Trabajo Práctico Nº 4 – Parser

Descendente Recursivo

1. **Objetivos**
   a. Armar un parser para lenguaje mini.
   b. Afianzar el conocimiento del parser descendente recursivo.
2. **Temas**
   a. Parser.
   b. Coordinación entre parser y escáner.
3. **Tareas**
   a. Se pide armar un parser descendente recursivo para el lenguaje mini que fue descripto en un TP previo. Como escáner le acoplaremos el realizado en el TP2.
   Seguiremos trabajando contra stdin y stdout, y redireccionando el archivo a probar al flujo stdin. Se proveen 3 archivos de entrada y sus correspondientes salidas.
   Comentarios:
      1. El archivo `entradaok.txt` es uno totalmente correcto, en tanto que `entrada.txt` tiene errores léxicos y `entradaerr.txt` tiene errores léxicos y sintácticos.
      2. La idea es mostrar un mínimo de por donde pasa el parser. Marcaremos el principio, el fin y para cada sentencia de que tipo es. Esto se puede ver claramente en `salidaok.txt`.
      3. El modo de mostrar por donde va es con un simple `printf` en el lugar adecuado. Si se quiere mostrar un lexema, el buffer que lo representa debería tener el valor adecuado luego de hacer un `match()`.
   b. Programar usando los siguientes fuentes:
      1. `main.c` simplemente llama al parser.
      2. `scanner.h` tiene el enumerado de tokens y toda la información que se necesita conocer en otros fuentes.
      3. `scanner.c` tomar el hecho en el TP2 y agregarle las funciones `match` y `prox_token`. Tener en cuenta que en TP2 no teníamos los tokens de las palabras reservadas. No modificaremos eso, pero guardaremos una lista de cuales son y con eso en la función `prox_token` haremos el cambio si es necesario.
      4. `parser.h` tiene toda la información que se necesita conocer en otros fuentes.
      5. `parser.c` tiene la función parser (u objetivo) y el resto de las funciones. Tener en cuenta al implementarlas que la documentación usa BNF y conviene convertir las recursiones en EBNF con el uso de las llaves como clausura de Kleene.
   c. Puede usar un makefile si lo considera útil, pero no es requerido.
   d. Los errores léxicos a reconocer son los mismos de TP2.
   e. Seguimos sin tener en cuenta los comentarios, es decir, nuestros fuentes para este TP no tienen comentarios por lo tanto no hay que considerarlos en el código.

4. Productos

```
`24-002-xx                          //Repositorio del grupo
    |-- readme.md                   // Carátula del grupo, ya hecha en TP1
    `-- TP4                         //Directorio para el TP2
         |-- readme.md              // Carátula del TP
         |-- main.c                 // Inicio del programa
         |-- scanner.h              // Interfaz del escáner
         |-- scanner.c              // escáner + funciones auxiliares
         |-- parser.h               // Interfaz del parser
         |-- parser.c               // Parser descendente recursivo
```

5. Fechas de entrega

a. Última fecha para primera entrega: 29/10/2024

b. Última fecha para segunda entrega: 12/11/2024