****

Universidad Tecnológica Centroamericana

Compiladores 2

Manual de Usuario de

Ing. Carlos Vallejo

LEONARDO MONTOYA 11741303

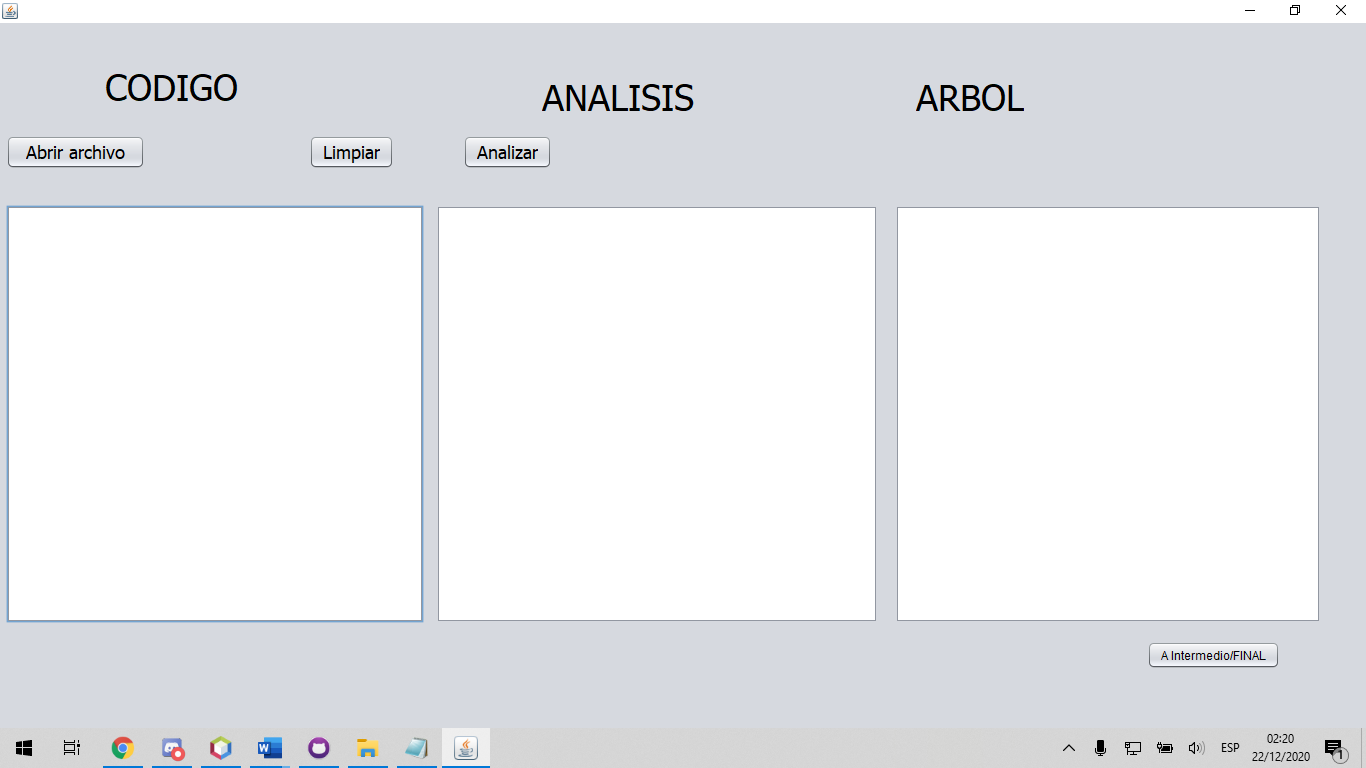
