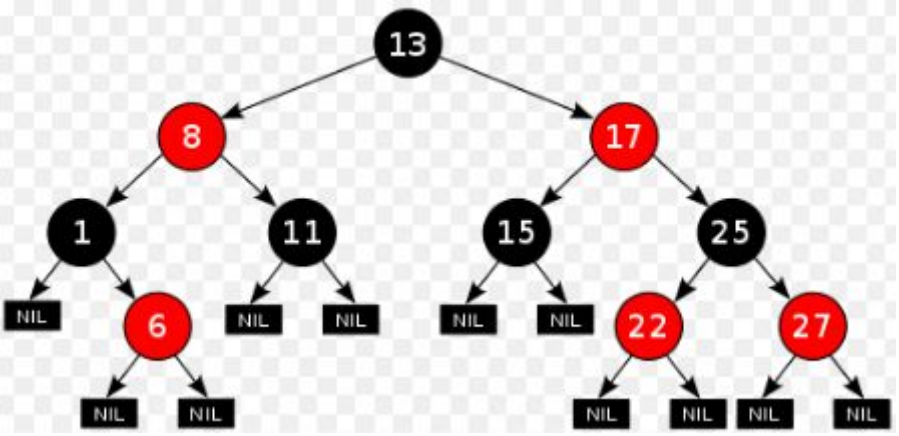
Evaluando los criterios anteriores en las alternativas que se mantienen, obtenemos la siguiente tabla:

	Criterio A	Criterio B	Criterio C	Criterio D	Criterio E	Total
Alternativa 1	2	3	2	3	2	12
Alternativa 2	1	2	1	2	2	6

Selección
De acuerdo con la evaluación anterior se debe seleccionar la Alternativa 1, ya que obtuvo la mayor puntuación de acuerdo con los criterios definidos. Se debe tener en cuenta que hay que hacer un manejo adecuado del criterio en el cual la alternativa fue peor evaluada que la otra alternativa.

Fase 6: Preparación de informes y especificaciones

TAD ARBOL N-ARIO		
		
{inv:Cada nodo posee un número indeterminado de hijos, se llena de derecha a izquierda}		
operaciones :		
• Inserción	N-ARIO x Nodo	→ N-ARIO
• eliminar	N-ARIO x Nodo	→ NULL
• Búsqueda	N-ARIO x Nodo	→ Nodo
• reccorridoIn	N-ARIO	→ N-ARIO
• reccoridoPre	N-ARIO	→ N-ARIO
• recorridoPos	N-ARIO	→ N-ARIO

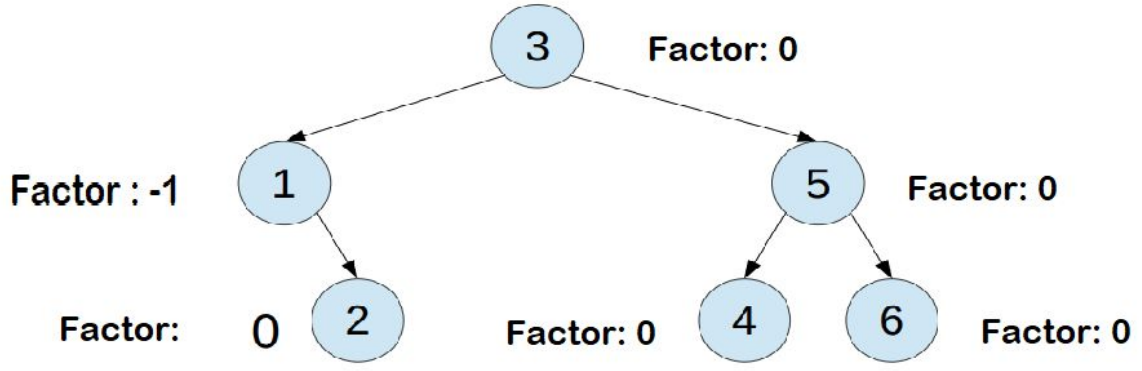
TAD ÁRBOL ROJO Y NEGRO


{inv: Cada nodo tiene estado rojo o negro
Raíz del árbol siempre es negra.
Los hijos de un nodo rojo son negros.
Las hojas del árbol son negras.
Todas las ramas del árbol (caminos desde la raíz hasta una hoja) tienen el mismo número de nodos negros .
Para todo nodo interno (que no sea hoja, es decir nulo), la altura negra de su hijo izquierdo es igual a la altura negra de su hijo derecho.
Nodo nuevo que entra al árbol, siempre entra Rojo
todo nodo hoja es nulo}

