Tecnológico de Monterrey

Documentación del compilador Vimo

Diseño de Compiladores

Desarrolladores:

Moisés Fernández Zárate A01197049

Víctor Andrés Villarreal Grimaldo A01039863

Descripción del proyecto

Propósito

Este proyecto tiene como objetivo desarrollar un lenguaje de programación de paradigma imperativo, cuya estructura sea muy similar a la manera en la que se desarrolla utilizando C++ o C, por mencionar algunos de los ejemplos más representativos. Tendrá la capacidad de realizar operaciones básicas como en esos lenguajes, por ejemplo, la declaración de variables y funciones, el uso de estatutos, etc. Asimismo, tendrá como característica adicional el desarrollo de videojuegos 2D con la implementación de objetos y funciones predefinidas.

Alcance

El resultado final de este proyecto sería un compilador de un lenguaje de programación de paradigma imperativo, como también una máquina virtual que sea capaz de ejecutar el archivo recibido una vez que haya sido compilado. El compilador podrá realizar un análisis léxico, sintáctico, semántico, generación de código intermedio con el uso de cuádruplos y además retroalimentación en los errores del desarrollador. Por otro lado, la máquina virtual será la encargada de la ejecución del código, produciendo un resultado basado en los cuádruplos generados. El lenguaje de programación que será compilado, va a poder ser utilizado para declarar variables y funciones, utilizar arreglos, realizar asignaciones, llamadas a funciones, el manejo de parámetros en las funciones, el uso de estatutos, expresiones aritméticas y de lógica, y por último, el uso de objetos y funciones predefinidas para el desarrollo de videojuegos 2D.

Análisis de Requerimientos

Lista de Requerimientos Funcionales

ID	**Requerimiento**	**Descripción**
RF0001	Declaración de variables	El sistema debe permitir al usuario declarar variables de cualquiera de los tipos de datos soportados por nuestro lenguaje. Los tipos de datos básicos son int, float, bool, char, string y además los tipos de objetos predefinidos son Square, Circle, Image, Text y Background.
RF0002	Declaración de funciones	El sistema debe permitir al usuario declarar funciones cuyo tipo de retorno sea cualquiera de los tipos de datos soportados por nuestro lenguaje.

		Además las funciones pueden tener cualquier número de parámetros, los cuales pueden ser variables de cualquier tipo, incluso arreglos.
RF0003	Declaración de arreglos	El sistema debe permitir al usuario declarar arreglos con un tamaño específico dado por el programador y de cualquiera de los tipos de datos soportados por nuestro lenguaje.
RF0004	Uso de comentarios	El sistema debe permitir al usuario escribir comentarios en el código utilizando la secuencia de caracteres “//”.
RF0005	Llamadas a funciones	El sistema debe permitir al usuario hacer llamadas a funciones que fueron previamente declaradas.
RF0006	Uso de expresiones aritméticas	El sistema debe permitir al usuario realizar operaciones aritméticas con los símbolos “+”, “-”, “*” y “/”.
RF0007	Uso de expresiones de lógica	El sistema debe permitir al usuario realizar operaciones de lógica con los símbolos “&&” (AND) y “||” (OR).
RF0008	Uso de expresiones relacionales	El sistema debe permitir al usuario realizar operaciones relaciones con los símbolos “>”, “<” y “==”.
RF0009	Uso de estatutos	El sistema debe permitir al usuario utilizar los estatutos “FOR”, “WHILE”, “IF”, “IF-ELSE”, “RETURN” y “PRINT”.
RF0010	Declaración de objetos predefinidos	El sistema debe permitir al usuario declarar objetos de cualquiera de los siguientes tipos: Square, Circle, Image, Text y Background.
RF0011	Uso de métodos predefinidos	El sistema debe permitir al usuario utiliza los métodos predefinidos, los cuales son: Render(), Clear(), Update(), KeyPressed(), CheckCollision(), Pow() y Sqrt().
RF0012	Crear una ventana para el videojuego	El sistema debe permitir al usuario crear una ventana en donde se podrán ver sus objetos renderizados.
RF0013	Renderizar los objetos predefinidos	El sistema debe permitir al usuario renderizar cualquier número de objetos predefinidos que haya declarado el usuario en una ventana.
RF0014	Recibir entrada del teclado	El sistema debe permitir al usuario recibir entrada

		del teclado, detectando cuando se presionan las teclas “W”, “A”, “S”, “D”, y las cuatro teclas de las flechas.
RF0015	Checar las colisiones entre objetos	El sistema debe permitir al usuario checar si hay colisiones entre objetos, únicamente entre objetos predefinidos de tipo Square y Circle.

Test Cases

Existen tres carpetas en las cuales se desarrollaron múltiples casos de prueba para verificar la funcionalidad del compilador junto con la máquina virtual. En la carpeta **“test”** están los programas realizados para probar las funcionalidades conforme se iban desarrollando. En la carpeta **“examples”** están los programas en los que se prueban implementaciones de algoritmos más específicos para realizar las pruebas básicas a desarrollar de la rúbrica. En la carpeta **“gameexamples”** se encuentran programas para probar la funcionalidad de los objetos y métodos predefinidos para el desarrollo de videojuegos 2D.

1. **test**
    a. **test1.vm**
        i. Declaración inicial del programa con su identificador
        ii. Declaración de variables globales
        iii. Declaración de variables locales
        iv. Declaración de funciones
        v. Manejo de parámetros en las funciones
        vi. Llamadas a funciones
        vii. Uso de estatuto condicional “if”
        viii. Uso de expresiones relacionales
        ix. Uso de estatutos “print” y “return”
    b. **test2.vm**
        i. Uso de estatuto print recibiendo como parámetro una función que tiene un valor de retorno diferente a “void”
        ii. Uso de estatuto “if-else”
        iii. Uso de estatuto “for”
        iv. Uso de recursividad
    c. **test3.vm**
        i. Declaración de arreglos de tipos de datos básicos con tamaño definido
        ii. Uso de arreglos como parámetro
        iii. Uso de estatutos “if” y “for” anidados
        iv. Acceso a los datos de los arreglos
        v. Asignación de valores a las casillas de los arreglos
    d. **test4.vm**
        i. Llamada a función
    e. **test5.vm**

i. Declaración de variables de objetos predefinidos
ii. Declaración de arreglo de tipo de alguno de los objetos predefinidos
iii. Asignación de datos a atributos de objeto predefinido
iv. Asignación de datos a casilla de arreglo de tipo objeto predefinido
v. Acceso a datos de arreglo de tipo de objeto predefinido
vi. Uso de estatuto “print” con objeto predefinido
vii. Declaración a funciones con objeto predefinido como tipo de retorno
viii. Uso de objetos predefinidos como parámetros en funciones
ix. Uso de estatuto print con atributo de objeto predefinido
x. Uso de estatuto print con casilla de arreglo de tipo objeto predefinido

f. test6.vm

i. Declaración de variables globales de tipo cualquier objeto predefinido
ii. Uso de función predefinida “Clear()”
iii. Uso de función predefinida “CheckCollision()” con objetos predefinidos de tipo Square y Circle como parámetros
iv. Uso de función predefinida “Update()”
v. Uso de función predefinida “Render()” con parámetro siendo cualquiera de los objetos predefinidos

Proceso seguido para el desarrollo

La manera en la que trabajamos fue al principio a dividirnos tareas que platicamos y planeamos para luego juntarlas, por eso es que al principio hay commits de cada uno. Sin embargo, después cuando se fue acercando más la fecha hicimos casi todo juntos utilizando Visual Studio Code con la extensión Live Share para poder trabajar en el código al mismo tiempo, poder ver dónde está trabajando cada uno y poder hacer los cambios necesarios y las pruebas al mismo tiempo.

Quedaba muy poco tiempo cuando volvimos a escribir código otra vez, por lo que acordamos despertarnos a partir de mediados de mayo todos los días a las 7 de la mañana y trabajar todo el día en nuestro tiempo libre cuando no tuviéramos clases o en tiempo de comida. A pesar de que estuvimos trabajando juntos gracias a Live Share, algunas cosas logramos trabajarlas en paralelo, aunque de todos modos tuvimos que platicar y planear lo que hacía cada quien por separado, luego lo revisamos con la estrategia de Peer Review, y por último, realizando pruebas.

La herramienta que utilizamos para el control de versiones fue GitHub, ya que es la herramienta que siempre hemos utilizado durante el curso de la carrera, la que sabemos utilizar y con la que nos sentimos con confianza.

Avance 1 (9 de abril del 2021)

- No hicimos una entrega en el primer avance porque todavía no teníamos la propuesta del proyecto aceptada.

Avance 2 (15 de abril del 2021)

- Se cambió el token de los comentarios de ‘#’ a ‘//’ para no tener problemas al momento de utilizar los valores de los colores en hexadecimal para los objetos.

- Se añadieron los siguientes tokens para las clases predefinidas por nosotros: SQUARE, CIRCLE, IMAGE, TEXT, BACKGROUND.
- Además se agregaron los siguientes tokens para el manejo de la gramática: VOID, PRINT, RETURN, TRUE, FALSE, BOOL.
- Se agregaron los tokens de las funciones especiales.
- Se agregaron los tokens de los square brackets.
- Se modificaron los tokens de las operaciones aritméticas para que fueran individuales por cada tipo de operación.
- Se cambió el diagrama de sintaxis de PROGRAMA para que inicie con program en lugar de game para no especificar que su uso sea únicamente para videojuegos.
- Se generó el código del archivo .bnf y los archivos resultantes de la compilación con el kit de compilador de “gocc” para Go.

Avance 3 (23 de abril del 2021)

- Se separaró el token de operadores aritméticos en diferentes tokens individuales para un manejo adecuado de los operadores en la sintaxis y en la semántica.
- Se investigó acerca de un cambio grande que haremos en el manejo de sintaxis, en el cual utilizaremos AST (Abstract syntax tree) pues éste se utiliza en la herramienta para golang de gocc.
- Se definieron los tipos que se definirán en el AST y el código del mismo.

La primera semana no hicimos la entrega semanal debido a que aún no teníamos aceptada la propuesta, pero las siguientes dos semanas sí hicimos la entrega semanal debida. Sin embargo, a partir de la cuarta semana ya no hicimos entrega semanal debido a que nuestro avance estaba muy atrasado, no teníamos el trabajo esperado para el avance semanal y así nos fuimos porque realmente no había avance en código. Lo que estuvimos haciendo fue investigación de la manera correcta de implementar un AST, también hicimos modificaciones en el léxico y en la sintaxis, pero realmente nada fue en código. No escribimos código otra vez hasta mediados de mayo, que fue cuando empezamos a subir commits nuevamente como se muestra en la tabla a continuación.