KENNETH NUÑEZ 11741149

ERICK CARRASCO 11641166

22 de Diciembre de 2020

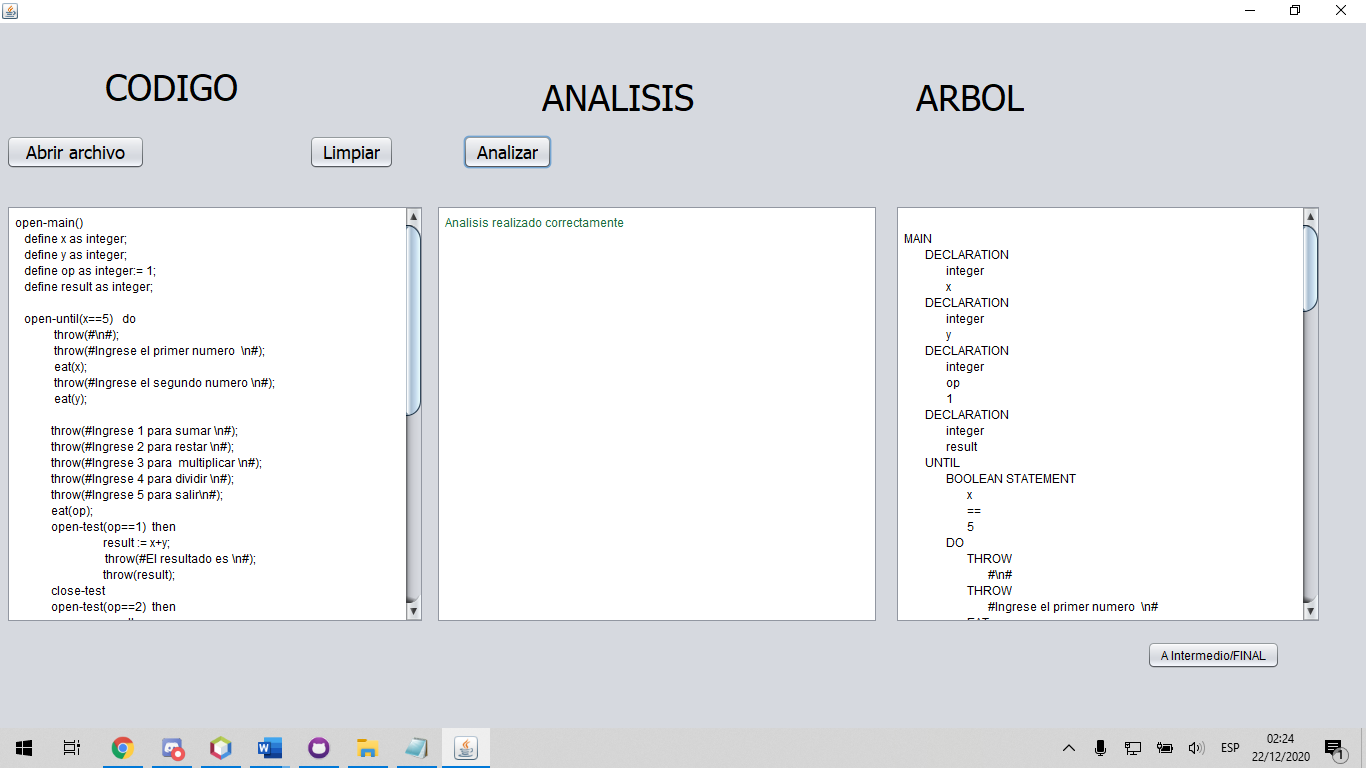
**Bienvenido a nuestro IDE**

**Ejecuta el programa y disfrútalo.