**Compiladores**

**Practica 3 Tabla de símbolos**

Primero compilar y ejecutar hoc3

Agregar una tabla de símbolos para permitir nombres de variables de mas de una letra y agregar a la gramática la producción para el operador de asignación. Para la tabla de símbolos use una lista simplemente ligada ( en leng C o Java) o una tabla hash ( de las librerias de Java).

Tambien si es posible agrege **builtins**. Por ejemplo:

Para números **complejos** agregue exponencial, seno , coseno y potencia

Para **polinomios**  agregue binomio de newton y serie geometrica

Para **conjuntos** agregue conjunto potencia

Para **matrices** agregue exponencial y potencia

**Litle quilt**

Las variables servirán para guardar colchitas

**Calculadora** para **complejos**

Las variables servirán para guardar números **complejos**

**Calculadora** para **polinomios**

Las variables servirán para guardar **polinomios**

**Calculadora** para **conjuntos**

Las variables servirán para guardar **conjuntos**

**Calculadora** para **vectores**

Las variables servirán para guardar **vectores**

**Calculadora** para **matrices**

Las variables servirán para guardar **matrices**

**Calculadora** para **racionales**

Las variables servirán para guardar **racionales**

**Calculadora** para **numeros grandes**

Las variables servirán para guardar **numeros grandes**

**Generador de código en ensamblador**

Permitir que el usuario escriba expresiones aritméticas que involucren variables y luego generar código ensamblador para dichas expresiones. Nota el código se generara para que sea ensamblado usando NASM (Netwide Assembler)

**Calculadora booleana**

Permitir que el usuario escriba una expresión booleana que involucre variables y luego imprimir su **tabla de verdad**. Nota si la expresión tiene n variables la tabla tendrá 2n renglones.

**Derivador Simbólico**

Agregar una tabla de símbolos para guardar los nombres de la funciones seno, coseno, tangente, logaritmo, exponencial para que se puedan derivar expresiones que involucran dichas funciones

**Robot**

Modificar la gramática y las acciones gramaticales para que se puedan sumar trayectorias del robot y para que una trayectoria se pueda almacenar en una variable. Incluir comandos para dibujar la trayectoria del robot y para levantar y bajar su pluma.

Nota: es mas fácil dibujar la trayectoria del robot si se usa java.

**Graficador de funciones**

Escribir una especificación de yacc para tabular funciones polinomiales seno, coseno, tangente, logaritmo, exponencial para ello construya el árbol sintáctico que representa la expresión y luego evalúelo para cada valor de la variable x y luego dibuje el punto correspondiente en pantalla.

Nota: Utilice un rango por default donde no se indetermine la función y es mas fácil dibujar la función si se usa java.

**Estrategia para la practica 3**

Las siguientes opciones tienen continuación practicas 1 a 7

complejos

polinomios

conjuntos

vectores

matrices

racionales

numeros grandes

Litle quilt

Generador de código en ensamblador

En lo que sigue supondremos que eligió la opción de complejos en la practica uno.

Si eligió otra opción sustituya complejos por la opción que haya elegido.

Acciones a realizar

-Copie los archivos en la carpeta de HOC3 a una carpeta nueva de nombre practica2

-Copie los archivos en la carpeta de su practica 1 a la carpeta practica2

-Modifique la parte val de la union u en la estructura Symbol en el archivo hoc.h para que trabaje con Complejos

-Modifique los parámetros y código de la función install para que trabaje con Complejos

-Modifique la parte func de la estructura en el archivo init.c para que los builtins trabajen con Complejos

-Modifique yylex() para que trabaje con Complejos puede usar LEX

ws [ \t]+

digits [0-9]

number (0|[1-9]+{digits}\*)\.?{digits}\*

im [i]

complexnum {ws}\*[-]\*{ws}\*{number}{ws}\*[+|-]{ws}\*{number}{ws}\*{im}{ws}\*

%%

{complexnum} {

double r, im;

RmWs(yytext);

sscanf(yytext,"%lf %lf", &r, &im);

yylval=creaComplejo(r, im);

return CNUMBER;

}

-Modifique val en %union en la especificación de YACC para que trabaje con Complejos

-Modifique la gramática para que trabaje con Complejos en particular modifique los tokens y las acciones gramaticales para que trabajen con Complejos.