Tabla de commits

Número	Fecha	Descripción	Link al commit
1	15/04/2021	Se creó el repositorio con el readme	Commit
2	15/04/2021	Se agregaron tokens del léxico	Commit
3	15/04/2021	Se borró un file de gocc	Commit
4	16/04/2021	Se añadieron más tokens y la primera mitad de la parte de la sintaxis	Commit
5	16/04/2021	Se añadió la otra mitad de la parte de análisis de sintaxis	Commit

6	16/04/2021	Se hizo un cambio en el nombre del archivo BNF	Commit
7	16/04/2021	Se hizo un merge de la branch “main” a la branch “diagramasvic”	Commit
8	16/04/2021	Se hizo un merge pull request en la branch “diagramasvic”	Commit
9	16/04/2021	Se añadieron más tokens	Commit
10	16/04/2021	Se hizo una corrección de un error de tipografía	Commit
11	16/04/2021	Se logró compilar gocc con nuestro archivo BNF	Commit
12	17/05/2021	Se reestructuró el orden de los archivos y las carpetas	Commit
13	17/05/2021	Se reestructuró el orden de los archivos y las carpetas	Commit
14	17/05/2021	Se añadió el directorio de las variables	Commit
15	17/05/2021	Se añadió el juego Ping Pong que mostramos para la propuesta	Commit
16	17/05/2021	Se añadió el directorio de las funciones	Commit
17	17/05/2021	Se hizo un merge de la branch “main”	Commit
18	17/05/2021	Se creó un folder para guardar el archivo donde se declaran los tipos de datos	Commit
19	17/05/2021	Se modificó el directorio de variables	Commit
20	18/05/2021	Se hizo el primer avance al AST	Commit
21	18/05/2021	Se hizo un merge de la branch “main”	Commit
22	18/05/2021	Se creó el archivo para los tipos	Commit
23	18/05/2021	Se agregaron los objetos predefinidos en los tipos	Commit
24	19/05/2021	Se hicieron modificaciones al AST, más estatutos	Commit
25	19/05/2021	Se hicieron modificaciones al AST, correcciones Assign y Block	Commit

26	19/05/2021	Se hicieron modificaciones al AST, funciones predefinidas, nueva estructura	Commit
27	19/05/2021	Se completó la primera versión del AST funcional	Commit
28	19/05/2021	Se creó la primera versión del generador de nodos del AST, el archivo astx	Commit
29	19/05/2021	Se hicieron modificaciones al astx, nueva estructura, nuevas funciones para generar nodos	Commit
30	20/05/2021	Se arregló el archivo BNF	Commit
31	20/05/2021	Se hicieron modificaciones al AST, nuevos métodos, nuevas estructuras y cambios a estructuras	Commit
32	20/05/2021	Se hicieron modificaciones al AST y al astx, nuevos nodos, funciones para crear nodos	Commit
33	21/05/2021	Se hicieron modificaciones al AST y al archivo BNF, nuevos nodos	Commit
34	21/05/2021	Se arregló método para crear nodo del astx	Commit
35	21/05/2021	Se corrigieron comentarios del astx	Commit
36	21/05/2021	Se hicieron cambios en el archivo BNF y en el archivo AST para probar la implementación de Expression	Commit
37	21/05/2021	Se hicieron más cambios en los archivos BNF y AST para probar Expression	Commit
38	21/05/2021	Se corrigió la gramática	Commit
39	25/05/2021	Se hicieron cambios a los archivos BNF y astx	Commit
40	25/05/2021	Se hizo la primera versión para el análisis de la semántica	Commit
41	26/05/2021	Se hicieron modificaciones al AST, al astx y al archivo BNF, nuevos nodos e implementación de funciones para crear nodos, primera versión funcional de AST con archivo vacío	Commit
42	26/05/2021	Se hicieron modificaciones a los archivos AST, astx y BNF, nuevos nodos, cambios a nodos y nuevas funciones para crear nodos	Commit

43	27/05/2021	Se hicieron modificaciones a los archivos AST, astx y BNF, nuevos nodos, cambios a nodos y a funciones, nuevas funciones para crear nodos, versión funcional del AST	Commit
44	27/05/2021	Merge pull request de branch “ast” a branch “main”	Commit
45	27/05/2021	Modificaciones a la semántica y pruebas con el AST	Commit
46	28/05/2021	Modificaciones a la semántica, primera versión funcional	Commit
47	31/05/2021	Primera versión funcional de la generación de código intermedio	Commit
48	01/06/2021	Se hicieron modificaciones al AST, versión funcional	Commit
49	01/06/2021	Se hicieron modificaciones a los archivos del análisis semántico, versión funcional	Commit
50	01/06/2021	Se hicieron modificaciones a la memoria, versión funcional	Commit
51	02/06/2021	Se hicieron modificaciones a la memoria virtual, versión funcional	Commit
52	02/06/2021	Se hicieron cambios a múltiples archivos y se añadió la Game Engine para utilizar Pixel para el desarrollo de videojuegos 2D, versión funcional de compilador con máquina virtual	Commit
53	02/06/2021	Se hizo merge pull request de la branch “vm” a la branch “main”	Commit
54	02/06/2021	Se añadió imagen del README	Commit
55	02/06/2021	Se borró imagen del README	Commit
56	02/06/2021	Se añadió imagen del README	Commit
57	02/06/2021	Se borró imagen del README	Commit
58	02/06/2021	Se añadió imagen del README	Commit
59	02/06/2021	Se añadió gif del README	Commit
60	02/06/2021	Se añadió gif del README	Commit
61	02/06/2021	Se añadió el README, también se añadieron	Commit

		añadieron pruebas y código para el README	
62	02/06/2021	Se hizo merge pull request de la branch “vm” a la branch “main”	Commit

Reflexiones

- Moisés Fernández Zárate

  Este proyecto ha sido uno de los proyectos más retadores que he tenido en mi vida, si no es que el más retador de todos, ya que desde el principio decidimos utilizar un lenguaje de programación que mi compañero y yo nunca habíamos utilizado, además de eso, aplicar lo aprendido en clase utilizando una herramienta completamente nueva para los dos. También tuvimos muchos problemas y no fue de mucha ayuda haber dejado una gran parte del proyecto para el final del semestre, pero a pesar de todos estos obstáculos, logré aprender mucho a lo largo del desarrollo de este proyecto, como tener una mejor planeación al principio del desarrollo de los proyectos, a leer correctamente y por mucho documentación de lenguajes de programación y herramientas nuevas. En fin, el mayor aprendizaje que me llevo es lograr comprender de mejor manera o al menos tener una idea de lo que está pasando cuando estoy desarrollando código y poder darle más importancia a la optimización y a las buenas prácticas.

- Víctor Andrés Villarreal Grimaldo

  Este proyecto requirió de una gran cantidad de esfuerzo por mi parte, especialmente porque decidimos probar un nuevo lenguaje de programación (Go) y por ello tuvimos que usar una herramienta de la cual no había mucha documentación en internet comparada con otras más populares. Sin embargo, ya habiendo concluido el proyecto me siento muy orgulloso del resultado y aprendí varias lecciones sobre mis hábitos de trabajo tanto colaborativo como individual. No sería exageración decir que este trabajo es posiblemente el más complejo que he desarrollado. Además logré aprender a utilizar un nuevo lenguaje que considero que ahora domino a un nivel suficientemente bueno y me terminó gustando por lo que me gustaría seguir explorando con él. Pero sobretodo me gustaría seguir trabajando en el compilador para agregar más funcionalidades y mejoras en versiones futuras.

Descripción del lenguaje

Nombre del lenguaje

Vimo.

Principales características

Vimo es un lenguaje de programación imperativo capaz de realizar las actividades básicas de un lenguaje de este tipo, como por ejemplo, declarar variables y funciones, declarar arreglos, tener expresiones aritméticas, relacionales y de lógica, y usar diferentes tipos de estatutos como ciclos y condicionales. Asimismo, para el desarrollo de videojuegos 2D se pueden utilizar objetos y métodos predefinidos para renderizarlos en una pantalla y poder crear un videojuego sencillo.

Posibles errores

Listado de errores

Tipo de error
Error en la sintaxis
Se intenta asignar un valor a una variable que es de diferente tipo
La variable no existe
Se intenta acceder a un índice que está fuera del tamaño de la lista
La función no existe
La variable ya existe
La función ya existe
Límite de memoria excedido
El id está fuera del scope
Enviar un dato de diferente tipo al que se espera en el parámetro
Intentar utilizar un arreglo que no existe
El arreglo ya existe
No regresar un return al final de una función cuyo tipo de retorno no es void

Descripción del Compilador

Equipo de cómputo, lenguaje y utilerías

Se utilizó una computadora Windows de escritorio y una laptop Windows para desarrollar y correr el compilador con las siguientes especificaciones:

Desktop Windows 10, Intel Core i5-7400, 16 GB RAM, NVIDIA GeForce GTX 1050 Ti	Laptop Asus Windows 10, Intel Core i5-8300H, 8 GB RAM, Intel UHD Graphics 630

El lenguaje de programación utilizado fue **Go** y en éste se programaron todos los componentes.

Además se hizo uso del kit de compilador para Go llamado Gocc, el cual sirvió para generar en analizador léxico y sintáctico a partir de un BNF, y se definió un AST (Abstract syntax tree) para representar la estructura sintáctica del código del programa lo que nos permitió hacer varias pasadas al programa y separar el chequeo semántico y la generación de código, además de visualizar la estructura del programa en memoria.

Puesto que Go no tiene estructuras genéricas, el paquete de librerías genéricas ErrUtil ofrece un módulo de errores que nos permitió darle formato a errores y depuración de código.

Para las funcionalidad adicionales que hacen nuestro lenguaje una herramienta para desarrollo de videojuegos 2D se utilizó la librería de Pixel en Go.

Análisis léxico

Tokens y sus expresiones regulares:

- _letter: 'a' - 'z' | 'A' - 'Z';
- _digit: '0' - '9';
- _id : _letter {(_letter | _digit)};
- _integer: _digit {_digit};
- _float: _digit {_digit} '.' _digit {_digit};
- _string: "" {_digit | _letter | ' ' | '#' |'!' | '?' | '.'} "";
- _true: 't' 'r' 'u' 'e';
- _false: 'f' 'a' 'l' 's' 'e';
- _boolean: _true | _false;
- !ws : ' ' | '\t' | '\n' | '\r';
- !comment : '/' '/' { . } '\n';

- program: 'p' 'r' 'o' 'g' 'r' 'a' 'm';
- inttype: 'i' 'n' 't';
- floattype: 'f' 'l' 'o' 'a' 't';
- stringtype: 's' 't' 'r' 'i' 'n' 'g';
- booltype: 'b' 'o' 'o' 'l';
- chartype: 'c' 'h' 'a' 'r';
- voidtype: 'v' 'o' 'i' 'd';
- squaretype: 'S' 'q' 'u' 'a' 'r' 'e';
- circletype: 'C' 'i' 'r' 'c' 'l' 'e';
- imagetype: 'I' 'm' 'a' 'g' 'e';
- texttype: 'T' 'e' 'x' 't';
- backgroundtype: 'B' 'a' 'c' 'k' 'g' 'r' 'o' 'u' 'n' 'd';
- if: 'i' 'f';
- else: 'e' 'l' 's' 'e';
- while: 'w' 'h' 'i' 'l' 'e';
- for: 'f' 'o' 'r';
- print: 'p' 'r' 'i' 'n' 't';
- return: 'r' 'e' 't' 'u' 'r' 'n';
- relop: '<' | '>' | '<' '=' | '>' '=' | '<' '>' | '=' '=';
- logicalop: '&' '&' | '|' '|';
- plus: '+';
- minus: '-';
- mult: '*';
- div: '/';
- leftparenthesis: '(';
- rightparenthesis: ')';
- leftbracket: '{';
- rightbracket: '}';
- leftsqrbracket: '[';
- rightsqrbracket: ']';
- colon: ':';
- semicolon: ';';
- dot: '.';
- comma: ',';
- equals: '=';
- cteint: ['-'] _integer;
- ctefloat: ['-'] _float;
- ctestring: _string;
- ctebool: _boolean;
- ctechar: '\'' (_letter | _digit | ' ') '\'';

- id: _id;

Análisis de sintaxis

A continuación se describe la gramática libre de contexto utilizada para el análisis de sintaxis del lenguaje.