**

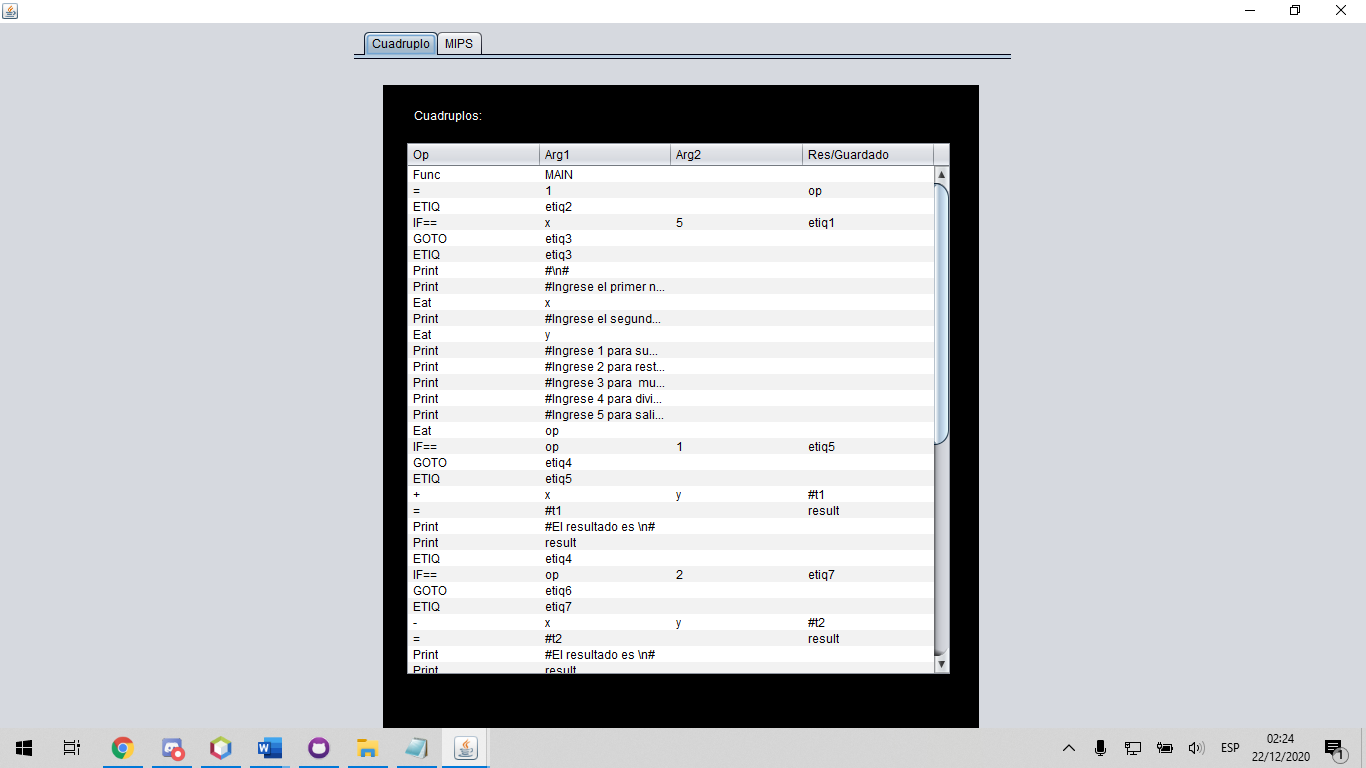


La interfaz cuenta con un text área que puede ser usado para editar en tiempo real el código.

