Proyecto 1: Analizador Léxico

Viernes 20 de Noviembre

I. Descripción

En este proyecto, Ud deberá desarrollar un *scanner* para el Lenguaje C. Toda la programación debe realizarse en C sobre Linux, usando la herramienta `flex`. La salida es una presentación `Beamer` que será desplegada automáticamente desde su programa. No se pueden cambiar las especificaciones de este documento.

II. Ejecución

El programa se invocará desde la línea de comando de una consola. Recibirá como argumento el nombre del archivo fuente a ser procesado, junto con cualquier opción al estilo tradicional de `UNIX/Linux`.

III. Preproceso

Antes de empezar el análisis léxico, su programa debe manejar una forma básica de preproceso que genera un nuevo archivo temporal con el fuente después del preproceso. Este archivo es la verdadera entrada al *scanner*.

Se siguen las mismas reglas sintácticas del preprocesador de C (el cual es muy estricto y limitado - afortunadamente). Nos limitaremos a las siguientes directivas:

- **`#include`:** permite incluir un archivo de texto como parte del fuente. Dicho archivo siempre reside en el directorio actual. Nótese que este archivo puede contener de nuevo directivas de preproceso que deben ser tomadas en cuenta.
- **`#define`:** asocia un nombre (con las mismas reglas de los nombres de variables) con una hilera de texto formada por el resto de los carácteres de la línea del archivo. Cualquier aparición del mismo nombre en el resto del fuente debe ser reemplazada por la hilera asociada. Esta hilera podría contener otros símbolos definidos con **`#define`** que deben ser expandidos. Para este proyecto no se manejarán macros con parámetros.

Trabajo extra opcional 1: Manejar macros con parámetros

IV. Scanner

Usando la herramienta `flex`, es relativamente fácil generar la mayor parte (sino todo) del código de un analizador léxico o *scanner*[1]. En este caso, se pide que Ud. escriba un *scanner* para el lenguaje C completo, tal y como está descrito en la documentación oficial del lenguaje. Entre otras cosas, se deberá obtener una función semejante al `Get_Token()` estudiado en clases, la que, al ser invocada, regresará el siguiente *token* del fuente procesado, en alguna estructura de datos apropiada que tenga cosas como: código de *token*, puntero al lexema, valor numérico del lexema, etc.

Se recomienda que estructure su programa de tal manera que `Get_Token()` sea independiente del resto del código para posibles usos en proyectos futuros.

La entrada a esta parte de su programa es el archivo de texto temporal generado por el preprocesador, que presumiblemente es un programa escrito en C, pero en realidad podría ser cualquier archivo, ya sea de texto o binario.

V. Salida

Su programa debe producir una presentación `Beamer` con los resultados descritos más abajo. Esto implica, en primer lugar, generar un programa fuente de LaTeX. Internamente, se ejecutará el comando `pdflatex` para que procese dicho archivo y que genere un PDF, el cual será desplegado de inmediato (posiblemente usando el comando `evince`) en modo presentación (*full screen*).

Se espera una presentación `Beamer` de gran calidad (con tablas, colores, dibujos, imágenes, gráficos, etc.) que incluirá como mínimo lo siguiente:

- Portada identificando al grupo de trabajo, al semestre del curso, y al proyecto.
- Una explicación general del proceso de *scanning* y de la herramienta `flex`.
- Múltiples *slides* con el programa fuente que le entró a la fase de *scanning* (*i.e.*, después del preproceso), pero con una clara distinción de cada lexema. Utilice distintos tipos de letra, pesos, inclinaciones y combinaciones de colores (juegos de *foreground* y *background*) para cada categoría léxica. Los errores léxicos deben ser reportados de la misma manera. Ponga una cantidad razonable de líneas del fuente original en cada *slide*.
- Histograma de las cantidades de cada tipo de *token* encontrados en el fuente. Para esto, podrían preparar internamente un archivo de datos e invocar al comando `gnuplot` desde su programa o usar el package `pgfplots` de LaTeX.

Trabajo extra opcional 2: generar un *slide* con un gráfico de pastel de las categorías léxicas encontradas (misma información que el histograma). Se debe usar indispensablemente el package `pgfplots` de LaTeX.

[1] Considere también el uso de `flex` para su preprocesador.

VI. Requisitos Indispensables

La ausencia de uno solo de los siguientes requisitos vuelve al proyecto "no revisable" y recibe un 0 de calificación inmediata:

- La colaboración entre grupos se considera fraude académico.
- Todo el código debe estar escrito en C (no C++).
- El proyecto debe compilar y ejecutar en Linux. Todo debe estar **integrado**, explicaciones del tipo "*todo está bien pero no pudimos pegarlo*"[2] provocan la cancelación automática de la revisión.
- La presentación debe ser de mucha calidad.
- No debe dar "Segmentation Fault" bajo ninguna circunstancia.
- Hacer la demostración en una máquina que levante Linux de manera real (puede ser dual), es decir no usar máquinas virtuales.

VII. Fecha de Entrega

Demostraciones el **Viernes 20 de Noviembre**. Mande además un `.tgz` con todo lo necesario (fuentes, makefile, readme, etc.) a `torresrojas.cursos.05@gmail.com`. Ponga como subject: `COMPILADORES - Proyecto 1 - Fulano - Mengano - etc.`, donde Fulano, Mengano, etc. son los miembros del grupo.

Mucha suerte...

[2] esto incluye los supuestos casos cuando alguien del grupo de trabajo no hizo su parte – el profesor no está interesado en sus problemas de organización.