Ambientes SML: Aplicación Web

Lenguajes de Programación

Tarea Programada 3

Profesor:

Ing. Andrei Fuentes L.

Alumnos:

Josué Espinoza C.

Mauricio Gamboa C.

Andrés Pacheco Q.

Junio, 2014

El siguiente escrito pretende documentar la tercera tarea programada, realizada en Python.

Contenidos

Resumen Ejecutivo

Nombre de la aplicación: Ambientes de SML.

Creadores de la aplicación: Josué Espinoza Castro, Mauricio Gamboa Cubero, Andrés Pacheco Quesada.

Objetivo principal de la aplicación: Consiste en la implementación de una aplicación web encargada de analizar programas en SML(Standard Meta Language) con el fin de extraer la información del ambiente dinámico y estático. La aplicación creada permite al usuario agregar un archivo de sml mediante un form, posteriormente se realizaran los ambientes dinámicos y estáticos, esta información es devuelta al usuario en forma de tabla. La información almacenada en los dos ambientes es la siguiente:

* Ambiente dinámico: en él se almacenan los diferentes valores de cada una de las expresiones. Se deben de tomar en cuenta si las variables poseen, a su vez, más variables anidadas, por lo que se requiere de un análisis más profundo.

* Ambiente estático: se almacenan los tipos de datos de las diferentes expresiones.

Como parte de los requerimientos se solicita que la aplicación soporte las asociaciones a las variables, variables, expresiones let y las condiciones if y su cuerpo (then, else). Al igual, como se mencionó anteriormente, la información debe ser mostrada en tablas.

Adicionalmente, como otro requerimiento se establece que como mínimo se deben de manejar los integers, booleans,listas y tuplas. Por lo tanto se deben de manejar operaciones sobre variables con valores enteros, el caracter cons (::), manejo de head(hd) y tale (tl), y el acceso a las tuplas mediante (#). Además, se contempla que las operaciones y la sintaxis del programa a recibir no debe poseer errores, ya que las validación es no forman parte de la especificación. Sin embargo, sí se contemplan varios errores básicos. Las operaciones permitidas son las siguientes :

*Suma (+)

*Resta (+~)

*Multiplicacion (*)

*Division (div)

*Modulo (mod)

Razón de ser de la aplicación

Dada la necesidad que existe actualmente alrededor del lenguaje SML, se ha creado este programa. Como es sabido por todos los programadores, no es sencillo conseguir información confiable y concreta de este lenguaje en la red. Las fuentes que se conocen de SML son escasas, por lo que un analizador como el que se realizó en este proyecto resolverá la problemática de aquellos que gustan de SML para implementar el paradigma funcional.

Si bien es cierto SML realiza compilaciones cuando se corre el programa creado, no desarrolla un análisis tan profundo de cada una de las variables que tiene el programa (al menos no a simple vista). Esta aplicación es útil en ese sentido, además para indicar el alcance que esa variable posee, y entender de una mejor manera cuándo se puede utilizar y por qué.

En caso de que el programador esté dando sus primeros pasos en este lenguaje, la aplicación realizada en este proyecto permite a cada uno brindarle un aprendizaje más completo, que le permite, tal vez , lograr dominarlo en un tiempo más corto.

Funcionalidades:

Las funcionalidades de la aplicación se definen en los siguientes puntos:

1. El usuario puede subir el programa SML a la aplicación mediante un form. El programa a analizar debe estar en la computadora del usuario. El usuario debe seleccionar el archivo a agregar.

2. El servidor analiza el archivo recibido con el fin de analizar el ambiente dinámico y estático.

3. Finalmente, se muestra al usuario el análisis realizado en forma de tabla donde se muestra el valor y el tipo de dato de las expresiones analizadas.

Herramientas utilizadas:

- Python: Consiste en el lenguaje multiparadigma que se utilizó para manejar la lectura de código en SML. Desde este lenguaje se define cada función necesaria para la ejecución correcta de la lectura de variables del archivo SML deseado.

- Flask: Consiste en un "framework" escrito en Python con una semejante amigabilidad con el usuario que este lenguaje, cuya función es desarrollar aplicaciones web, razón por la cual fue incluido en este proyecto.
- Web responsive: Consiste en un diseño web que provee una experiencia de visualización óptima, ajustando la aplicación web que se esté realizando a casi cualquier pantalla, logrando que se pueda ver perfectamente ajustado, con una visión de los íconos óptima y proporcional al tamaño de la misma.

Descripción detallada del proyecto

A continuación se presenta una descripción detallada de las decisiones de diseño, los lenguajes utilizados, las tecnologías usadas y las librerías importadas. Además se muestran las diferentes funciones creadas con el fin de mostrar la funcionalidad, y los argumentos que recibe cada una, así como las decisiones tomadas para su implementación de tal forma que nos pudieran beneficiar en la creación de la aplicación.

SML

Consiste en lenguaje de programación funcional que desciende del lenguaje ML.

Python

Es un lenguaje de programación multipropósito, de alto nivel. Posee gran amigabilidad con el usuario, ya que con pocas líneas de código se pueden expresar conceptos que en otros lenguajes resulta más complejo. Es un lenguaje multiparadigma, ya que contiene el orientado a objetos, imperativo y funcional.