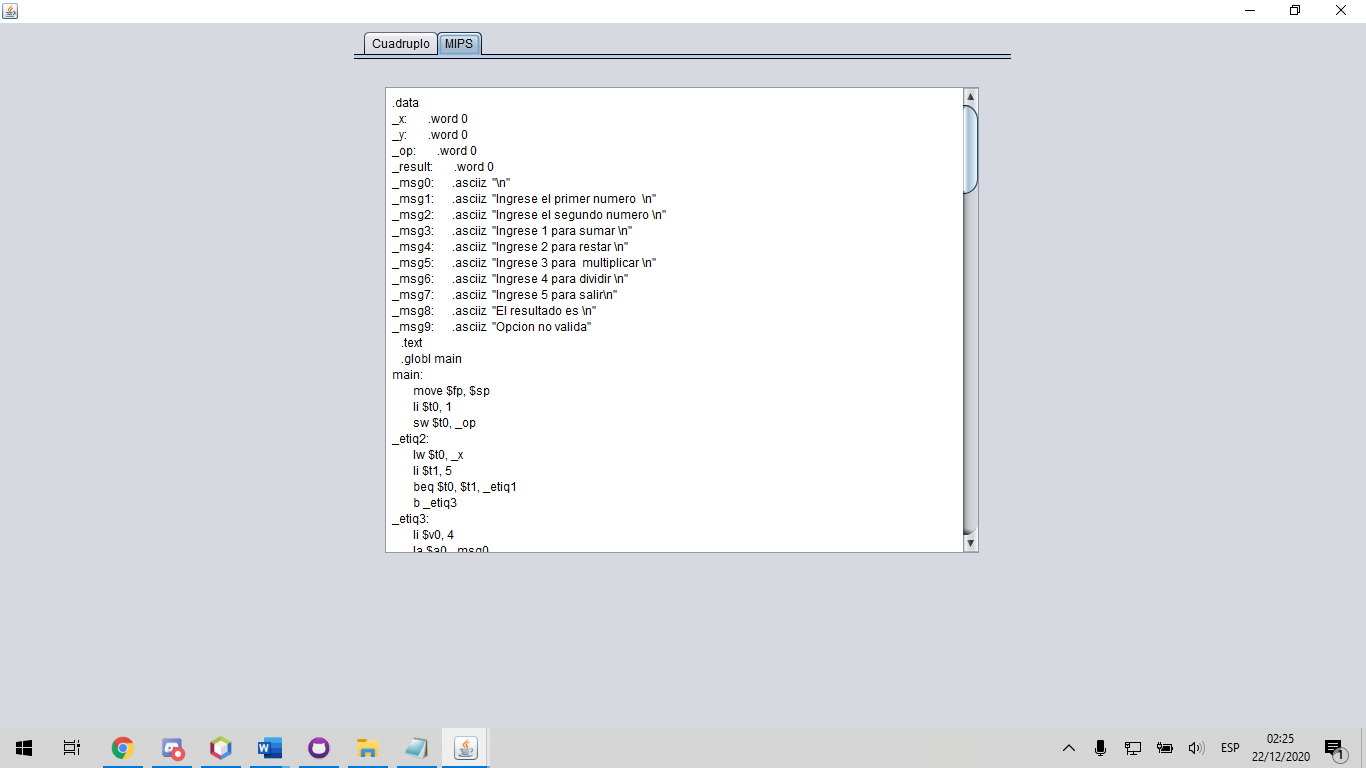
¡Tan fácil como ingresar el código y correrlo!



Nuestro proyecto cuenta con una ventana para la demostracion de los cuadruplos generados.



Adicionalmente cuenta con una ventana para la demostración del código MIPS.



**Manual Tecnico del Compilador.**

* Nuestro compilador utiliza variables de tipo entero, booleanas, caracteres y arreglos de una dimensión.

Dirigirse a la sección de declaraciones para más información.

* Permite la lectura y escritura de todos los tipos básicos mencionados anteriormente.

Dirigirse a la sección de lectura y escritura para más información.

* Utiliza bloques de decisión (if) e iteración.

Dirigirse a la sección de IF, IF ELSE, FOR, UNTIL para más información.

* Utiliza un bloque de decisión similar al switch de Java para valores de tipo entero y caracter. o Permitir la recursividad.

Dirigirse a la sección de IF, IF ELSE, FOR, UNTIL para más información.

* Nuestro compilador no es igual a ningún lenguaje ya existente, y utiliza simbología intuitiva y con sentido.
* El ámbito de las variables es ámbito léxico con bloques; es decir, se permite la declaración de variables dentro de if, for, etc. para variables que vivirán solamente dentro de este ámbito. El compilador no posee la capacidad de ocultación de variables.
* Todo programa correcto deberá tener siempre un cuerpo principal (main).
* El analizador sintáctico crea un AST (Árbol de Análisis Sintactico) que es utilizado en las fases del proceso de compilación. Este AST puede ser visualizado de manera clara para su revisión.
* Se maneja la precedencia común de operadores, es decir + y – tienen precedencia más baja que \* y /. De igual forma, el operador = tiene precedencia más baja que el resto de los operadores relacionales.
* Permite la inclusión de comentarios en el código fuente.

Dirigirse a la sección de comentarios para más información.