Programa
: program id semicolon leftbracket VarsOp rightbracket Functions

VarsOp
: Vars
| empty

Vars
: Type Ids semicolon Vars
| Type Ids semicolon

VarsDec
: Vars

Ids
: id comma Ids
| id

Params
: ParamsAux
| empty

ParamsAux
: Type id comma ParamsAux
| Type id

Functions
: FunctionsAux id leftparenthesis Params rightparenthesis Block Functions
| FunctionsAux id leftparenthesis Params rightparenthesis Block

FunctionsAux
: Type
| voidtype

Block
: leftbracket BlockAux rightbracket
| leftbracket rightbracket

BlockAux
: Statement
| Statement BlockAux

Statement
: VarsDec
| Assign semicolon
| Condition
| Return
| For
| While
| Write
| CallFunction semicolon

BasicType
: inttype
| floattype
| booltype
| stringtype
| chartype
| Object

Object
: squaretype
| circletype
| imagetype
| texttype
| backgroundtype

Expression
: Exp
| Exp Operations Expression

Operations
: relop

| logicalop

Exp
: Term
| Term plus Exp
| Term minus Exp

Term
: Factor
| Factor mult Term
| Factor div Term

Factor
: leftparenthesis Expression rightparenthesis
| Varcte

Assign
: id equals Expression
| Attribute equals Expression
| ListElem equals Expression

Write
: print leftparenthesis Expression rightparenthesis semicolon

Condition
: if leftparenthesis Expression rightparenthesis Block
| if leftparenthesis Expression rightparenthesis Block else Block

Return
: return Expression semicolon

For
: for leftparenthesis Assign semicolon Expression semicolon Assign rightparenthesis Block

While
: while leftparenthesis Expression rightparenthesis Block

CallFunction
: id leftparenthesis CallFunctionAux rightparenthesis

| id leftparenthesis rightparenthesis

CallFunctionAux
: Expression
| Expression comma CallFunctionAux

Varcte
: id
| cteint
| ctefloat
| ctestring
| ctechar
| ctebool
| ListElem
| Attribute
| CallFunction

ListElem
: id leftsqrbracket Expression rightsqrbracket

Type
: BasicType
| BasicType leftsqrbracket cteint rightsqrbracket

Attribute
: id dot id

Generación de código intermedio y Análisis Semántico

Código de operación y direcciones virtuales asociadas a los elementos del código.
o Diagramas de Sintaxis con las acciones correspondientes marcadas sobre ellos (puntos neurálgicos).
o Breve descripción de cada una de las acciones semánticas y de generación de código (no más de 2 líneas).
o Tabla de consideraciones semánticas (combinaciones factibles y errores de tipo).

Como fue mencionado anteriormente, para el desarrollo de este compilador se optó por hacer uso de un árbol abstracto de sintaxis o **AST** por lo que en vez de realizar las acciones de análisis semántico y generación de código mientras se recorría la gramática, se recorre el árbol sintáctico 5 veces para realizar el análisis y generar los cuádruplos. Estos 5 recorridos son la construcción de tablas de funciones y de

variables, análisis semántico del scope de las variables y las funciones, chequeo de tipos, asignaciones de direcciones de memoria virtual y finalmente la generación de cuádruplos.

Los cuádruplos generados tienen el siguiente formato:

Operación - lop - rop - resultado

Dependiendo de la operación los parámetros utilizados y la función que tienen es diferente. Dichas operaciones son las siguientes:

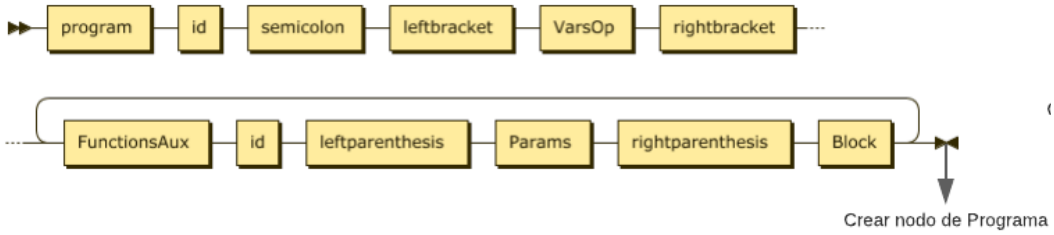
- **Goto**: Señala el la dirección del índice del cuádruplo con la operación al que debe ir el flujo de ejecución. Puede quedar pendiente la dirección para ser agregado luego de sacarla del stack.
- **Init**: Inicializa las variables en la memoria virtual con el valor default dependiendo del tipo de variable.
- **Ret**: Mata el activation record actual, obtiene el valor de retorno, reactiva las variables locales y si la función no es void actualiza la dirección obtiene la dirección donde se guarda el valor de retorno con el valor obtenido.
- **Add**: Suma dos números y guarda el resultado en un temporal.
- **Sub**: Resta dos números y guarda el resultado en un temporal.
- **Mult**: Multiplica dos números y guarda el resultado en un temporal.
- **Div**: Divide dos números y guarda el resultado en un temporal.
- **And**: Guarda en un temporal el resultado de la operación lógica “AND” entre dos booleanos.
- **Or**: Guarda en un temporal el resultado de la operación lógica “OR” entre dos booleanos.
- **Not**: Guarda en un temporal el resultado de la operación lógica “NOT” de un booleano.
- **Gt**: Calcula si un número es mayor que otro y guarda en un temporal el resultado booleano
- **Lt**: Calcula si un número es menor que otro y guarda en un temporal el resultado booleano.
- **Equal**: Calcula si un número es igual a otro y guarda en un temporal el resultado booleano.
- **Print**: Imprime en consola el valor recibido como parámetro.
- **Era**: Crea el récord de activación para una función.
- **Param**: Añade los parámetros recibidos al récord de activación.

- **Call**: Obtiene la llamada a la función actual y la que está apunto de pasar y la quita del stack, y al final obtiene la dirección del cuádruplo al que tiene que dirigirse para llevar a cabo la función que se llama.
- **Assign**: Asigna un valor recibido como parámetro en otra posición de memoria recibida.
- **GotoT**: Señala el la dirección del índice del cuádruplo con la operación al que debe ir el flujo de ejecución si el operando es verdadero. Puede quedar pendiente la dirección para ser agregado luego de sacarla del stack.
- **GotoF**: Señala el la dirección del índice del cuádruplo con la operación al que debe ir el flujo de ejecución si el operando es falso. Puede quedar pendiente la dirección para ser agregado luego de sacarla del stack.
- **Pow**: Eleva un número a una potencia recibida y guarda el resultado en un temporal.
- **Sqrt**: Calcula la raíz cuadrada de un número recibido y guarda el resultado en un temporal.
- **CheckBound**: Realiza el chequeo de que un índice de una lista esté dentro del rango del tamaño de dicha lista.
- **AddAddr**: Suma un offset a un address donde lop es el address, rop es el offset y se guarda el resultado.
- **AssignIndex**: Asigna un valor recibido como parámetro en otra posición de memoria recibida que corresponde a una posición en un índice dentro de una lista.
- **AssignIndexInv**: Asigna un valor recibido como parámetro, que corresponde a una posición en un índice dentro de una lista, en otra posición de memoria recibida.
- **Render**: Renderiza en pantalla el objeto recibido en lop.
- **Clear**: Limpia la pantalla de todos los objetos renderizados.
- **Update**: Actualiza la pantalla con los nuevos renderizados para la siguiente frame.
- **KeyPressed**: Obtiene el valor booleano del evento en el que una tecla recibida en lop sea presionada por el usuario y lo guarda en resultado.
- **CheckCollision**: Obtiene el valor booleano del evento de dos objetos intersectando en la pantalla, recibe los objetos en lop y ro y guarda el resultado.

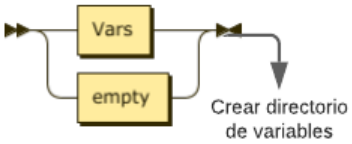
Diagramas de Sintaxis con acciones semánticas

Para generar el árbol abstracto de sintaxis se realizan las acciones que pueden ser visualizadas en estos diagramas:

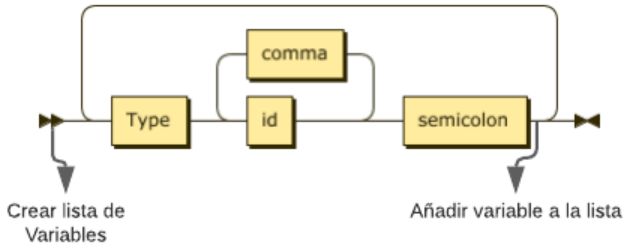
Programa



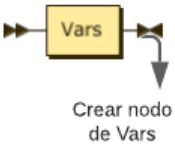
VarsOp



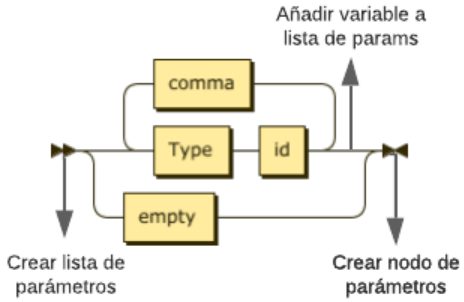
Vars:



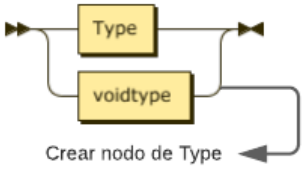
VarsDec:



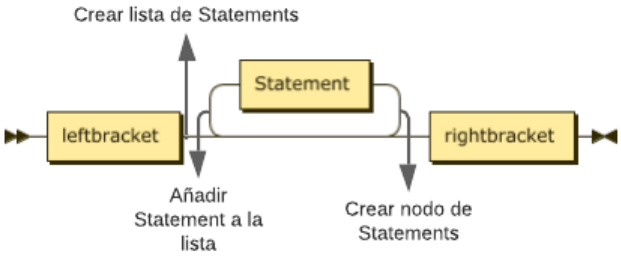
Params:



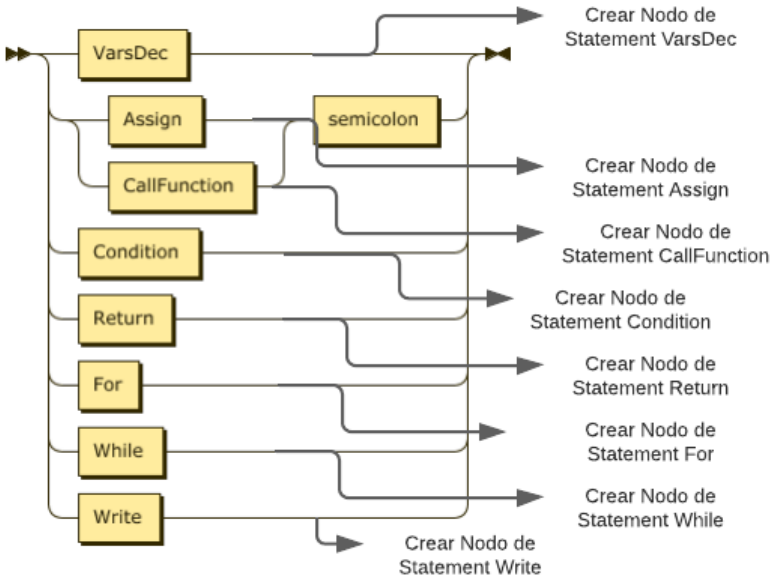
FunctionsAux:



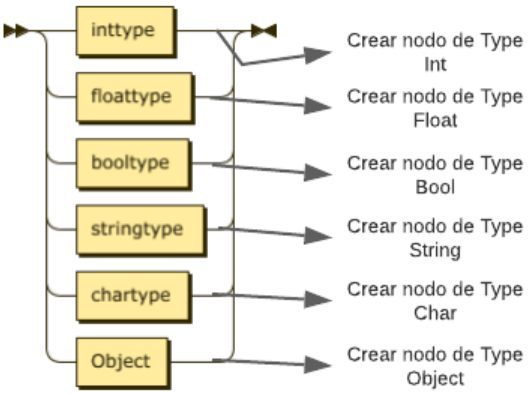
Block:



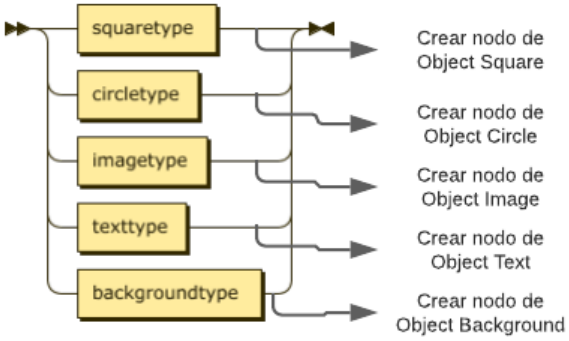
Statement:



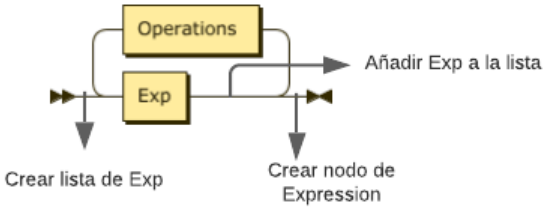
BasicType:



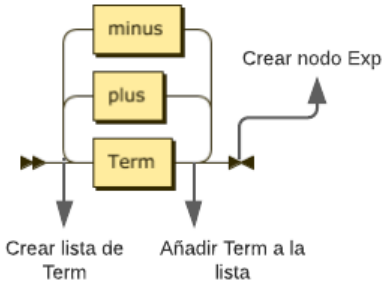
Object:



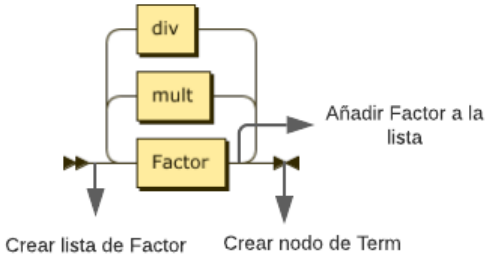
Expression:



Exp:

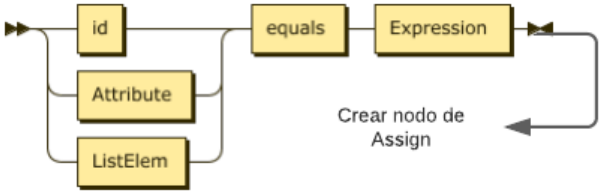


Term:



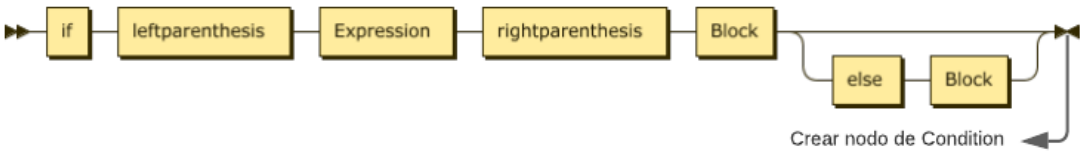
Factor:

Assign:

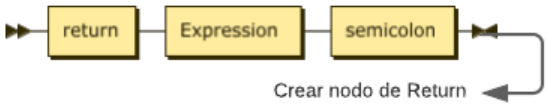


Write:

Condition:



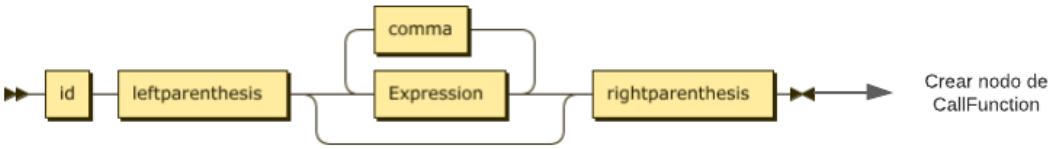
Return:



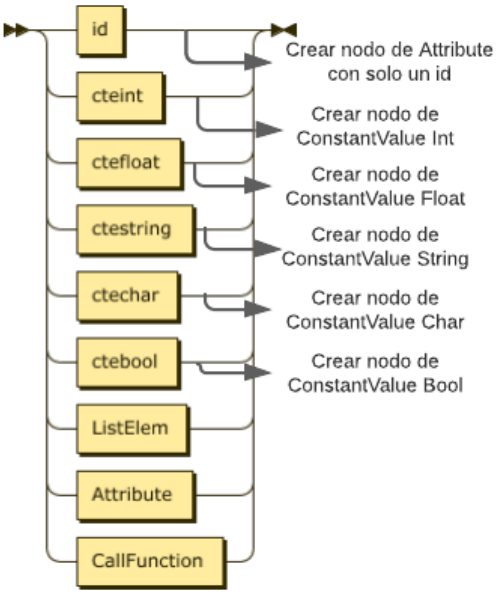
For:

While:

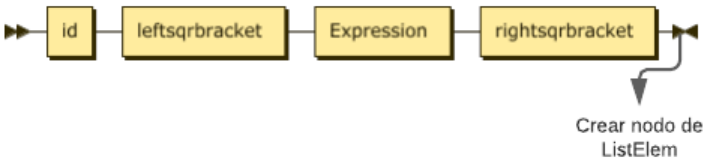
CallFunction:



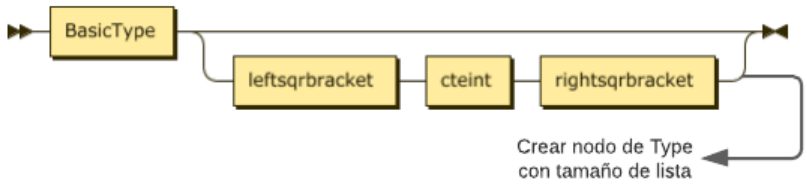
Varcte:



ListElem:



Type:



Attribute:

Árbol abstracto de sintaxis

Cada acción semántica que se ve en los diagramas anteriores sirve para construir el AST. Para ello definimos una estructura para el AST que indica los nodos del árbol y sus respectivos hijos. Dichos nodos permiten representar los diferentes elementos de nuestro programa de forma jerárquica. Los nodos definidos por nuestra estructura son:

- Program: Nodo raiz.
    - Functions ([]Function): Contiene todas las funciones de nuestro programa.
    - Id (string): Nombre del programa.
    - Vars ([]VarEntry): Contiene todas las variables globales del programa
- Function: Contiene los datos de las funciones
    - id (string): nombre de la función
    - key (string): Llave en la tabla de funciones
    - Params ([]VarEntry): Parámetros de la función
    - Statements([]Statement): Estatutos dentro de la función
    - token (Token): Apuntador a la posición del código
- Statement: Una interface utilizada para los diversos estatutos que pueden ser:
    - Vars
    - Assign
    - Condition
    - Write
    - Return
    - For
    - While
    - FunctionCall

- Factor: Contiene una expresión y una constante
    - Exp (Expression): Nodo de Expression
    - Cv (Constant): Nodo de Constant
    - Tok (Token): Apuntador a la posición de código
- Term:
    - Facs ([]Factor): Lista de Factores
    - Ops ([]string): Lista de Operaciones
    - Tok (Token): Apuntador a la posición de código
- Exp:
    - Terms ([]Term): Lista de Términos
    - Ops ([]string): Lista de Operaciones
    - Tok (Token): Apuntador a la posición de código
- Expression:
    - Exps ([]Exp): Lista de Exps
    - Ops ([]string): Lista de Operaciones
    - Tok (Token): Apuntador a la posición de código
- Attribute:
    - objId (string): Nombre del la variable
    - varId (string): Nombre del atributo en caso de ser un objeto
    - Index (Expression): índice en caso de ser una lista
    - Tok (token): Apuntador a la posición de código
- ListElem:
    - Id (string):Nombre de la lista
    - Index (Expression): índice de la posición de la lista
    - Tok (Token): Apuntador a la posición de código
- Vars
    - Variables ([]VarEntry): Lista de entradas de variables
    - Tok (Token): Apuntador a la posición de código
- Assign:
    - Attr (Attribute): Elemento al que se le desea hacer la asignación
    - Exp (Expression): Expresión que contiene el valor a asignar
    - Tok (Token): Apuntador a la posición de código
- Write:
    - Exp (Expression): Expresión que se desea imprimir en consola.
    - Tok (Token): Apuntador a la posición de código
- Condition:
    - Exp (Expression): Expresión booleana a evaluar
    - Stmts ([]Statement): Estatutos a realizar dentro del “if”
    - elseStmts ([]Statement): Estatutos a realizar dentro del “else”
    - Tok (Token): Apuntador a la posición de código

- Return:
    - Exp (Expression): Expresión a regresar en el return
    - Tok (Token): Apuntador a la posición de código
- For:
    - Init (Assign): Asignación inicial de un valor a una variable
    - Cond (Expression): Expresión booleana que define cuándo termina el For
    - Op (Assign): Operación a realizar cada iteración
    - Blck ([]Statement): Lista de estatutos dentro del For
    - Tok (Token): Apuntador a la posición de código
- While
    - Exp (Expression): Expresión booleana
    - Blck ([]Statement): Lista de estatutos dentro del while
    - Tok (Token): Apuntador a la posición de código
- FunctionCall:
    - Id (String): nombre de la función a llamar
    - Params ([]Expression): lista de los parámetros
    - Tok (Token): Apuntador a la posición de código
- Constant: Una interface utilizada para los diversos valores “constantes” que pueden ser:
    - ConstantValue
    - Attribute
    - FunctionCall
    - ListElem

