**Laboratorio**

**Analizador Semantico con JAVACC**

Vamos a realizar un analizador semántico, para un lenguaje que permite hacer una operación matemática o ingresar un identificador y terminar con el signo ?

La gramática para el lenguaje es:

<calculate> ::= <expr>

<expr>::= <term> "+" <expr> | <term> - <expr> | <term>

<term>::= <factor> "\*" <term> | <factor> "/" <term> | <factor>

<factor> ::= <numero> | "(" < expr> ")"

El programa si le entra una variable, imprimirá “El resultado de la operación es -1”; pero si le ingresa una operación matemática, el imprimirá “El resultado de la operación es XX”, donde XX es el valor que calculo.

Ejemplo

Ingrese el código a analizar:

3+4\*5?

Se ha compilado con éxito

El resultado de la operación es: 23

Ahora como siempre vamos a crear un proyecto con la gramática, para esto:

Usando JAVACC en eclipse.

Usando JAVACC en eclipse.

1. Cree un proyecto en JAVA (Java Project)
2. Una vez que el proyecto esta creado, seleccionar la carpeta src y dar clic derecho sobre ella.
3. Seleccione New->Other.
4. En la ventana que sale escoja JavaCC->JavaCC Template File y de click en Next.
5. Escriba el nombre de la clase por ejemplo Calculadora
6. Por defecto el crea un código fuente de prueba. Borrenlo todo y escriban en siguiente código

**options{**

STATIC = **false**;

**}**

**PARSER\_BEGIN(Calculadora)**

**public** **class** Calculadora{

**public** **static** **void** main(String args []) **throws** ParseException

{

**try**

{

System.out.println("Ingrese el código a analizar:");

Calculadora calc = **new** Calculadora(System.in);

**int** resultado;

resultado = calc.calculate();

System.out.println("Se ha compilado con éxito");

System.out.println("El resultado de la operación es: "+ resultado);

}

**catch**(ParseException e)

{

System.err.println("Se han encontrado errores");

System.err.println(e.getMessage());

}

}

}

**PARSER\_END(Calculadora)**

/\*Sección de Tokens\*/

**SKIP**:**{**

" "

**|** "\t"

**|** "\n"

**|** "\r"

**}**

**TOKEN**:**{**

**<**ADD:**(**"+"**)>**

**|** **<**SUB:**(**"-"**)>**

**|** **<**MUL:**(**"\*"**)>**

**|** **<**DIV:**(**"/"**)>**

**|** **<**LBR:**(**"("**)>**

**|** **<**RBR:**(**")"**)>**

| *<INT*:(["0"-"9"])+*>*

| *<IDENTIFICADOR*:(["*a*"-"*z*"])+*>*

| *<FIN* : "?"*>*

**}**

/\*Sección Gramatica y Semantica

<calculate> ::= <expr>

<expr>::= <term> "+" <expr> | <term> - <expr> | <term>

<term>::= <factor> "\*" <term> | <factor> "/" <term> | <factor>

<factor> ::= <numero> | "(" < expr> ")"

\*/

**int** calculate():**{**

**int** result;

**}**

**{**

result = expr() <FIN>**{**

**return** result;

}

| *<IDENTIFICADOR>* *<FIN>*

{

**return** -1;

**}**

**}**

**int** expr():**{**

**int** a;

**int** b;

**int** result = 0;

**}**

**{**

**(**

/\*

a = term(){ result+=a; } <ADD> b = expr(){ result+=b; }

| a = term(){ result+=a; } <SUB> b = expr(){ result-=b; }

| a = term(){ result+=a; }\*/

a = term()**{** result+=a; **}**

(*<ADD>* *b* = *term*()**{** *result*+=*b*; **}** | *<SUB>* *b* = *term*()**{** *result*-=*b*; **}** )\*

)

{ **return** result; }

}

**int** term():**{**

**int** a;

**int** b;

**int** result = 0;

**}**

**{**

a = factor()**{** result = a; **}**

(*<MUL>b* = *factor*()**{** *result*\*=*b*; **}** | *<DIV>b* = *factor*()**{** *result*/=*b*; **}**)\*

{

**return** result;

**}**

}

**int** factor():**{**

Token t;

**int** result;

**}**

**{**

t = <INT>**{** **return** Integer.parseInt(t.image);

**}**

| *<LBR>result* = *expr*()*<RBR>***{** *return* *result*; **}**

**}**

1. Ahora revise el código fuente que acaba de copiar y pegar y escriba que ve diferente al código que hasta el momento hemos trabajado.
2. Luego analice el código fuente que acaba de copiar y pegar y describa como funciona básicamente.
3. Ahora modifique el código para que permita realizar varias operaciones matemáticas, separadas por ; imprima el resultado para cada una. Ejemplo

3+4\*2; (5/2)+3; 1\*2;?

Deberá mostrar

Resultado: 11

Resultado: 5

Resultado: 2

Envíe un documento con lo realizado en el laboratorio y envíe el archivo .jj

**Los siguientes pasos son opcionales para el que quiera tener 1 unidad adicional en el parcial final.**

1. Adicione la declaración de variables de tipo entero y de tipo string, que solo permita de una en una. Ejemplo

int a;

int b;

string c;

1. Adicione la asignación de expresiones Ejemplo

a=3;

b= 2\*4;

c= “Hola”;

1. Adicione la verificación de tipos, es decir que si para los ejemplos dados aparece algo como:

4+c\*3;

Muestre que no es posible realizar la operación y retorne un -1

Pero si aparece algo como

2\*b;

Debe mostrar El resultado es: 12