* Es capaz de reconocer y recuperarse de errores léxicos y errores sintácticos.
* Las funciones y procedimientos podrán tener cualquier cantidad de parámetros de tipos integer, character y boolean , así como el valor de retorno.
* El paso de parámetros es por valor. De igual forma el sistema de reconocimientos de tipos es por valor. Los primeros 4 parámetros serán enviados en registros del procesador y el resto en el stack. El valor de retorno de las funciones también deberá ser enviado en un registro del procesador, de acuerdo al estándar del MIPS.
* Ningún identificador puede ser declarado dos veces en el mismo ámbito.
* Chequeo del ámbito de los identificadores. No se permite utilizar una variable en el ámbito incorrecto.
* En la asignación estática de tipos, ningún identificador puede ser usado sin ser declarado previamente. Cualquier que sea el tipo de asignación a utilizar, ésta deberá especificarse claramente en el documento.
* Chequeo de tipos. No se permiten asignaciones ni comparaciones entre variables o constantes de diferentes tipos. También se chequean que los parámetros enviados a las funciones sean el número y tipo correcto
* Retorno de valores solamente en funciones
* Generación de código intermedio. El programa es capaz de mostrar el código intermedio generado.
* El análisis semántico, así como la generación de código intermedio es realizado recorriendo el AST.
* Generacion de Código MIPS.
* El programa debe es gráfico, mostrar todos los errores en este ambiente y permite ahí mismo la actualización del archivo que se esté compilando.

**Descripción del lenguaje:**

**Tipos**

1. Entero: integer
2. Boolean: boolean
3. Char: character
4. Arreglo: tipo[]

**Declaración**

1. define id as tipo;
2. define h1 as integer;
3. define classic as character[9];

**Asignación**

1. h1 := 2;
2. define h2 as character := 'x';

**OPERADORES RELACIONALES**

**& AND**

**| OR**

**! NOT**

**<**

**>**

**<=**

**=>**

**==**

**!=**

**Comentarios:**

* Una línea: //
* Multilínea:/\*

\*/

**Lectura y escritura:**

* eat (id); //LECTURA
* throw(id); //ESCRITURA
* throw(#CADENA#)//Impresión de Strings
* throwln(id);ESCRITURA

**IF:**

open-test (cond) then

b1;

close-test

**IF ELSE:**

open-test (cond) then

b1;

or

b2;

close-test

**FOR:**

open-for (/\*integer i:= num; condición;incremento\*/) do

//Sentencias

close-for

**until:**- cuando la condición se cumple para

open-until (/\*condición\*/) do

//Sentencias

close-until

**SWITCH:**

open-match(x) do

when A := statement;

stop;

close-match

**Funciones:**

open-method tipo f()

close-method

**main:**

open-main()

close-main

**Ejemplo:**

open-main()

define x as integer;

define y as integer;

define op as integer:= 1;

define result as integer;

open-until(x==5) do

throw(#\n#);

throw(#Ingrese el primer numero \n#);

eat(x);

throw(#Ingrese el segundo numero \n#);

eat(y);

throw(#Ingrese 1 para sumar \n#);

throw(#Ingrese 2 para restar \n#);

throw(#Ingrese 3 para multiplicar \n#);

throw(#Ingrese 4 para dividir \n#);

throw(#Ingrese 5 para salir\n#);

eat(op);

open-test(op==1) then

result := x+y;

throw(#El resultado es \n#);

throw(result);

close-test

open-test(op==2) then

result := x-y;

throw(#El resultado es \n#);

throw(result);

close-test

open-test(op==3) then

result := x\*y;

throw(#El resultado es \n#);

throw(result);

close-test

open-test(op==4) then

result := x/y;

throw(#El resultado es \n#);

throw(result);

close-test

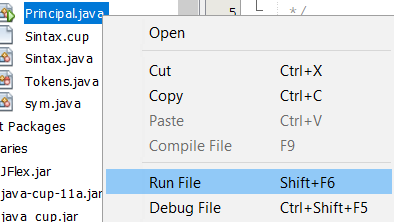
open-test(op>5) then

throw(#Opcion no valida#);