Descripción breve de acciones semánticas

- Program
    - NewProgram: Crea el nodo raíz del AST del programa.
- Vars
    - AppendVarsList: Agrega la variable a la lista de Vars
    - NewVarsList: Crea una lista de Vars y agrega la variable que acaba de parsear
- Id
    - AppendIdList: Agrega el Id a la lista de Ids
    - NewIdList: Crea una lista de ids y agrega el id que acaba de parsear
- Params
    - AppendParamsList: Agrega el parámetro a la lista de parámetros
    - NewParamsList: Crea una lista de parámetros y agrega el parámetro que acaba de parsear

- Functions
    - AppendFunction: Agrega la Función a la lista de funciones
    - FirstFunction: Crea un Nodo de función que contiene un arreglo también creado que contendrá todas las funciones
- Block
    - NewStatementsList: Crea una lista de Statements
    - AppendStatementsList: Agrega el Statement a la lista de Statements
- Statement
    - NewStatement: Crea y regresa un Statement
- BasicType
    - NewType: Crea un nuevo Type de un BasicType
- Object
    - NewType: Crea un nuevo Type de un ObjectType
- Expression
    - NewExpression: Crea un nodo Expression
    - AppendExpression: Añade un Exp y una operación a las listas de Exps y Ops de una Expression.
- Exp
    - NewExp: Crea un nodo Exp
    - AppendExp: Añade un Term y una operación a las listas de Term y Ops de una Exp.
- Term
    - NewTerm: Crea un nodo Term
    - AppendTerm: Añade un Factor y una operación a las listas de Factor y Ops de un Term.
- Factor
    - NewFactor: Crea el nodo Factor con Expression adentro
    - NewVCFactor: Crea el nodo Factor con varcte adentro
- Assign
    - NewAssignWithoutAttr: Crea nodo de Statement tipo Assign para un id
    - NewAssignWithAttr: Crear nodo de Statement tipo Assign para un Attribute
    - NewAssignWithIndex: Crear nodo de Statement tipo Assign con para listas con un índice
- Write
    - NewWrite: Crea un nodo de Statement tipo Write
- Condition
    - NewConditionNoElseStmts: Crea nodo de Statement tipo Condition con if
    - NewCondition: Crea nodo de Statement tipo Condition con if y else
- Return

    - NewReturn: Crea un nodo de Statement tipo Return
- For
    - NewFor: Crea un nodo de Statement tipo For
- While
    - NewWhile: Crea un nodo de Statement tipo While
- CallFunction
    - NewFunctionCall: Crea un nodo FunctionCall sin parámetros
    - NewFunctionCallId: Crea un nodo FunctionCall con parámetros
- CallFunctionAux
    - NewArgumentExpression: Crea una lista de Expressions
    - AppendArgumentExpression: Añade un Expression a la lista de Expressions
- Varcte
    - NewAttribute: Crea un nodo Constant de tipo Attribute con solo un id
    - NewConstantInt: Crea un nodo Constant de tipo ConstantValue de tipo int
    - NewConstantFloat: Crea un nodo Constant de tipo ConstantValue de tipo float
    - NewConstantBool: Crea un nodo Constant de tipo ConstantValue de tipo bool
    - NewConstantString: Crea un nodo Constant de tipo ConstantValue de tipo string
    - NewConstantChar: Crea un nodo Constant de tipo ConstantValue de tipo char
- ListElem
    - NewListElem: Crea un nodo Constant de tipo ListElem
- Type
    - NewTypeArray : Crea un Type del tipo correspondiente con tamaño, indicando que es una lista
- Attribute
    - NewAttribute: Crea un nodo Constant de tipo Attribute

Tabla de consideraciones semánticas

Las consideraciones semánticas hacen uso de un hashmap cuya llave es un string que contiene la operación a realizar y los tipos de sus operandos en el formato “[Operacion]@[Tipo1][Tipo2]”. Los tipos están representados por números enteros o caracteres que representan un tipo diferente de la siguiente forma:

```
func ConvertInverse(t string) BasicType {
```

```
	if t == "1" {
		return Float
	}

	if t == "2" {
		return Char
	}

	if t == "3" {
		return Bool
	}

	if t == "4" {
		return Int
	}

	if t == "5" {
		return String
	}

	if t == "6" {
		return Void
	}

	return Null
}
```

Y para los objetos:

```
func (t ObjType) convert() rune {
	if t == Square {
		return '7'
	}

	if t == Circle {
		return '8'
	}

	if t == Image {
```

```
            return '9'
      }

      if t == Text {
            return 'a'
      }

      if t == Background {
            return 'b'
      }

      return 'n'
}
```

De tal forma que la tabla de consideraciones semánticas queda definida de la siguiente manera:

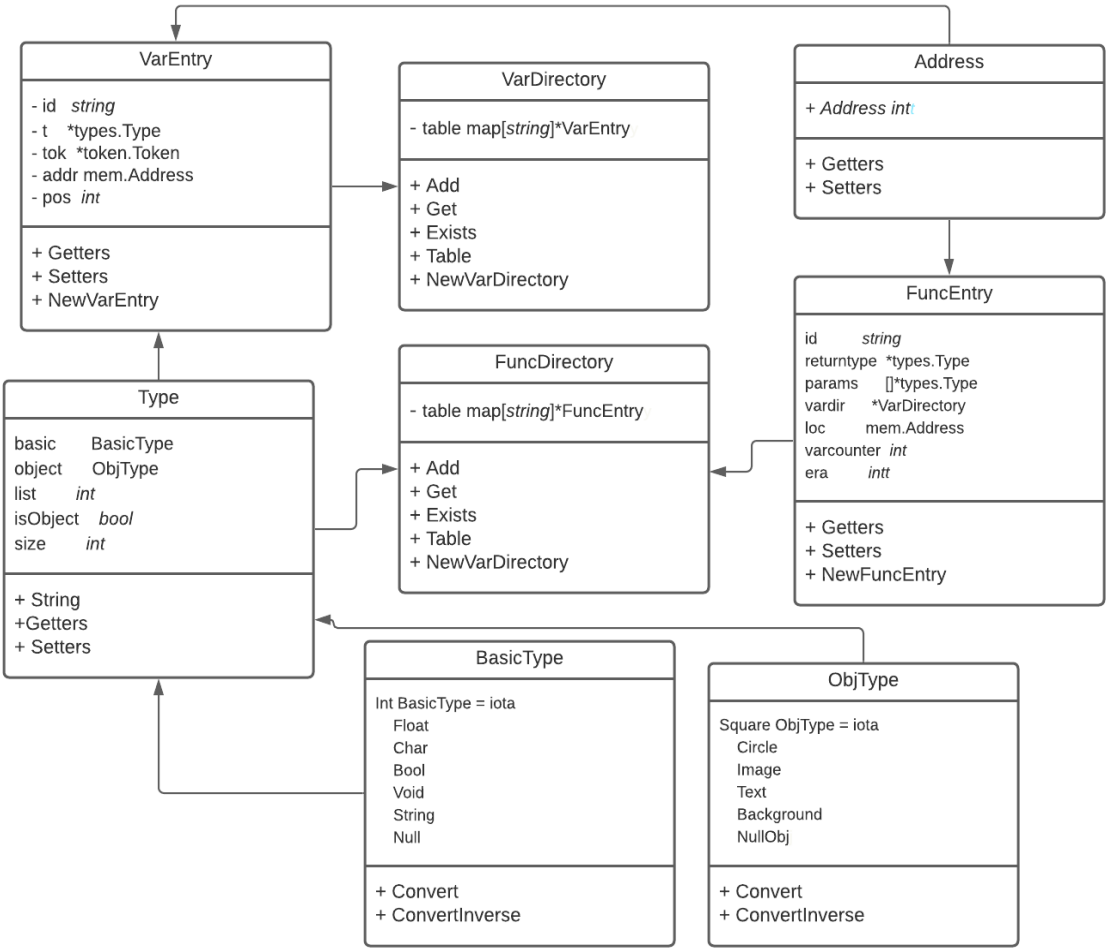
```
            //Arithmetical Operators
            "+@44": types.Int,
            "-@44": types.Int,
            "/@44": types.Int,
            "*@44": types.Int,
            "+@11": types.Float,
            "-@11": types.Float,
            "/@11": types.Float,
            "*@11": types.Float,

            //Relational Operators
            "<@44": types.Bool,
            ">@44": types.Bool,
            "<@11": types.Bool,
            ">@11": types.Bool,
            "==@44": types.Bool,
            "==@11": types.Bool,
            "==@33": types.Bool,
            "==@22": types.Bool,
            "==@55": types.Bool,
            "!=@11": types.Bool,
            "!=@55": types.Bool,
```

```
"!=@33": types.Bool,
"!=@22": types.Bool,
"!=@44": types.Bool,
//Logical Operators
"&&@33": types.Bool,
"||@33":  types.Bool,
"!@3":    types.Bool
```

Cabe mencionar que para el manejo de objetos y funciones reservadas, se definieron funciones adicionales que se encargan de asegurarse de que se realice el chequeo semántico de sus parámetros o atributos y operaciones entre éstos.

Proceso de Administración de Memoria



Aquí se pueden visualizar las diferentes estructuras que fueron usadas para generar el directorio de funciones y de variables. El directorio de funciones corresponde a la estructura FuncDirectory mientras que el directorio de variables corresponde a VarDirectory. Ambas contienen un hashmap de las entradas del directorio ya sea FuncEntry o VarsEntry. También tenemos la estructura de Type donde se puede representar un tipo de dato de nuestro lenguaje, ya sea basictype o un tipo de un objeto, por lo que contamos con BasicType y ObjType. Type es utilizado para crear variables, funciones, listas, constantes u objetos. FuncEntry contiene un arreglo de parámetros que recibe la función y un VarDirectory de todas las variables locales declaradas dentro de la función.

Existen 4 segmentos de memoria definidos por nosotros, siendo estos la memoria global, local, temporal y constante. El mapa de memoria se ve de la siguiente manera:

```
Floats -> Chars -> Booleans -> Ints -> Strings -> Squares -> Circles
-> Images -> Texts -> Backgrounds
========Global = 0
0-999     Float
1000-1999 Char
2000-2999 Bool
3000-3999 Int
4000-4999 String
5000-5999 Square
6000-6999 Circle
7000-7999 Image
8000-8999 Text
9000-9999 Background
========Local = 10000
10000-10999 Float
11000-11999 Char
12000-12999 Bool
13000-13999 Int
14000-14999 String
15000-15999 Square
16000-16999 Circle
17000-17999 Image
18000-18999 Text
19000-19999 Background
======= Temp = 20000
```

```
20000-20999 Float
21000-21999 Char
22000-22999 Bool
23000-23999 Int
24000-24999 String
25000-25999 Square
26000-26999 Circle
27000-27999 Image
28000-28999 Text
29000-29999 Background
========Constant = 30000
30000-30999 Float
31000-31999 Char
32000-32999 Bool
33000-33999 Int
34000-34999 String
35000-35999 Square
36000-36999 Circle
37000-37999 Image
38000-38999 Text
39000-39999 Background
```

La asignación de memoria de una variable depende entonces del segmento en el que se ubica y su tipo de dato. Por ejemplo, un temporal flotante tomaría una dirección entre 20000 y 20999 dependiendo de cuántos temporales haya en memoria. El primero tomará la posición 20000, el segundo el 20001 y así sucesivamente.

