Compiladores U-2017
Informe Proyecto
Traductor
C a Bash

Juan Andrés Vivas
Julián Briceño

12 de diciembre de 2017

1. Descripción del problema

Se plantea hacer una traducción simple del lenguaje C a Bash, cabe destacar que se está pasando de un lenguaje fuertemente tipado a uno débilmente tipado.

El objetivo es traducir expresiones matemáticas sencillas, lectura de variables, impresiones por pantalla y la estructura de decisión if y de repetición while.

1.1. Alcance

Lograr traducir expresiones matemáticas simples, tales como: sumas, restas, multiplicaciones, divisiones y calculo de modulo. Las expresiones se limitaran a tener un solo tipo de operación.

Se tendrán expresiones lógicas las cuales sirven para las estructuras if y while, limitadas a un solo operador lógico.

Se leerá una sola variable por la entrada estándar y se acoto a 3 variables la cantidad que se pueden imprimir.

El tipo de datos en C a usar son int, float y char. A pesar de reconocer los flotantes, al momento de traducir solo se tomara en cuenta su parte entera.

1.2. Especificación del lenguaje fuente

El lenguaje fuente es C reducido, limitado a operaciones matemáticas sencillas y las estructuras if y while. Se reconocerá las bibliotecas que pueda tener y lo que este dentro de la función main sin ningún parámetro.

Tiene los tipos de datos entero, carácter y flotante. Se podrán declarar y asignar variables y valores.

Se leerá una sola variable por la entrada estándar y la salida se acoto hasta 3 variables.

Todo lo demás no sera reconocido por nuestra gramática ni traducido.

1.3. Especificación del lenguaje a traducir

El lenguaje de consola Bash (Bourne again shell) interpreta ordenes en una shell de Unix. Interpretara operaciones matemáticas simples, comparación de variables, lectura y escritura de variables por la entrada y salida estándar

2. Solución al problema planteado

La traducción se realiza a partir de verificar la correctitud del lenguaje fuente, de lo contrario no se realiza.

Se divide en las siguiente secciones.

2.1. Análisis Léxico

Se encarga de detectar todos los lexemas y pasarlos al análisis sintáctico como tokens.
Contiene el siguiente conjunto de expresiones regulares:

LETRA [a–zA–Z_]
DIGITO [0 – 9]
{LETRA}({LETRA}|{DIGITO})*
(–)?{DIGITO}+
(–)?{DIGITO}*"."{DIGITO}+

Las cuales hacen match para detectar identificadores, tipos de datos, palabras reservadas y valores numéricos

También detecta el siguiente conjunto de caracteres:

. ; { } ( ) < > ! = + – * / ^ : # &
<= >= != += –= *= /= %= == || &&
++ — /% , ' "

Cada expresión regular enviá un token al analizador sintáctico al igual que cada carácter antes descrito produce un token.

El analizador léxico también se encarga de eliminar del lenguaje fuente los espacios en blanco, saltos de linea y tabulaciones. Cuenta saltos de linea y rechaza aquellos caracteres que no coincidan con ningún patrón valido definido.

