

Compiladores 2023-2

Facultad de Ciencias UNAM

Práctica 2: Lexer y Tokens.

Lourdes del Carmen Gonzáles Huesca Juan Alfonso Garduño Solís
Braulio Aaron Santiago Carrillo

15 de febrero
Fecha de Entrega: 3 de marzo

*“El análisis Léxico incluye un escaneo del código fuente para detectar posibles errores de escritura además de identificar las partes sintácticas mediante la clasificación de **tokens** generando una secuencia de símbolos significativos.”*

Preliminares

Para comenzar con esta primer etapa del compilador es necesario que revise el documento de especificaciones de Jelly para conocer el lenguaje con el que vamos a trabajar, este documento ya se encuentra en tu repositorio como JellySpecs.

Para la parte práctica utilizaremos el lenguaje nanopass y el módulo `parser-tools/lex` del paquete `parser-tools-lib`, si aún no tienes instalado puedes instalarlo con `raco` de la misma manera que nanopass. Así que necesariamente tu archivo debe tener el siguiente encabezado:

```
#lang nanopass
(require parser-tools/lex
         (prefix-in : parser-tools/lex-sre))
```

Implementación

El módulo que requerimos en esta práctica nos permite definir un lexer con la siguiente sintaxis:

```
(lexer [trigger action-expr])
```

Donde `trigger` puede ser una expresión regular y `action-expr` es la acción que desencadena el trigger. Como estamos definiendo un lexer la acción que buscamos es la generación de un token y estos debemos definirlos antes de la siguiente manera:

```
(define-tokens group-id (token-id ...))
(define-empty-tokens group-id (token-id ...))
```

En ambos casos `group-id` es un identificador para cada grupo de tokens y se generan constructores `token-token-id`, sin embargo para el grupo que define la primer función los constructores toman un valor cualquiera y lo pegan al `token-id` correspondiente y de la segunda manera los constructores no toman argumento alguno.

Las acciones que generan el token pueden hacer uso de la variable `lexeme` para recuperar el lexema que está disparando la acción e `input-port` para referirse a la entrada que se está procesando.

Ejercicios

- (4 puntos) A partir de la especificación del lenguaje, crea una *gramática libre de contexto* que pueda generar Jelly, coloca esta respuesta en tu readme.
- (4 puntos) Define un lexer que se llame `jelly-lex` que reconozca todos los posibles lexemas de Jelly y genere el token correcto para cada caso.
- (2 puntos) Define la expresión regular y acción adecuada para los comentarios.

Notas

- Asegurate de que cuando preguntes por una duda que requiera revisar código tu repositorio esté actualizado.
- Para dudas rápidas puedes encontrarme en [Telegram](#).
- Esta práctica ya es en equipos de a lo más 4 personas. **No están permitidos los equipos de una sola persona.**
- Documentación del módulo utilizado: [parser-tools/lex](#).