**Métodos REEMPLAZADOS**

public bool hayAgrupacionesNoTerminadas() {

if (listadoNoCerrados.darTamanio()==0) {

return true;

}

return false;

}

public void determinarSiDebeAnalizarseEntrante(char entrante) {

if (listadoNoCerrados.estaVacia() || listadoNoCerrados.darContenidoUltimoNodo().Equals('(')) {//puesto que en la lista solo se encontrarán los signos de apertura por el hecho de que al hallar su cierre serán eliminados de la lista de espera para pasar a concretarse, entonces no habrá problema con esta comparación...

}//fin del if que se encarga de agregar el caracter a la lista de espera de su cierre

}

**MÉTODO PARA OBTNER LA FILA Y COLUMNA [pero solo funciona cuando se escribe, no cuando se posiciona en otro punto al cursor ni con mouse ni con flechitas…]**

int indice = areaDesarrollo.SelectionStart;//obtiene el caracter en donde se encuentra la barra de intercalado [cursor] si está seleccionado entonces devuelve el punto hasta donde llegó dicha selección...

int linea = areaDesarrollo.GetLineFromCharIndex(indice)+1;//recuerda el indice no inicia desde 0 en cada fila sino que sigue su conteo hasta llegar al fin de todos los caracteres que pueda almacenar...

int primerCaracter = areaDesarrollo.GetFirstCharIndexFromLine(linea-1);

int columna = indice - primerCaracter;//donde índice es el número de col dónde el

cursor de intercalado está ubicado...

txtBx\_Informativo.Text ="Linea: "+Convert.ToString(linea)+ " Columna: "+Convert.ToString(columna);

CASO DEL PARÉNTESIS DE CIERRE

[no es un método pero sí es parte de él y si fue reemplazado]

else if(listadoNoCerrados.darUltimoNodo().contenido.StartsWith("(") && lineaDesglosada[caracterActual].Equals(')'))

{

resultadoParentesisODivsion = new Resultado(")", "parentesis\_Cierre", numeroLinea, caracterActual, caracterActual);

ultimoCaracterAnalizado = caracterActual;

listadoNoCerrados.eliminarUltimoNodo();

return resultadoParentesisODivsion;

}

**MÉTODO DE ANÁLISIS DE DECIMAL**

[PUESTO QUE NO CONSIDERABA EL HECHO DE MARCAR ERROR CUANDO EL PUNTO DESUBICADO ESTABA EN LA POSICIÓN 2 A PARTIR DEL PUNTO DEFINITORIO… Y QUE NO CONSIDERABA EL TENER QUE CONCATENAR CUANDO VINIERAN MÁS PUNTOS LUEGO DE ESA POSICIÓN...]

private String[] analizarDecimal(int numeroLinea, int posicionInicialAnalisis, String concatenadoHastaElMomento, char[] lineaAEstudiar)

{

int posicionAnalisis = posicionInicialAnalisis+2;//le sumo 2 por el hecho de que la posición dada corresponde al punto y la siguiente al número por el cual se permitió entrar a este método, entonces es nece analizar el sigueinte a dicho punto que se obtiene al suarle 2

String[] detallesDecimal = { "", "primitiva\_decimal", concatenadoHastaElMomento+Convert.ToString(lineaAEstudiar[posicionInicialAnalisis])+ Convert.ToString(lineaAEstudiar[posicionInicialAnalisis+1]) };//default erronea porque si no viene un número no entraría a un bloque específico y no tendría donde nombrarle de esta manera, por ello mejor cuando se encuentra que todo esta bien, catalogo respecto a ese bienestar...

bool parar = false;

if (posicionAnalisis< lineaAEstudiar.Length && herramienta.determinarTipoCaracter(lineaAEstudiar[posicionAnalisis]) == 'd')//no habrá problema con revisar así de una vez, porque media vez le dieron la posicón, es porque si existe como ubicación dentro del arreglo xD

{

while ((posicionAnalisis < lineaAEstudiar.Length) && (int)lineaAEstudiar[posicionAnalisis] != 32 && (parar == false))

{//Es decir mientras no halle un espacio en blanco... para cuando ya se agregue el identificador esta condi de !palabra deberá cambiar puesto que sin importar que sea palabra o identificador, deberá parar... ahi si deberá usarse un bool, que cb de val para que ya no siga, media vez entre al if para usar el método de Excepción...

if (herramienta.determinarTipoCaracter(lineaAEstudiar[posicionAnalisis]) != 'd')

{

parar = true;

detallesDecimal[1] = excepcionLexico.excepcionDecimal(numeroLinea, posicionAnalisis, lineaAEstudiar);

}

if (parar == false)

{

detallesDecimal[2] += lineaAEstudiar[posicionAnalisis];

posicionAnalisis++;//debe estar aquí porque sino estaría entregando al for 2 passo adelante hasta donde tiene concatenados los caracteres como el tipo que le corresponde al método, en este caso decimal...

}//esto desaperecerá al tener al identificador, porque independientemente de lo que se tenga se debe parar

}

}

if (detallesDecimal[1].Equals("primitiva\_decimal")) {

tipoAgrupacion = "decimal";//aquí ya no hay pierde, puesto que o es decimal porque no había nada más correspodiente o es erronea porque cuando se halló min otro núm había algo que si no debe estar

}

detallesDecimal[0] = Convert.ToString(posicionAnalisis);//esta es la pos que le corresponde al for dar, es decir 1 por más alla de donde se tiene clasificado como decimal en este caso...

return detallesDecimal;

}//listo

**MÉTODO PARA GUARDAR**

[puesto que aquí estabas empleando la ayuda del rich, pero esto provocaría que tuvieras que guardar con extensión rtf o xml… pero debes crearla…]

private void guardarToolStripMenuItem\_Click(object sender, EventArgs e)

{

SaveFileDialog dialogoGuardado = new SaveFileDialog();

dialogoGuardado.DefaultExt = "\*.rtf";

dialogoGuardado.Filter = "archivos .gt|\*.gt";

dialogoGuardado.Title = "Guardar trabajo";

if (dialogoGuardado.ShowDialog()== System.Windows.Forms.DialogResult.OK && dialogoGuardado.FileName.Length>0) {

areaDesarrollo.SaveFile(dialogoGuardado.FileName, RichTextBoxStreamType.RichNoOleObjs);//puede que tengamos problemas con esto, debido a la extensión... si es así

//es un hecho que tendrás que usar el botón de compilado... y por ello leer sin ayuda del rich su debido contenido

}

}

OTRO MÉTODO PARA GUARDAR... SOLO QUE EL TEXTO PLANO…

try {

using (StreamWriter escritor = File.AppendText(path))

{

var decision = MessageBox.Show("El arhivo ya existe, desea reemplazarlo?", "", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if (decision == DialogResult.Yes)

{

File.Delete(path);

escritor.Write(texto);

escritor.Flush();

}

}//se supone que esta parte solo se mostrará cuando se abra un archivo o se "guarde como"

}

catch (DirectoryNotFoundException dirEx) {

MessageBox.Show("La dirección establecida no es correcta");

MessageBox.Show(dirEx.Message);

}

catch (Exception e) {

MessageBox.Show("No se pudo guardar el archivo\nintente nuevamente");

MessageBox.Show(e.Message);

}