Proyecto C++

Proyecto primer trimestre de la asignatura de programación.
Autor: Agustín Álvarez Fijo.
Revisor: Don José Antonio Rufo.

En este proyecto trabajaremos con datos relacionados con la criptomoneda llamada Bitcoin. Los datos están recogidos del fichero "*Bitcoin.csv*" en la carpeta data del proyecto, el fichero está en formato CSV, con los datos separados por comas.

Cada línea consta en primer lugar de la fecha (date), en segundo lugar, la cantidad de apertura(open), seguido de los valores "high' y "low' que representan los valores más alto y más bajo respectivamente alcanzados, a estos valores les sigue el precio de cierre en la fecha indicada previamente. En penúltimo lugar está el precio de cierre ajustado (adjusted closing price) que es el valor en efectivo del último precio negociado antes del cierre del mercado y sirve para que los inversores puedan tener un registro preciso del rendimiento de las acciones. Por último, está el volumen, que corresponde a todas las compras y ventas hechas durante el período de tiempo, el volumen será mayor cuantas más transacciones se hayan hecho.

En el proyecto analizaremos la evolución del Bitcoin durante los años 2014-2019:

	A	B	C	D	E	F	G
1	Date	Open	High	Low	Close	Adj Close	Volume
2	17/09/2014	359546204	361468506	351586884	355957367	355957367	16389165
3	18/09/2014	355588409	355505402	319789459	328539368	328539368	26691849
4	19/09/2014	328278503	330936707	298921021	307761139	307761139	29560102
5	20/09/2014	307665253	329978180	303931244	318758972	318758972	28736825

Aquí podemos ver las primeras líneas del CSV y su distribución previamente explicada, con estos datos estudiaremos la evolución del Bitcoin a lo largo de los años y algunos de los factores que han hecho que se convierta en la criptomoneda más valiosa del mercado actualmente, como el promedio de volumen necesario para tales ganancias, o los máximos y mínimos de diferentes factores que afectan a su crecimiento.

Requisitos Funcionales:

Los requisitos funcionales están especificados para cada función del programa.

1. Función 1: Lectura de datos
    - 1.1: El sistema debe ser capaz de leer los datos del archivo "Bitcoin.csv".
    - 1.2: Los datos deben almacenarse en una estructura de datos que contenga información como date, apertura, high, low, close, adj_close y volumen.

2. Función 2: Cálculo de media volumen
    - 2.1: El sistema debe calcular la media de los volúmenes entre los años 2014 y 2016.
    - 2.2: Solo se deben considerar los registros cuyas fechas están dentro del rango especificado.

3. Función 3: Selección del máximo volumen en el año deseado
    - 3.1: El sistema debe identificar y devolver el registro con el volumen máximo del año deseado.
    - 3.2: Se deben proporcionar todos los campos presentes en la línea correspondiente a ese máximo volumen.

4. Función 4: Lista ordenada de los Close (precios de cierre)
    - 4.1: El sistema debe generar una lista ordenada de mayor a menor de los precios de cierre ('close').
    - 4.2: La lista debe tener un tamaño ‘n’ elegido por el usuario.

5. Función 5: Agrupación de Registros por los Open (precio de apertura)
    - 5.1: El sistema debe crear un diccionario que agrupe los registros según el valor 'open' en un año específico y decidido por el usuario.
    - 5.2: Cada 'open' debe tener asociada una lista que contenga los valores 'close' y 'volumen'.

6. Función 6: Agrupación ordenada de los Adjusted close price
    - 6.1: El sistema debe crear un diccionario que agrupe los Adjusted close price y asocie cada precio con una lista formada por el volumen y la fecha.
    - 6.2: Este diccionario debe estar ordenado por el campo del volumen.

Requisitos NO Funcionales:

Los requisitos no funcionales están especificados en general de todo el código.

1. Rendimiento
    - 1.1: Debe ser capaz de manejar grandes conjuntos de datos de manera eficiente.
    - 1.2: El tiempo de respuesta para operaciones no debe exceder un límite de tiempo razonable.
2. Errores
    - 2.1: Su uso debe ser intuitivo para el usuario
      .
    - 2.2: Se deben proporcionar mensajes de error claros en caso de problemas.
3. Compatibilidad
    - 3.1: El sistema debe ser compatible con diferentes sistemas operativos.
4. Mantenibilidad
    - 4.1: El código debe estar bien documentado para facilitar futuras actualizaciones y modificaciones.
    - 4.2: Se debe seguir un estilo de codificación consistente en todo el proyecto.

Funciones Auxiliares usadas:

- get_time() →La función get_time() la utilizo para extraer valores de tiempo desde un string y almacenarlos en un objeto tm
- vector<> → para poder tener un almacenamiento dinámico de datos y la posibilidad de usar funciones como sort(), reverse() o find()
- Istringstream() → método para dividir la cadena (la línea) en tokens le indico el nombre del token que estoy leyendo y el delimitador que indica el final del token, que en este caso es una coma.
- StringToDouble() → para convertir de string a double
- StringToll() → para convertir de string a Long Long
- push_back() → para que vaya poniendo cada registro al final del vector de registros
- make_tuple() → para crear la tupla dentro de mi vector

- Sort() → para ordenar mi vector de menor a mayor
- Reverse() → para invertir las posiciones de mi vector
- Begin() → para acceder a la primera posición del vector
- End() → para acceder a la última posición del vector
- Find() → para buscar si existe un elemento en mi mapa
- Get<> → para acceder a una posición concreta de mi tupla o vector
- ShellExecuteA() → para abrir una URL
- Sleep() → para suspender un proceso durante unos segundos

Cómo bibliografía mucho stackoverflow y distintos foros de auténticos frikardos y un poquito de chat gpt para que me ayudara a entender el funcionamiento interno de alguna función.