Universidad Nacional Autónoma de México
Facultad de Ciencias
Lenguajes de Programación

Actividad de laboratorio 5

Karla Ramírez Pulido
karla@ciencias.unam.mx

J. Ricardo Rodríguez Abreu
ricardo_rodab@ciencias.unam.mx

Manuel Soto Romero
manu@ciencias.unam.mx

Luis A. Patlani Aguilar
ayudantefc@gmail.com

6 de septiembre de 2017
La actividad se resuelve en clase
Hora límite de entrega: 16:59:59
Semestre 2018-1

Objetivos

Entender algunas de las etapas de generación de código ejecutable mediante la implementación de un analizador sintáctico para un pequeño lenguaje llamado `WAE`.

Descripción general

La actividad consiste en implementar un analizador sintáctico para el lenguaje `WAE` que se muestra a continuación mediante la gramática expresada en notación EBNF:

```
<expr> ::= <id>
         | <num>
         | {<binop> <expr> <expr>}
         | {with {<id> <expr>} <expr>}

<id> := a |...|z|A|...|Z|aa|...|zz|...
       (Cualquier combinación de carácteres)

<num> := ...|-2|-1|0|1|2|...
         (Cualquier número aceptado por Racket)

<binop> := +|-|*|/
```

Archivos requeridos

El material de esta actividad de laboratorio consta de los siguientes archivos[1]:

- `grammars.rkt` archivo con la definición del tipo de dato abstracto que define los árboles de sintaxis abstracta para el lenguaje `WAE`.
- `parser.rkt` archivo dónde se implementará el analizador sintáctico para el lenguaje `WAE`.
- `test-actividad5.rkt` archivo dónde se agregarán las pruebas unitarias de la actividad.

Actividad

Resolver los siguientes ejercicios de forma limpia y ordenada.

Ejercicio 5.1 Traducir las siguientes expresiones en sintaxis concreta a expresiones en sintaxis abstracta (los ASA), a mano, es decir, se deben apuntar los resultados en una hoja.

1. Identificador:

   ```
   foo
   ```

2. Número:

   ```
   1729
   ```

3. Operación binaria:

   ```
   {+ 17 29 {* 35 {/ 10 {- 8 2}}}}
   ```

4. Asignación local:

   ```
   {with {a 2}
      {with {b 3}
         {with {c 4}
            {+ a {+ b c}}}}}
   ```

Ejercicio 5.2 Por cada una de las expresiones definidas en el ejercicio anterior, agregar una prueba unitaria en el archivo `test-actividad5.rkt` que describa el mapeo entre las expresiones en sintaxis concreta y los ASA mediante el análisis léxico y sintáctico. Por ejemplo, para la expresión `{with {a 2} {+ a a}}` se debería agregar la prueba

```
(test (parse '{with {a 2} {+ a a}})
      (with 'a (num 2) (binop + (id 'a) (id 'a))))
```

No olvidar poner el símbolo quote que representa la etapa de análisis léxico.

[1] Los archivos pueden descargarse desde la página del curso `http://lenguajesfc.com`.

Ejercicio 5.3 Completar el cuerpo de la función `parse` que convierte expresiones devueltas por el analizador léxico (*s-expressions*) en árboles de sintaxis abstracta.

```
;; Función que toma una s-expression y regresa el
;; árbol de sintaxis abstracta correspondiente.
;; parse: s-expression -> WAE
(define (parse sexp)
   ...)
```

```
> (parse '{with {a 2} {+ a a}})
(with 'a (num 2) (binop + (id 'a) (id 'a)))
```

Entrega

El programa se debe entregar debidamente comentado, siguiendo las reglas de estilo mencionadas en clase y con pruebas unitarias para cada función (en caso de que se pueda).

- Si el ejercicio se termina en clase, levantar la mano para que alguno de los ayudantes de laboratorio califique presencialmente.
- Si por cuestiones de tiempo el ejercicio no se puede calificar presencialmente, enviar el código en un archivo `actividad5.tar.gz` a los correos `manu+ldp@ciencias.unam.mx` y `ayudantefc@gmail.com` con el asunto `[LDP-Actividad 5]`. Incluir el nombre de los integrantes en el cuerpo del correo.

Esta actividad sólo será tomada en cuenta a estudiantes que aparezcan en la lista de asistencia de la sesión de laboratorio y equivale a medio punto extra sobre la calificación final de la Práctica 3.