Sintaxis y semántica de los lenguajes

Trabajo Práctico Nº 1

Web Scraping Financiero

Alumno: Nicolás Acosta

Docente: Ing. Pablo Damián Mendez

Fecha de entrega: 16/07/2020

Resumen:

La estrategia consistió principalmente en descargar la página utilizando html. Recorrer el archivo hasta encontrar la tabla e ir cargando cada línea con información en un nodo de una lista del struct accion. Una vez que se tiene la lista con la información, analizar los datos de acuerdo a las necesidades de cada reporte. En el A, se muestran por pantalla; en el B, se vuelcan los datos en una tabla csv; y en el C, en una tabla dentro de un archivo html.

Se muestra un menú en el que el usuario elige el reporte deseado y se ejecuta un procedimiento correspondiente a ese reporte. Estos procedimientos comienzan con la función init () que hace todo lo necesario desde descargar la página hasta devolver la lista, y finalizan liberando la memoria destinada a la lista. Entre estas dos, se realiza lo correspondiente y particular de cada reporte.

La estructura de datos, funciones y procedimientos necesarios para todos los reportes se encuentran en archivos de acuerdo a su funcionalidad; y cada reporte tiene su archivo correspondiente en el cual están las funciones y procedimientos necesarios para las cuestiones específicas del mismo.

Se utilizan las funciones auxiliares como stringAFloat, stringAInt y floatAString para hacer coincidir los formatos de los números, poder interpretarlos y evaluarlos correctamente.

Archivos:

funcionesArchivos.h - descargar página y, encontrar encabezados y el fin de la tabla.

estructuraDeDatos.h - struct del nodo accion, mostrar nodo de accion por pantalla, mostrar lista de acciones y liberar memoria de una lista.

stringAFloat.h - a partir de un array de caracteres que contiene un numero racional devuelve un float. “-1,02” -> -1.0200000 (para poder cargarlo en el nodo como float y evaluarlo aritméticamente). En algunos casos hubo un error mínimo que no supe solucionar algorítmicamente.

stringAInt.h - a partir de un array de caracteres que contiene un numero entero devuelve un int. “300.123.123” -> 300123123 (para poder cargarlo en el nodo como int y evaluarlo aritméticamente)

floatAString.h - a partir de un float, lo copia en un *char en un formato que Microsoft Excel interpreta como número racional en el .csv. -1.02 -> “-1,02”

htmlALista.h - a partir de un archivo html, devuelve la lista de acciones correspondiente.

main.c - #include del resto de los archivos .h y [stdio.h, stdlib.h, string.h, stdbool.h]; ejecuta menu ( ) en loop. No tiene corte ya que hay un exit en una de las opciones del menú.

menu.h - Interfaz de usuario menu ( ): muestra el menú con las diferentes opciones de reportes, toma por stdin la opción elegida por el usuario y ejecuta el procedimiento del reporte correspondiente. Esto se repite hasta que el usuario ingrese ‘5’ -> exit(0);

mostrarTodasLasAcciones.h - muestra todas las acciones por pantalla

reporteA.h - Lista en pantalla las especies cuyo % de variación supera el 0.5%

reporteB.h - Crea un archivo reporteB.csv con las acciones en el formato pedido

reporteC.h - Crea un archivo reporteC.html con una tabla que contiene el listado del reporte A indicando en color rojo las filas las especies cuyo precio de compra y precio de venta es menor al precio de apertura.