close-test

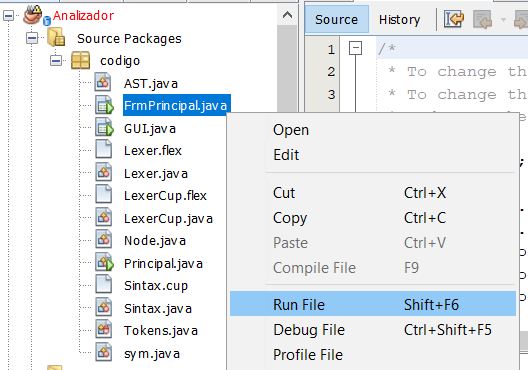
close-until

close-main

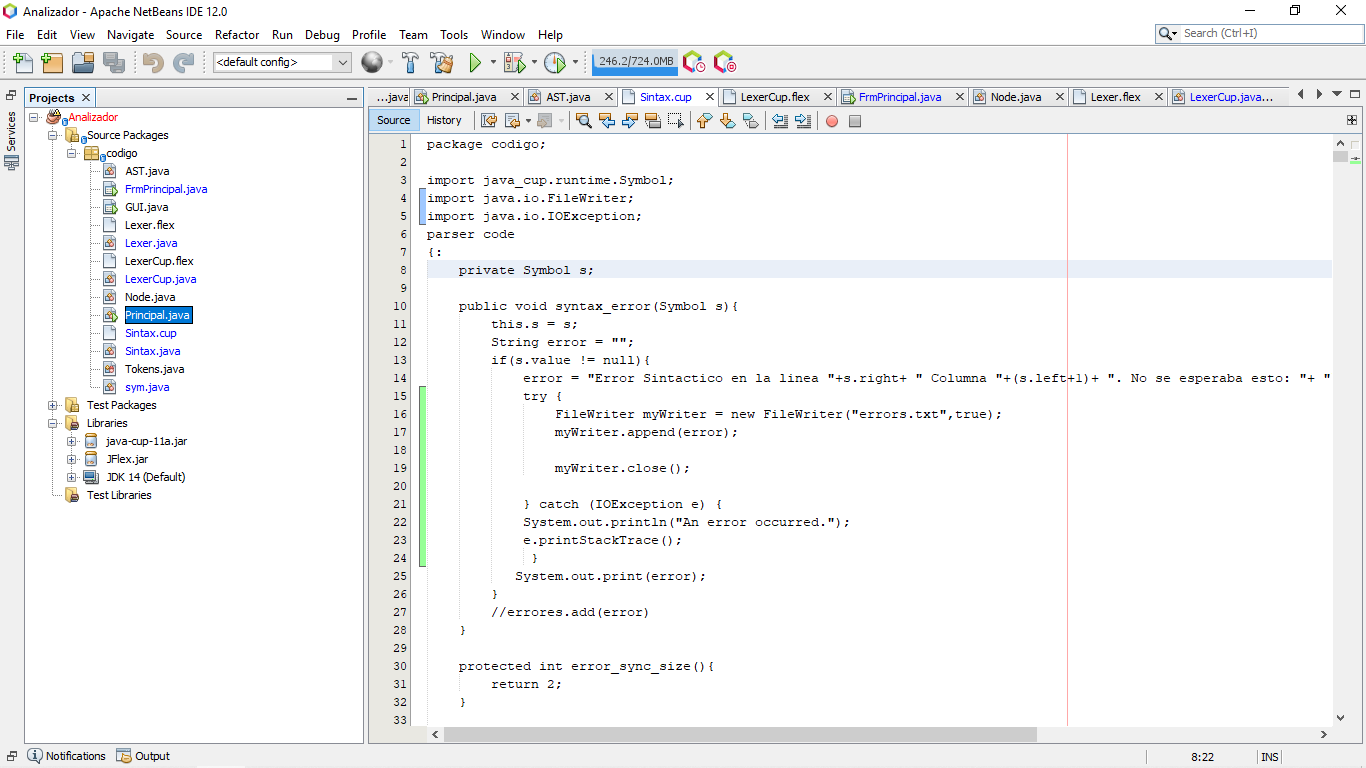


Si se modifica la gramatica ejecuta el Principal.java para actualizar el analizador lexico y sintactico.

Posteriormente ejecuta el FrmPrincipal.java y disfruta del IDE.



///////////////



Importar en este orden.