Operaciones :

- Insertar RyN x Nodo → RyN
- Eliminar RyN → Null
- darPreorden RyN → Nodo x Nodo
- Buscar RyN x Nodo → Nodo RyN
- darMinimo RyN → Nodo
- darMaximo RyN → Nodo

TAD ÁRBOL AVL

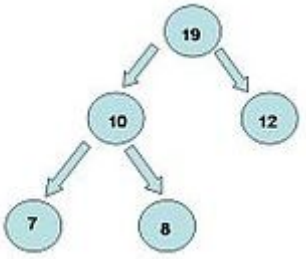


{inv : Existe un factor de diferencia que para cada nodo del árbol de estar -1<Nodo<1.}

Operaciones :

• Insertar	AVL x Nodo	→ AVL
• Eliminar	AVL x Nodo	→ Null
• Buscar	AVL x Nodo	→ Nodo AVL
• RotarDerecha	AVL x Nodo Factor No cumple	→AVL
• RotarIzquierda	AVL x Nodo Factor No cumple	→ AVL
• DobleRotarDerecha	AVL x Nodo Factor No cumple	→ AVL
• DobleRotarIzquierda	AVL x Nodo Factor No cumple	→ AVL

TAD MAX-HEAP

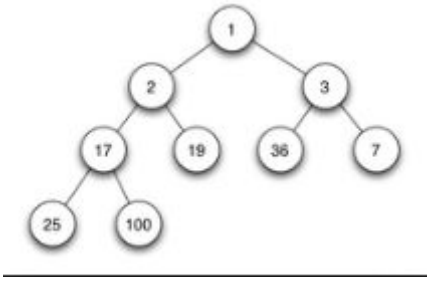


{inv: cada nodo padre tiene un valor mayor que el de cualquiera de sus nodos hijos, debe cumplir con árbol completo cuando todos los niveles están llenos, se llena desde la izquierda hacia la derecha, el nodo raíz es el mayor elemento de todo el heap}

Operaciones:

• Insertar	MAX-HEAP x VALUE	→ MAX-HEAP
• Eliminar	MAX-HEAP x VALUE	→ Null
• Buscar	MAX-HEAP x VALUE	→ MAX-HEAP x VALUE
• Heap-fyUp	MAX-HEAP	→ MAX-HEAP
• Heap-fyDown	MAX-HEAP	→ MAX-HEAP

TAD MIN-HEAP



{inv: cada nodo padre tiene un valor menor que el de cualquiera de sus nodos hijos, debe

cumplir con árbol completo cuando todos los niveles están llenos, se llena desde la izquierda hacia la derecha, el nodo raíz es el menor elemento de todo el heap}
Operaciones: ● Insertar MAX-HEAP x VALUE → MAX-HEAP ● Eliminar MAX-HEAP x VALUE → Null ● Buscar MAX-HEAP x VALUE → MAX-HEAP x VALUE ● Heap-fyUp MAX-HEAP → MAX-HEAP ● Heap-fyDown MAX-HEAP → MAX-HEAP

Escenario de pruebas para las estructuras implementadas

AVL

P1. Probar que el método insertar(T elemento) agrega un nodo al arbol AVL

P2 Probar que el método eliminar (T elemento) ELIMINA un nodo del arbol AVL

P3 Probar que el método Busacar(T elemento) busca un nodo en el arbol AVL

Clase	**Método**	**Escenario**	**Valores Entrada**	**Resultado**

ArbolAVL	**P1 insertar(T elemento)**	escenario1
		arbol = instanciarArbol( );
		verificador = instanciarVerificador( );
		datos = 20000;
		escenario2
		elem = (int) (Math.random( ) * datos)
			Elemento T que será insertado	**Se insertó correctamente el nuevo nodo.**