Estructura de datos (estructurasDeDatos.h):

Todos los datos de la tabla se cargan en una lista para acceder más eficientemente a los datos necesarios para cada reporte y poder evaluar aritméticamente los números cuando sea necesario.

```
typedef struct accion accion;

struct accion {
    char    especie[6];
    char    vto[6];
    int     compra_cantidadNominal;
    float   compra_precio;
    float   venta_precio;
    int     venta_cantidadNominal;
    float   ultimo;
    float   variacion;
    float   apertura;
    float   maximo;
    float   minimo;
    float   cierreAnterior;
    int     volumenNominal;
    int     montoOperado;
    int     cantidadOperaciones;
    char    horaCotizacion[9];
    accion  *sgte;
};
```

Formato de los reportes

Los procedimientos de los reportes respetan el siguiente formato:

```
void reporteX () {

    accion *listaDeAcciones = init(); // htmlALista.h

    printf("\n\n\n\t\t\t\tReporte X:\n\n\n");

    // [ Funciones y procedimientos específicos de cada reporte ]

    liberarMemoriaDeLista(listaDeAcciones); //
estructurasDeDatos.h

}
```

Definición de init( )

```
accion * init (){

    descargarPagina(url);
//      Descarga el archivo de texto html dada una url utilizando
el comando wget y popen.

    accion *listaDeAcciones = tabla_HTML_A_Lista();
    /*
     -  Abre el archivo html descargado en modo “r”
     -  Avanza hasta encontrar los encabezados ( irALaTabla en
funcionesArchivos.h )
     -  Hasta encontrar el fin de tabla, va cargando cada linea
html que contiene información en un nodo acción *
( linea_HTML_A_Nodo en htmlALista.h ) utilizando strtok y “</ td>”
como delimitador para serparar los datos.
     -  Cierra el archivo html
     -  Devuelve la lista de acciones.
     */

    int del = remove(archivoHTML);
//      Elimina el archivo html una vez que ya se cargaron los
datos en la lista

    return listaDeAcciones;
//      Devuelve la lista de acciones
}
```

Reporte A

```
void reporteA () {

    accion *listaDeAcciones = init();

    accion *aux = listaDeAcciones; // Puntero auxiliar para ir
recorriendo la lista

    printf("\n\n\n\t\t\t\tReporte A:\n\n\n");

    while (aux) {
        if (condicionReporteA (aux))
            mostrarEspecieYVariacion (aux);
        aux = aux->sgte;
    }
//      Recorre la lista mostrando las acciones que cumplan con la
condición.

    liberarMemoriaDeLista(listaDeAcciones);

}
```

Reporte B

```
void reporteB () {

    accion *listaDeAcciones = init();

    printf("\n\n\n\t\t\t\tReporte B:\n\n\n");

    FILE *csv = crearCSV();
     /*
      -   crea el archivo reporteB.csv
      -   devuelve el FILE * al archivo abierto en modo “w”
      */

    escribirAccionesEnCSV(csv, listaDeAcciones);
    /*
     -   escribe los encabezados separados por ";"
     -   recorre la lista escribiendo cada accion en una línea,
     con los datos separados por ";"
     */

    fclose(csv);

    printf("\n\nEl reporte se encuentra dentro del directorio de
su usuario en el archivo \"reporteB.csv\"\n\n");

    liberarMemoriaDeLista(listaDeAcciones);

}
```

Reporte C

```
void reporteC() {

    accion *listaDeAcciones = init();

    printf("\n\n\n\t\t\t\tReporte C:\n\n\n");

    FILE *html = crearHTML();
    /*
     -  crea el archivo reporteC.html
     -  devuelve el FILE * al archivo abierto en modo “w”
     */

    escribirAccionesEnHTML (html, listaDeAcciones);
    /*
     -  escribe el titulo de la pagina
     -  escribe el estilo de la tabla
     -  escribe los encabezados
     -  recorre la lista escribiendo cada acción en una fila de la
tabla.
     Si la acción cumple con la condición del reporte A, se
utiliza el estilo de color rojo.
     Para esto último se debió utilizar CSS porque HTML5
     ya no soporta el tag <font> que permitía cambiar el color
utilizando html.
     -  Todo lo anterior se realizó utilizando los tags
correspondientes:
     html, body, table, th, tr, td.
     */

    fclose (html);

    printf("\n\nEl reporte se encuentra dentro del directorio de
su usuario en el archivo \"reporteC.html\"\n\n");

    liberarMemoriaDeLista (listaDeAcciones);

}
```