Cabe añadir que tenemos también un stack de tipos con el que manejamos tareas como la validación de tipos y se añaden y sacan del stack con los cuádruplos que lo requieran.

Máquina Virtual

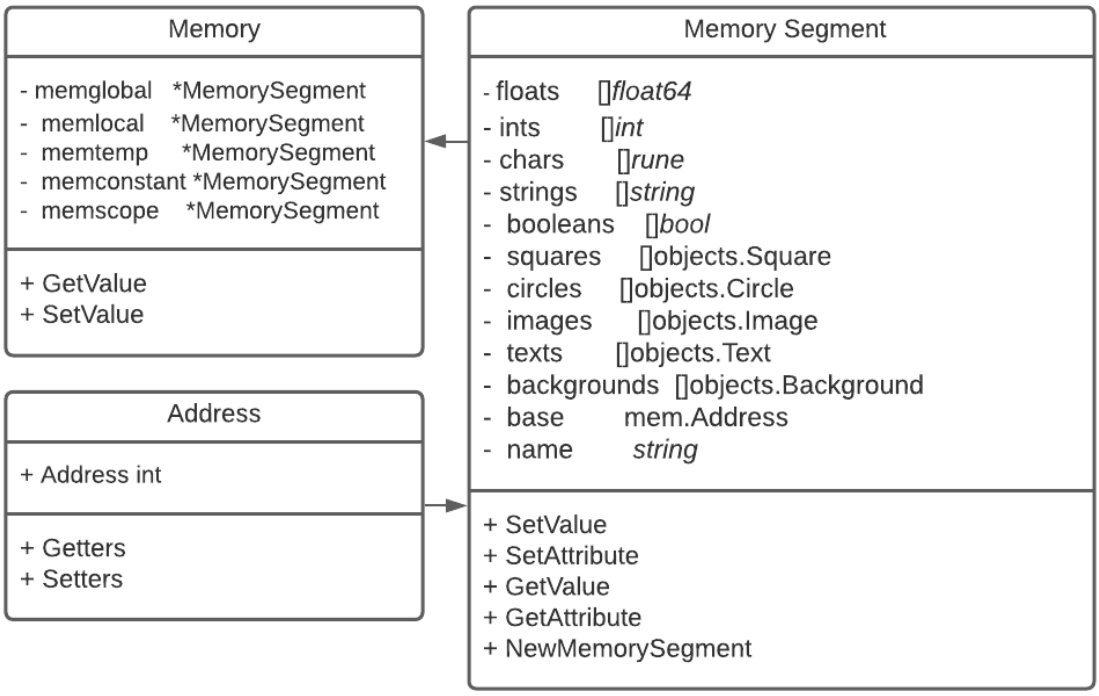
Equipo de cómputo, lenguaje y utilerías especiales usadas

Se utilizó el mismo que para el compilador.

Descripción detallada del proceso de Administración de Memoria en ejecución

La Maquina Virtual recibe los cuádruplos generados y se encarga de ejecutarlos dependiendo de la operación que defina cada cuádruplo. A continuación se explica cómo funciona la administración de memoria virtual en ejecución.

Especificación gráfica de CADA estructura de datos usada para manejo de scopes



Para administrar la memoria en la máquina virtual se crearon las estructuras que se muestran en el diagrama. La estructura de Memory contiene 5 estructuras de tipo Memory Segment que se utiliza para fragmentar la memoria dependiendo del tipo de dato que se desea guardar. Memory Segment contiene 9 arreglos que guardan diferentes tipos de datos. Para más detalle de la fragmentación de memoria, ver el mapa de memoria de la sección anterior. Esto significa que si se desea guardar un número tipo entero éste se guardaría en el arreglo de ints de tipo ints, los flotantes en el arreglo de tipo float, los booleanos en un arreglo de tipo bool, etc.

En cuanto a los objetos, se tienen arreglos de cada tipo de objeto válido en nuestro lenguaje al igual que con los tipos de dato básicos. Sin embargo, algo importante de

mencionar es que cada objeto contiene atributos que pueden ser modificados y tienen un valor. Para ello dejamos un offset del números de atributos que existen para los objetos. De este modo, un objeto Square local se guardaría por ejemplo en la dirección 15000 mientras que el siguiente Square declarado localmente tomaría la dirección 15009 para dejar espacio para los 8 atributos del primer objeto.

Esta misma convención se aplicó para guardar listas en memoria, puesto que el tamaño de una lista se define desde que se declara el arreglo, es posible reservar las *n* direcciones de memoria donde *n* es el tamaño del arreglo.

Asociación hecha entre las direcciones virtuales (compilación) y las reales (ejecución)

Para identificar una dirección en particular en memoria, se utiliza la convención que fue descrita previamente en el mapa de memoria, donde usamos los rangos de direcciones para saber si estamos hablando de un segmento local, global, etc. Con esto sabemos en cuál de los MemorySegments de la estructura Memory se encuentra la dirección que nos dan. Después podemos determinar el tipo de dato que representa realizando la resta entre la dirección dada y la dirección base del memory segment, y comparando este resultado con los rangos de direcciones por tipos que se definieron. Con esto logramos determinar en qué arreglo del memory segment debemos acceder, y en qué posición dentro de este arreglo.

Por ejemplo si queremos ubicar la dirección 13024, vemos que está ubicada entre 10000 y 20000 lo que significa que está dentro del memory segment local. Después realizamos la resta de nuestra posición 13024 con la posición base del memory segment local de 10000 y nos da como resultado 3024. Este número nos dice que es un int pues está entre el rango de 3000 y 4000, y restarle este offset de 3000 nos da la dirección 24. Con esto sabemos que se encuentra en la posición 24 del arreglo de ints del memory segment. Además sabemos ahora que existen 24 enteros guardados en memoria local porque generamos la memoria de forma eficiente conforme se requieren más direcciones, y esto significa que los espacios entre 0 y 23 también han sido ocupados. La función *GetValue (mem.Address)* incluída en el diagrama se encarga de llevar a cabo todo este proceso.

Pruebas del Funcionamiento del Lenguaje

```
program factorial;
```

```
{

}

int fac(int n){
    if(n == 1){
        return 1;
    }
    else{
        return n * fac(n - 1);
    }
}

void main(){
    print(fac(5));

}
```

0: Operation: Goto -1 -1 12
12: Operation: Era -1 -1 -1
13: Operation: Param 33002 -1 -1
14: Operation: Call 1 -1 23000
1: Operation: Equal 13000 33001 22000
2: Operation: GotoF 22000 -1 5
5: Operation: Era -1 -1 -1
6: Operation: - 13000 33001 23000
7: Operation: Param 23000 -1 -1
8: Operation: Call 1 -1 23001
1: Operation: Equal 13000 33001 22000
2: Operation: GotoF 22000 -1 5
5: Operation: Era -1 -1 -1
6: Operation: - 13000 33001 23000
7: Operation: Param 23000 -1 -1
8: Operation: Call 1 -1 23001
1: Operation: Equal 13000 33001 22000
2: Operation: GotoF 22000 -1 5
5: Operation: Era -1 -1 -1
6: Operation: - 13000 33001 23000
7: Operation: Param 23000 -1 -1
8: Operation: Call 1 -1 23001
1: Operation: Equal 13000 33001 22000
2: Operation: GotoF 22000 -1 5

```
5: Operation: Era -1 -1 -1
6: Operation: - 13000 33001 23000
7: Operation: Param 23000 -1 -1
8: Operation: Call 1 -1 23001
1: Operation: Equal 13000 33001 22000
2: Operation: GotoF 22000 -1 5
3: Operation: Ret -1 -1 33001
9: Operation: * 13000 23001 23002
10: Operation: Ret -1 -1 23002
9: Operation: * 13000 23001 23002
10: Operation: Ret -1 -1 23002
9: Operation: * 13000 23001 23002
10: Operation: Ret -1 -1 23002
9: Operation: * 13000 23001 23002
10: Operation: Ret -1 -1 23002
15: Operation: Print -1 -1 23000
120
16: Operation: Ret -1 -1 -1
```

```
program FiboIterative;

{
   int i;
}

int fibIt(int n) {
   int a, b, tmp;
   a = 0;
   b = 1;

   if (n < 2) {
      return n;
   }

   for (i = 2; i < n; i = i+1) {
      tmp = a + b;
      a = b;
      b = tmp;
   }

   return b;
}
```

```
void main() {
  print(fibIt(5));
}
```

```
0: Operation: Init 3000 1 4
1: Operation: Goto -1 -1 22
22: Operation: Era -1 -1 -1
23: Operation: Param 33003 -1 -1
24: Operation: Call 2 -1 23000
2: Operation: Init 13001 1 4
3: Operation: Init 13002 1 4
4: Operation: Init 13003 1 4
5: Operation: Assign 33000 -1 13003
6: Operation: Assign 33001 -1 13002
7: Operation: < 13000 33002 22000
8: Operation: GotoF 22000 -1 10
10: Operation: Assign 33002 -1 3000
11: Operation: < 3000 13000 22001
12: Operation: GotoF 22001 -1 20
13: Operation: + 13003 13002 23000
14: Operation: Assign 23000 -1 13001
15: Operation: Assign 13002 -1 13003
16: Operation: Assign 13001 -1 13002
17: Operation: + 3000 33001 23001
18: Operation: Assign 23001 -1 3000
19: Operation: Goto -1 -1 11
11: Operation: < 3000 13000 22001
12: Operation: GotoF 22001 -1 20
13: Operation: + 13003 13002 23000
14: Operation: Assign 23000 -1 13001
15: Operation: Assign 13002 -1 13003
16: Operation: Assign 13001 -1 13002
17: Operation: + 3000 33001 23001
18: Operation: Assign 23001 -1 3000
19: Operation: Goto -1 -1 11
11: Operation: < 3000 13000 22001
12: Operation: GotoF 22001 -1 20
13: Operation: + 13003 13002 23000
14: Operation: Assign 23000 -1 13001
15: Operation: Assign 13002 -1 13003
16: Operation: Assign 13001 -1 13002
17: Operation: + 3000 33001 23001
18: Operation: Assign 23001 -1 3000
19: Operation: Goto -1 -1 11
```

```
11: Operation: < 3000 13000 22001
12: Operation: GotoF 22001 -1 20
20: Operation: Ret -1 -1 13002
25: Operation: Print -1 -1 23000
3
26: Operation: Ret -1 -1 -1
```

```
program FiboRecursive;

{
    int i;
}

int fib(int n) {
    if (n == 2) {
        return 1;
    }

    if (n == 1) {
        return 0;
    }

    int y;
    y = fib(n - 2);
    return fib(n - 1) + y;
}

void main() {
  print(fib(8));
}
```

```
0: Operation: Init 3000 1 4
1: Operation: Goto -1 -1 21
21: Operation: Era -1 -1 -1
22: Operation: Param 33003 -1 -1
23: Operation: Call 2 -1 23000
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
8: Operation: Init 13001 1 4
9: Operation: Era -1 -1 -1
10: Operation: - 13000 33001 23000
11: Operation: Param 23000 -1 -1
```