	P2 eliminar(T elemento)	escenario1
		arbol = instanciarArbol( );
		verificador = instanciarVerificador( );
		datos = 20000;
		escenario2
		elem = (int) (Math.random( ) * datos)
			Elemento T que será eliminado	**Se eliminó correctamente el nodo**

	P3 Buscar (T Elemento)	escenario1 arbol =	Elemento T que será	**Se encontró el elemento nodo**

		instanciarArbol( ); verificador = instanciarVerificador( ); datos = 20000; escenario2 elem = (int) (Math.random( ) * datos) elementoAbuscar=15	encontrado	

Rojo y negro

P1. Probar que el método insertar(T elemento) agrega un nodo al arbol Rojo y Negro

P2 Probar que el método eliminar (T elemento) ELIMINA un nodo del arbol Rojo y Negro

P3 Probar que el método Busacar(T elemento) busca un nodo en el arbol Rojo y Negro

Clase	**Método**	**Escenario**	**Valores Entrada**	**Resultado**
ArbolRojoNegro	**P1 insertar(T elemento)**	arbol = instanciarArbol( ); verificador = instanciarVerificador( ); datos = 20000;	Elemento T que será insertado	**Se insertó correctamente el nuevo nodo.**
	P2 eliminar(T elemento)	escenario1 arbol = instanciarArbol( ); verificador = instanciarVerificador( ); datos = 20000; escenario2 elem = (int) (Math.random( ) * datos)	Elemento T que será eliminado	**Se eliminó correctamente el nodo**
	P3 Buscar (T	escenario1	Elemento T	**Se encontró el**

	Elemento)	arbol = instanciarArbol( ); verificador = instanciarVerificador( ); datos = 20000; escenario2 elem = (int) (Math.random( ) * datos) elementoAbuscar=15	que será encontrado	**elemento nodo**

Fase 7: Implementación del diseño

Lista de tareas a implementar.

- Consultar el precio más alto.
- Consultar el precio más bajo.
- Consultar mayor crecimiento.
- Mostrar gráfica.
- Consultar qué divisas superan un valor.
- Consultar las 3 acciones con mayor crecimiento.
- Medir tiempo de diferencia.

Nombre	Consultar el precio más alto	
Descripción	Este método calcula el precio más alto de una acción o mercado de divisas en un rango de tiempo	
Entrada	rango de tiempo,nombre de la divisa o acción	
salida	precio más alto	

Nombre	Consultar el precio más bajo	
Descripción	Este método calcula el precio más bajo de una acción o mercado de divisas en	

	un rango de tiempo	
Entrada	rango de tiempo,nombre de la divisa o acción	
salida	precio más bajo	

Nombre	Consultar mayor crecimiento	
Descripción	Este método calcula el el periodo de tiempo donde una acción o divisa tuvo su mayor crecimiento	
Entrada	Nombre de la acción o divisa	
salida	Periodo de tiempo	

Nombre	Mostra grafica	
Descripción	Este método muestra la gráfica de una divisa en un rango de tiempo.	
Entrada	rango de tiempo,nombre de la divisa o acción	
salida		

.

Nombre	Consultar qué divisas superan un valor	
Descripción	Este método consulta qué acciones o divisas superan un valor en un rango de tiempo	
Entrada	rango de tiempo,valor	
salida	nombre de las divisas	

Nombre	Consultar las 3 acciones con mayor crecimiento	
Descripción	Este método muestra las 3 acciones o divisas con mayor crecimiento en un rango de tiempo	
Entrada	rango de tiempo	
salida	nombre de las acciones	

Nombre	Medir tiempo de diferencia	
Descripción	Este método mide el tiempo de diferencia entre el mercado de bitCoin en comparación con los otros mercados	
Entrada		
salida	tiempo que tarda en buscar	