2.2. Análisis Sintáctico

El análisis sintáctico se realiza a través de la siguiente gramática, el cual recibe los tokens y los agrupa de acuerdo a las reglas de producción de la gramática que se presenta a continuación

```
programa:
        codigo;

codigo:
```

```
        cabecera principal | principal;

cabecera:
        cabecera NUMERAL RESERVADA MENOR ID MAYOR
        | NUMERAL RESERVADA MENOR ID MAYOR
        | cabecera NUMERAL RESERVADA COMILLAS TEXTO COMILLAS
        | NUMERAL RESERVADA COMILLAS TEXTO COMILLAS
        | cabecera NUMERAL RESERVADA MENOR ID PUNTO ID MAYOR
        | NUMERAL RESERVADA MENOR ID PUNTO ID MAYOR;

principal:
        TIPO RESERVADA PARENTESISABR PARENTESISCERR LLAVEABR cuerpo LLAVECERR;

cuerpo:
        asignacion cuerpo | asignacion | declaracion cuerpo | declaracion
        | retornar cuerpo | retornar | scan cuerpo | scan
        | print cuerpo | print | estructura cuerpo | estructura
        | estructura LLAVEABR cuerpo LLAVECERR cuerpo
        | estructura LLAVEABR cuerpo LLAVECERR
        | RESERVADA LLAVEABR cuerpo LLAVECERR cuerpo
        | RESERVADA LLAVEABR cuerpo LLAVECERR | RESERVADA cuerpo;

estructura:
        RESERVADA PARENTESISABR condicional PARENTESISCERR;

scan:
        RESERVADA PARENTESISABR COMILLAS PRCVAL COMILLAS COMA
        AMPERSAND ID PARENTESISCERR PTOCOMA;

print:
        RESERVADA PARENTESISABR COMILLAS TEXTO COMILLAS PARENTESISCERR PTOCOMA
        | RESERVADA PARENTESISABR COMILLAS TEXTO PRCVAL TEXTO COMILLAS COMA
        ID PARENTESISCERR PTOCOMA
        | RESERVADA PARENTESISABR COMILLAS PRCVAL TEXTO COMILLAS COMA
        ID PARENTESISCERR PTOCOMA
        | RESERVADA PARENTESISABR COMILLAS TEXTO PRCVAL COMILLAS COMA
        ID PARENTESISCERR PTOCOMA
        | RESERVADA PARENTESISABR COMILLAS PRCVAL COMILLAS COMA ID
        PARENTESISCERR PTOCOMA
        | RESERVADA PARENTESISABR COMILLAS PRCVAL TEXTO PRCVAL COMILLAS
        COMA ID COMA ID PARENTESISCERR PTOCOMA
        | RESERVADA PARENTESISABR COMILLAS PRCVAL TEXTO PRCVAL TEXTO PRCVAL
        COMILLAS COMA ID COMA ID COMA ID PARENTESISCERR PTOCOMA;

condicional:
        ID IGUALD ID | NUM IGUALD ID | ID IGUALD NUM
```

```
        | ID MAYOR ID | ID MAYOR_I ID | ID MENOR ID | ID MENOR_I ID
        | NUM MAYOR ID | NUM MAYOR_I ID | NUM MENOR ID | NUM MENOR_I ID
        | ID MAYOR NUM | ID MAYOR_I NUM | ID MENOR NUM | ID MENOR_I NUM;

retornar:
        RESERVADA NUM PTOCOMA | RESERVADA ID PTOCOMA
        | RESERVADA PARENTESISABR NUM PARENTESISCERR PTOCOMA
        | RESERVADA PARENTESISABR ID PARENTESISCERR PTOCOMA;


declaracion:
        TIPO ID PTOCOMA | TIPO ID IGUAL NUM PTOCOMA
        | TIPO ID IGUAL COMISIMPLE ID COMISIMPLE PTOCOMA
        | TIPO ID IGUAL COMISIMPLE NUM COMISIMPLE PTOCOMA
        | TIPO ID IGUAL ID PTOCOMA;

asignacion:
        ID IGUAL ID PTOCOMA | ID IGUAL NUM PTOCOMA | ID SUM_ASSIGN ID PTOCOMA
        | ID SUM_ASSIGN NUM PTOCOMA | ID SUB_ASSIGN ID PTOCOMA
        | ID SUB_ASSIGN NUM PTOCOMA | ID MUL_ASSIGN ID PTOCOMA
        | ID MUL_ASSIGN NUM PTOCOMA | ID IGUAL suma PTOCOMA
        | ID IGUAL resta PTOCOMA | ID IGUAL multi PTOCOMA
        | ID IGUAL div PTOCOMA | ID IGUAL ID PORCENTAJE ID PTOCOMA
        | ID IGUAL ID PORCENTAJE NUM PTOCOMA
        | ID IGUAL NUM PORCENTAJE ID PTOCOMA
        | ID IGUAL NUM PORCENTAJE NUM PTOCOMA | ID INC PTOCOMA
        | ID DEC PTOCOMA | INC ID PTOCOMA | DEC ID PTOCOMA;

suma:
        suma SUMA ID | suma SUMA NUM | ID SUMA ID | NUM SUMA NUM
        | ID SUMA NUM | NUM SUMA ID;

resta:
        resta MENOS ID | resta MENOS NUM | ID MENOS ID | NUM MENOS NUM
        | ID MENOS NUM | NUM MENOS ID;

multi:
        multi MULTI ID | multi MULTI NUM | ID MULTI ID | NUM MULTI NUM
        | ID MULTI NUM | NUM MULTI ID;

div:
        div DIV ID | div DIV NUM | ID DIV ID | NUM DIV NUM
        | ID DIV NUM | NUM DIV ID;
```

La tabla de símbolos se implemento con un vector dinámico de tuplas, el cual agrupa el tipo de dato con su identificador.

2.3. Análisis Semántico

Se encarga de verificar que las palabras reservadas estén correctamente ubicadas dentro del lenguaje fuente. Las variables deben ser declaradas antes de ser usadas y al momento de hacer una asignación los tipos de datos sean iguales.

2.4. Manejo de errores

Empieza en el análisis léxico, lleva la cuenta de las lineas que se van leyendo para poder decir en que linea se ubica un error en el caso de ser detectado, ademas de emitir un mensaje si no se llega a reconocer alguna palabra.

El análisis semántico falla al momento de que no se pueda construir un árbol de derivación correcto a partir de la gramática definida. Se detiene la traducción y se avisa en que linea esta el error sintáctico

Al momento de revisar la semántica, se valida que todas las variables sean declaradas antes de invocarse, al momento de ser asignadas tengan el mismo tipo de dato y por ultimo que las palabras reservadas coincidan con su ubicación y uso dentro del lenguaje. De lo contrario se detiene la traducción y se emite un mensaje de error con la linea donde se encontró y la variable o palabra reservada involucrada.