12: Operation: Call 2 -1 23001
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
8: Operation: Init 13001 1 4
9: Operation: Era -1 -1 -1
10: Operation: - 13000 33001 23000
11: Operation: Param 23000 -1 -1
12: Operation: Call 2 -1 23001
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
8: Operation: Init 13001 1 4
9: Operation: Era -1 -1 -1
10: Operation: - 13000 33001 23000
11: Operation: Param 23000 -1 -1
12: Operation: Call 2 -1 23001
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
4: Operation: Ret -1 -1 33002
13: Operation: Assign 23001 -1 13001
14: Operation: Era -1 -1 -1
15: Operation: - 13000 33002 23002
16: Operation: Param 23002 -1 -1
17: Operation: Call 2 -1 23003
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
8: Operation: Init 13001 1 4
9: Operation: Era -1 -1 -1
10: Operation: - 13000 33001 23000
11: Operation: Param 23000 -1 -1
12: Operation: Call 2 -1 23001
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
7: Operation: Ret -1 -1 33000
13: Operation: Assign 23001 -1 13001
14: Operation: Era -1 -1 -1
15: Operation: - 13000 33002 23002

16: Operation: Param 23002 -1 -1
17: Operation: Call 2 -1 23003
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
4: Operation: Ret -1 -1 33002
18: Operation: + 23003 13001 23004
19: Operation: Ret -1 -1 23004
18: Operation: + 23003 13001 23004
19: Operation: Ret -1 -1 23004
13: Operation: Assign 23001 -1 13001
14: Operation: Era -1 -1 -1
15: Operation: - 13000 33002 23002
16: Operation: Param 23002 -1 -1
17: Operation: Call 2 -1 23003
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
8: Operation: Init 13001 1 4
9: Operation: Era -1 -1 -1
10: Operation: - 13000 33001 23000
11: Operation: Param 23000 -1 -1
12: Operation: Call 2 -1 23001
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
8: Operation: Init 13001 1 4
9: Operation: Era -1 -1 -1
10: Operation: - 13000 33001 23000
11: Operation: Param 23000 -1 -1
12: Operation: Call 2 -1 23001
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
7: Operation: Ret -1 -1 33000
13: Operation: Assign 23001 -1 13001
14: Operation: Era -1 -1 -1
15: Operation: - 13000 33002 23002
16: Operation: Param 23002 -1 -1
17: Operation: Call 2 -1 23003
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5

4: Operation: Ret -1 -1 33002
18: Operation: + 23003 13001 23004
19: Operation: Ret -1 -1 23004
13: Operation: Assign 23001 -1 13001
14: Operation: Era -1 -1 -1
15: Operation: - 13000 33002 23002
16: Operation: Param 23002 -1 -1
17: Operation: Call 2 -1 23003
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
8: Operation: Init 13001 1 4
9: Operation: Era -1 -1 -1
10: Operation: - 13000 33001 23000
11: Operation: Param 23000 -1 -1
12: Operation: Call 2 -1 23001
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
4: Operation: Ret -1 -1 33002
13: Operation: Assign 23001 -1 13001
14: Operation: Era -1 -1 -1
15: Operation: - 13000 33002 23002
16: Operation: Param 23002 -1 -1
17: Operation: Call 2 -1 23003
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
8: Operation: Init 13001 1 4
9: Operation: Era -1 -1 -1
10: Operation: - 13000 33001 23000
11: Operation: Param 23000 -1 -1
12: Operation: Call 2 -1 23001
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
7: Operation: Ret -1 -1 33000
13: Operation: Assign 23001 -1 13001
14: Operation: Era -1 -1 -1
15: Operation: - 13000 33002 23002
16: Operation: Param 23002 -1 -1
17: Operation: Call 2 -1 23003

2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
4: Operation: Ret -1 -1 33002
18: Operation: + 23003 13001 23004
19: Operation: Ret -1 -1 23004
18: Operation: + 23003 13001 23004
19: Operation: Ret -1 -1 23004
18: Operation: + 23003 13001 23004
19: Operation: Ret -1 -1 23004
18: Operation: + 23003 13001 23004
19: Operation: Ret -1 -1 23004
13: Operation: Assign 23001 -1 13001
14: Operation: Era -1 -1 -1
15: Operation: - 13000 33002 23002
16: Operation: Param 23002 -1 -1
17: Operation: Call 2 -1 23003
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
8: Operation: Init 13001 1 4
9: Operation: Era -1 -1 -1
10: Operation: - 13000 33001 23000
11: Operation: Param 23000 -1 -1
12: Operation: Call 2 -1 23001
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
8: Operation: Init 13001 1 4
9: Operation: Era -1 -1 -1
10: Operation: - 13000 33001 23000
11: Operation: Param 23000 -1 -1
12: Operation: Call 2 -1 23001
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
8: Operation: Init 13001 1 4
9: Operation: Era -1 -1 -1
10: Operation: - 13000 33001 23000
11: Operation: Param 23000 -1 -1
12: Operation: Call 2 -1 23001
2: Operation: Equal 13000 33001 22000

3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
7: Operation: Ret -1 -1 33000
13: Operation: Assign 23001 -1 13001
14: Operation: Era -1 -1 -1
15: Operation: - 13000 33002 23002
16: Operation: Param 23002 -1 -1
17: Operation: Call 2 -1 23003
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
4: Operation: Ret -1 -1 33002
18: Operation: + 23003 13001 23004
19: Operation: Ret -1 -1 23004
13: Operation: Assign 23001 -1 13001
14: Operation: Era -1 -1 -1
15: Operation: - 13000 33002 23002
16: Operation: Param 23002 -1 -1
17: Operation: Call 2 -1 23003
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
8: Operation: Init 13001 1 4
9: Operation: Era -1 -1 -1
10: Operation: - 13000 33001 23000
11: Operation: Param 23000 -1 -1
12: Operation: Call 2 -1 23001
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
4: Operation: Ret -1 -1 33002
13: Operation: Assign 23001 -1 13001
14: Operation: Era -1 -1 -1
15: Operation: - 13000 33002 23002
16: Operation: Param 23002 -1 -1
17: Operation: Call 2 -1 23003
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
8: Operation: Init 13001 1 4
9: Operation: Era -1 -1 -1
10: Operation: - 13000 33001 23000
11: Operation: Param 23000 -1 -1

12: Operation: Call 2 -1 23001
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
7: Operation: Ret -1 -1 33000
13: Operation: Assign 23001 -1 13001
14: Operation: Era -1 -1 -1
15: Operation: - 13000 33002 23002
16: Operation: Param 23002 -1 -1
17: Operation: Call 2 -1 23003
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
4: Operation: Ret -1 -1 33002
18: Operation: + 23003 13001 23004
19: Operation: Ret -1 -1 23004
18: Operation: + 23003 13001 23004
19: Operation: Ret -1 -1 23004
18: Operation: + 23003 13001 23004
19: Operation: Ret -1 -1 23004
13: Operation: Assign 23001 -1 13001
14: Operation: Era -1 -1 -1
15: Operation: - 13000 33002 23002
16: Operation: Param 23002 -1 -1
17: Operation: Call 2 -1 23003
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
8: Operation: Init 13001 1 4
9: Operation: Era -1 -1 -1
10: Operation: - 13000 33001 23000
11: Operation: Param 23000 -1 -1
12: Operation: Call 2 -1 23001
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
8: Operation: Init 13001 1 4
9: Operation: Era -1 -1 -1
10: Operation: - 13000 33001 23000
11: Operation: Param 23000 -1 -1
12: Operation: Call 2 -1 23001
2: Operation: Equal 13000 33001 22000

3: Operation: GotoF 22000 -1 5
4: Operation: Ret -1 -1 33002
13: Operation: Assign 23001 -1 13001
14: Operation: Era -1 -1 -1
15: Operation: - 13000 33002 23002
16: Operation: Param 23002 -1 -1
17: Operation: Call 2 -1 23003
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
8: Operation: Init 13001 1 4
9: Operation: Era -1 -1 -1
10: Operation: - 13000 33001 23000
11: Operation: Param 23000 -1 -1
12: Operation: Call 2 -1 23001
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
7: Operation: Ret -1 -1 33000
13: Operation: Assign 23001 -1 13001
14: Operation: Era -1 -1 -1
15: Operation: - 13000 33002 23002
16: Operation: Param 23002 -1 -1
17: Operation: Call 2 -1 23003
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
4: Operation: Ret -1 -1 33002
18: Operation: + 23003 13001 23004
19: Operation: Ret -1 -1 23004
18: Operation: + 23003 13001 23004
19: Operation: Ret -1 -1 23004
13: Operation: Assign 23001 -1 13001
14: Operation: Era -1 -1 -1
15: Operation: - 13000 33002 23002
16: Operation: Param 23002 -1 -1
17: Operation: Call 2 -1 23003
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
8: Operation: Init 13001 1 4
9: Operation: Era -1 -1 -1

10: Operation: - 13000 33001 23000
11: Operation: Param 23000 -1 -1
12: Operation: Call 2 -1 23001
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
8: Operation: Init 13001 1 4
9: Operation: Era -1 -1 -1
10: Operation: - 13000 33001 23000
11: Operation: Param 23000 -1 -1
12: Operation: Call 2 -1 23001
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
7: Operation: Ret -1 -1 33000
13: Operation: Assign 23001 -1 13001
14: Operation: Era -1 -1 -1
15: Operation: - 13000 33002 23002
16: Operation: Param 23002 -1 -1
17: Operation: Call 2 -1 23003
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
4: Operation: Ret -1 -1 33002
18: Operation: + 23003 13001 23004
19: Operation: Ret -1 -1 23004
13: Operation: Assign 23001 -1 13001
14: Operation: Era -1 -1 -1
15: Operation: - 13000 33002 23002
16: Operation: Param 23002 -1 -1
17: Operation: Call 2 -1 23003
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
8: Operation: Init 13001 1 4
9: Operation: Era -1 -1 -1
10: Operation: - 13000 33001 23000
11: Operation: Param 23000 -1 -1
12: Operation: Call 2 -1 23001
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
4: Operation: Ret -1 -1 33002

