Proyecto 2: Análisis Sintáctico - *Prettyprint*

I. Descripción

En este proyecto, Ud deberá desarrollar un *parser* para el Lenguaje C con el cual se construye un *prettyprint*. Toda la programación debe realizarse en C sobre Linux, usando las herramientas `flex` y `bison`. No se pueden cambiar las especificaciones de este documento.

Su programa recibe como entrada un programa fuente, aparentemente escrito en lenguaje C, y de no tener errorres de sintaxis produce como salida un nuevo archivo fuente con un reacomodo "bonito".

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 necesaria para ejecutar el programa al estilo tradicional de `UNIX/Linux`. Si el usuario no da los argumentos, o si estos contienen errores (en cantidad o forma) se desplegará una ayuda explicando las opciones disponibles.

III. Preproceso

En el proyecto anterior se desarrolló un preprocesador que podía manejar las directivas `#include` y `#define` sin parámetros. Dicha funcionalidad debe estar correcta para este proyecto, sin embargo por defecto estará desactivada (el usuario puede activarla si lo desea). Esto busca que, en principio, el texto original del programa fuente no se altere en contenido, solo en forma.

IV. Errores Sintácticos

Con la ayuda de `bison`, Ud. escribirá un *parser* del lenguaje C **completo**. Si bien el manejo detallado de errores sintácticos quedará para el siguiente proyecto, en caso de encontrar cualquier error sintáctico su programa dará un mensaje genérico y suspenderá el proceso de *prettyprint*.

Trabajo extra opcional 1: Desplegar la línea con error sintáctico y señalar la posición aproximada del error.

Si no se encuentran errores de sintaxis, se ejecutarán las otras acciones solicitadas por el usuario.

V. *prettyprint*

Un *prettyprint* es un programa que toma un fuente escrito en algún lenguaje de programación (Lenguaje C para este proyecto) y, dado que este no tenga errores de sintaxis, lo reacomoda con ciertas normas estilísticas de espaciado y anidamiento que lo hacen mucho más legible. Obviamente, estas acciones no alteran el significado del programa. Para tener esta capacidad, es necesario entender la estructura gramatical del programa fuente original.

La nueva versión podría reemplazar al programa original o ir a un archivo diferente según indique el usuario en los argumentos de línea de comando. También con algún argumento de línea de comando el preproceso podrá estar desactivado (por defecto) o activado.

Se deben ofrecer al menos 3 estilos de acomodo:

- "GNU style".
- "BSD style".
- Un estilo propuesto por cada grupo de proyecto.

VI. Espacios y *tabs*

Muchos editores de texto reemplazan cierta cantidad de espacios en blanco por uno o varios caracteres de tabulación (*tabs*). Es totalmente posible que el programa fuente que le entra a su proyecto los contenga y deben ser manejados apropiadamente. Su programa debe eliminar estos *tabs* y reemplazarlos por los espacios en blanco necesarios para conseguir los efectos de anidamiento e indentación correspondientes al estilo solicitado.

Trabajo extra opcional 2: Ofrecer la opción de que la salida contenga *tabs* pero respetando el estilo solicitado.

VII. Requisitos Indispensables

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

- Todo el código debe estar escrito en C
- El proyecto debe compilar y ejecutar en Linux
- El proyecto se debe presentar en una **máquina física** que levante en Linux (puede ser dual)

VIII. Fecha de Entrega

Demostraciones el **Viernes 5 de Mayo**. Mande además un `.tgz` con todo lo necesario (fuentes, makefile, readme, etc.) a `torresrojas.cursos@gmail.com`. Ponga como subject: `[COMP] Proyecto 2 - Fulano - Mengano - Sotano`, donde Fulano, Mengano y Sotano son los miembros del grupo.

Mucha suerte...