Para esta aplicación se debió leer archivos .sml desde Python, y se leía cada variable que contenía para insertarla en la página web.

Import re

Al tener que separarse el programa por medio de muchos caracteres, se investigó la librería "re", que permite separar por varios caracteres simultáneamente.

Ejemplo:

temp=re.split(' |;|\n|(=)*',linea) , donde línea es el string al que se le desea hacer el split, y se separará por espacios(' '), punto y coma (;), saltos de línea (\n) y signos

de igual (=); este último se encuentra entre paréntesis y con un asterisco(*) porque, además de dividirlo tomando el símbolo de igual, lo inserta en lista deseada (en este caso sería temp).

Decisiones de diseño

Se diseñó el programa de tal forma que la matriz que contiene cada línea de código de SML como vector, se traspase de una función a otra. De esta forma, se tratará de que, recursivamente, se pueda recorrer la lista, obteniendo cada dato necesario.

Funciones principales:

Función Leersml:

Esta función lee un archivo de extensión '.sml', y cada una de las líneas es enviada a NuevaLinea donde se procede a almacenarlos (en forma de lista) en una matriz. Posteriormente, desde leersml() se envía la matriz a la función agrupar donde efecctúa el funcionamiento explicado a continuación.

Función agrupar:

Función que recibe a la matriz, un entero identificador, una matriz con el valor que lleva actual, y una matriz agrupada o total.

Si el entero identificador (en el código se llama ide) es igual a 1, significa que el primer valor de la lista es 'val'. Después de esto, agrega cada carácter hasta que llegue a un val, let o in. Cuando lo haga, se llama a la función recursivamente, y dentro de valActual va una 'sublista' con lo que se lleva al momento (esto en caso de que continúe en la siguiente línea de código), si lo que sigue es un if, ide será igual a 3, si es un let será 2, y si es un val será 1, y con cada uno realizará el mismo funcionamiento.

Ejemplo:

Matriz=[['val','x','=','2','let'],['val','y','=','x+4'],['in','x*3','end]]
Y agrupar retornaría lo siguiente:
agruapda2=[['val','x','=','2'],['let',['val','y','=','x+4'],'in','x*3','end]]

Función para agregar datos :

Debido a que se manejó una pila para el almacenamiento de los ambientes, la función de agregar datos inserta al inicio de la pila la información necesaria para almacenar los datos. Los datos son guardados en la lista global establecida.

Operaciones complejas:

Para manejar las operaciones complejas se utilizaron dos pilas para almacenar en una los operadores y en otra los números. De esta forma se analizan las prioridades teniendo acceso a un diccionario para conocer la precedencia de los operadores. Si un valor fue asignado anteriormente a una variable, los métodos para resolver las operaciones complejas pueden encontrar este valor en la lista donde se almacenan los ambientes.

Operaciones sobre listas:

Estas operaciones sobre listas, permiten concatenar listas y sus diferentes partes. Es decir, las cabezas y colas. Cabe resaltar que en SML el manejo de los niveles de anidamiento varía en cada caso, por lo tanto no cualquier operación está definida.

Analizador de tipos de tuplas y listas:

Estos métodos analizan recursivamente los diferentes tipos de los datos almacenados en las tuplas y en las listas. Por lo tanto, se verifican si los datos son enteros, booleans, otras listas o tuplas.

Verificadores de tipos:

Sirven para analizar los tipos de los datos que han sido almacenados en los ambientes, ya que depende del bloque de código en el que una variable sea invocada, se debe utilizar el valor almacenado.

Acceso a las tuplas:

Para tener acceso a los datos de las tuplas, se manejaron contadores y una lista para almacenar los valores por ejemplo (#1(e)) se almaceno a parte el 1. Debido a la complejidad del manejo de múltiples niveles de anidamiento, solo se puede tener acceso a los valores de las tuplas a un nivel.

Problemas encontrados

El problema más importante que se tuvo en la programación, y que es meritorio mencionarlo en el documento presente, consiste en la cantidad de variantes que podía tener el código. Es complicado poder validar cada detalle que el usuario quiere ingresar, esto asumiendo que no posee errores.

Manual de Usuario

Para ejecutar Flask, debe correr las siguientes líneas de código en la terminal:

1. sudo apt-get install python-virtualenv
2. Ubicarse en la terminal en la carpeta del programa (vía comandos cd) por ejemplo:cd TP3_SML y en la siguiente línea : cd VersionFinal
3. Ejecutar la siguiente línea: . venv/bin/activate
4. Ejecutar la siguiente línea: sudo pip install

Para ejecutar el programa, debe hacer lo siguiente:

1. Cambiar la última línea del archivo app.py: app.run(host='IP actual'). Ejemplo: app.run(host='192.168.98.130')
2. Abrir la terminal, ubicarse en la carpeta donde esta el programa: Ej: cd TP3_SML y en la siguiente línea: cd VersionFinal
3. . venv/bin/actívate
4. python app.py