13: Operation: Assign 23001 -1 13001
14: Operation: Era -1 -1 -1
15: Operation: - 13000 33002 23002
16: Operation: Param 23002 -1 -1
17: Operation: Call 2 -1 23003
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
8: Operation: Init 13001 1 4
9: Operation: Era -1 -1 -1
10: Operation: - 13000 33001 23000
11: Operation: Param 23000 -1 -1
12: Operation: Call 2 -1 23001
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
5: Operation: Equal 13000 33002 22001
6: Operation: GotoF 22001 -1 8
7: Operation: Ret -1 -1 33000
13: Operation: Assign 23001 -1 13001
14: Operation: Era -1 -1 -1
15: Operation: - 13000 33002 23002
16: Operation: Param 23002 -1 -1
17: Operation: Call 2 -1 23003
2: Operation: Equal 13000 33001 22000
3: Operation: GotoF 22000 -1 5
4: Operation: Ret -1 -1 33002
18: Operation: + 23003 13001 23004
19: Operation: Ret -1 -1 23004
18: Operation: + 23003 13001 23004
19: Operation: Ret -1 -1 23004
18: Operation: + 23003 13001 23004
19: Operation: Ret -1 -1 23004
18: Operation: + 23003 13001 23004
19: Operation: Ret -1 -1 23004
18: Operation: + 23003 13001 23004
19: Operation: Ret -1 -1 23004
18: Operation: + 23003 13001 23004
19: Operation: Ret -1 -1 23004
24: Operation: Print -1 -1 23000
13
25: Operation: Ret -1 -1 -1

```
program PingPong;

{
    int playerOnePoints, playerTwoPoints;
    Circle ball;
    Square playerOne, playerTwo;
    float ballXVel, ballYVel;
    bool gameOver, starting;

    Text scoreOne, scoreTwo, oneWins, twoWins;
}

void movePlayerOne(){
    if(KeyPressed("W")){
       if(playerOne.y < 390.0){
            playerOne.y = playerOne.y + 10.0;
        }
    }
    if(KeyPressed("S")){
        if(playerOne.y > 10.0){
            playerOne.y = playerOne.y - 10.0;
        }
    }
}

void movePlayerTwo(){
    if(KeyPressed("Up")){
        if(playerTwo.y < 390.0){
            playerTwo.y = playerTwo.y + 10.0;
        }
    }
    if(KeyPressed("Down")){
        if(playerTwo.y >  10.0){
            playerTwo.y = playerTwo.y - 10.0;
        }
    }
}

void checkCollisionOne(){
    if (CheckCollision(ball, playerOne) ) {

    bool playerBetween;
    playerBetween = (ball.x < playerOne.x)  && (ball.x < playerOne.x + playerOne.width) ;
    //verify if the ball's upper edge is bewteen the player's height
```

```
    bool upBetweenPlayer;
    upBetweenPlayer = (ball.x + ball.width < playerOne.x)  && (ball.x + ball.width > playerOne.x +
playerOne.width);
    //verify if the ball's lower edge is between the player's height
    bool downBetweenPlayer;
    downBetweenPlayer = (ball.x > playerOne.x) && (ball.x + ball.width < playerOne.x +
playerOne.width);

        if(playerBetween || upBetweenPlayer || downBetweenPlayer){
            if(ball.y>playerOne.y+playerOne.width/2.0){
                //the ball hits the player from above
                //make y vel negative
                if(ballXVel > 0.0){
                    ballYVel = ballYVel * (-1.0);
                }
            }
            else{
                //the ball hits the player from the bottom
                //make y vel positive
                if(ballYVel < 0.0){
                    ballYVel = ballYVel * (-1.0);
                }
            }
        }

        else{
            //ball hits from up
            //Calculate the distance the ball is from the center of the player pad
            float dist;
            dist = ball.y + ball.height / 2.0 - (playerOne.y + playerOne.height / 2.0);

            //Map the dist value in proportion to the maxVel of the ball
            float deltaVel;
            deltaVel =  (dist / 50.0) * 10.0;

            //Update ball's yVel accordingly
            ballYVel = ballYVel + deltaVel;

            //Bounce the velocity in the x component
            ballXVel = ballXVel * (-1.0);
        }
    }
}
```

```
void checkCollisionTwo(){
   if (CheckCollision(ball, playerTwo) ) {
      bool playerBetween;
      playerBetween = (ball.x > playerTwo.x)  && (ball.x > playerTwo.x + playerTwo.width );

      //verify if the ball's upper edge is between the player's height
      bool upBetweenPlayer;
      upBetweenPlayer = (ball.x + ball.width > playerTwo.x ) && (ball.x + ball.width <
playerTwo.x + playerTwo.width);

      //verify if the ball's lower edge is between the player's height
      bool downBetweenPlayer;
      downBetweenPlayer = (ball.x < playerTwo.x) && (ball.x + ball.width > playerTwo.x +
playerTwo.width);

      if(playerBetween || upBetweenPlayer || downBetweenPlayer){
         if(ball.y < playerTwo.y+playerTwo.width/2.0){
            //the ball hits the player from above
            //make y vel negative
            if(ballXVel > 0.1){
               ballYVel = ballYVel * (-1.0);
            }
         }
         else{ //the ball hits the player from the bottom
            //make y vel positive
            if(ballYVel < 0.1){
               ballYVel = ballYVel * (-1.0);
            }
         }
      }

      else{  //ball hits from up
         //Calculate the distance the ball is from the center of the player pad
         float dist;
         dist = ball.y + ball.height / 2.0 - (playerTwo.y + playerTwo.height / 2.0);

         //Map the dist value in proportion to the maxVel of the ball
         float deltaVel;
         deltaVel =  (dist / 50.0) * 10.0;

         //Update ball's yVel accordingly
         ballYVel = ballYVel + deltaVel;

         //Bounce the velocity in the x component
```

```
            ballXVel = ballXVel * (-1.0);
        }
    }
}

void moveBall(){
    ball.x = ball.x + ballXVel;
    ball.y = ball.y + ballYVel;
}

void render() {
    Render(playerOne);
    Render(playerTwo);
    Render(ball);
    Render(scoreOne);
    Render(scoreTwo);
    if(playerOnePoints==3){
            Render(oneWins);
    }
    if(playerTwoPoints==3){
        Render(twoWins);
    }

}

void updateScore(){
    if(playerOnePoints == 1){
        scoreOne.message = "1";
    }
    if(playerOnePoints == 2){
        scoreOne.message = "2";
    }
    if(playerOnePoints == 3){
        scoreOne.message = "3";
    }
    if(playerTwoPoints == 1){
        scoreTwo.message = "1";
    }
    if(playerTwoPoints == 2){
        scoreTwo.message = "2";
    }
    if(playerTwoPoints == 3){
        scoreTwo.message = "3";
    }
```

```
    Render(scoreOne);
    Render(scoreTwo);
}

void tick(){
    Clear();

    movePlayerOne();
    movePlayerTwo();
    if(gameOver==false){
        if(starting){
            if(KeyPressed("Space")){
                starting = false;
                ballXVel = 10.0;
                ballYVel = 10.0;
            }
        }
        else{
            checkCollisionOne();
            checkCollisionTwo();

            //bounce from top and bottom boundaries
            if(( (ball.y < 0.1) || (ball.y + ball.height > 500.0))  && ((ball.x > 0.1) && (ball.x + ball.width
< 750.0))){
                ballYVel = ballYVel * (-1.0);
            }

            //player one scores a point
            if(ball.x + ball.width > 750.0){
                playerOnePoints = playerOnePoints + 1;
                starting = true;
                playerOne.x = 0.0;
                playerOne.y = 255.0;
                playerTwo.x = 710.0;
                playerTwo.y = 255.0;
                ball.x = 680.0;
                ball.y = 235.0;
                ballXVel = -10.0;
                ballYVel = -10.0;
                updateScore();
            }

            //player two scores a point
            if(ball.x < 1.0){
```

```
            playerTwoPoints = playerTwoPoints + 1;
            starting = true;
            playerOne.x = 0.0;
            playerOne.y = 255.0;
            playerTwo.x = 710.0;
            playerTwo.y = 255.0;
            ball.x = 20.0;
            ball.y = 235.0;
            ballXVel = 10.0;
            ballYVel = 10.0;
            updateScore();
        }

        moveBall();

        if(playerOnePoints == 3){

            oneWins.message = "PLAYER 1 WINS";
            oneWins.x = 275.0;
            oneWins.y = 200.0;
            oneWins.size = 40.0;
            Render(oneWins);
            gameOver = true;
        }

        if(playerTwoPoints == 3){
            twoWins.message = "PLAYER 2 WINS";
            twoWins.x = 275.0;
            twoWins.y = 200.0;
            twoWins.size = 40.0;
            Render(twoWins);
            gameOver = true;
        }
      }
   }
   render();
   Update();

}

void main(){


   playerOnePoints = 0;
```

```
    playerTwoPoints = 0;

    gameOver = false;
    starting = true;

    ballXVel = 10.0;
    ballYVel = 10.0;



    ball.width  = 30.0;
    ball.height = 30.0;
    ball.color = "ffffff";
    ball.x = 21.0;
    ball.y = 235.0;

    playerOne.color = "ffffff";
    playerOne.width = 20.0;
    playerOne.height = 100.0;
    playerOne.x = 0.0;
    playerOne.y = 225.0;

    playerTwo.color = "ffffff";
    playerTwo.width = 20.0;
    playerTwo.height = 100.0;
    playerTwo.x = 710.0;
    playerTwo.y = 225.0;

    scoreOne.message = "0";
    scoreOne.size = 30.0;
    scoreOne.x = 300.0;
    scoreOne.y = 10.0;
    Render(scoreOne);
    scoreTwo.message = "0";
    scoreTwo.size = 30.0;
    scoreTwo.x = 330.0;
    scoreTwo.y = 10.0;
    Render(scoreTwo);

    while(true){
        tick();
    }
}
```

(Corriendo por 1 a 2 segundos)

0: Operation: Init 1 1 1
1: Operation: Init 5000 1 7
2: Operation: Init 5009 1 7
3: Operation: Init 8009 1 10
4: Operation: Init 8018 1 10
5: Operation: Init 8027 1 10
6: Operation: Init 2000 1 3
7: Operation: Init 2001 1 3
8: Operation: Init 3000 1 4
9: Operation: Init 3001 1 4
10: Operation: Init 8000 1 10
11: Operation: Init 0 1 1
12: Operation: Init 6000 1 8
13: Operation: Goto -1 -1 273
273: Operation: Assign 33000 -1 3001
274: Operation: Assign 33000 -1 3000
275: Operation: Assign 32000 -1 2001
276: Operation: Assign 32001 -1 2000
277: Operation: Assign 30002 -1 1
278: Operation: Assign 30002 -1 0
279: Operation: Assign 30019 -1 6002
280: Operation: Assign 30019 -1 6001
281: Operation: Assign 34011 -1 6006
282: Operation: Assign 30020 -1 6003
283: Operation: Assign 30012 -1 6004
284: Operation: Assign 34011 -1 5015
285: Operation: Assign 30015 -1 5011
286: Operation: Assign 30021 -1 5010
287: Operation: Assign 30000 -1 5012
288: Operation: Assign 30022 -1 5013
289: Operation: Assign 34011 -1 5006
290: Operation: Assign 30015 -1 5002
291: Operation: Assign 30021 -1 5001
292: Operation: Assign 30010 -1 5003
293: Operation: Assign 30022 -1 5004
294: Operation: Assign 34012 -1 8034
295: Operation: Assign 30019 -1 8032
296: Operation: Assign 30023 -1 8030
297: Operation: Assign 30002 -1 8031
298: Operation: Render 8027 -1 -1
299: Operation: Assign 34012 -1 8007
300: Operation: Assign 30019 -1 8005

301: Operation: Assign 30024 -1 8003
302: Operation: Assign 30002 -1 8004
303: Operation: Render 8000 -1 -1
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1

306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000

15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1

194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39

39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001

199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1

158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000

163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1

272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1

190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20

20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1

27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1

195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203

203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1

159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165

165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1

307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1

191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002

21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000

28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000

196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269

269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1

160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001

166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304

304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1

192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26

26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33

33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269

197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1

270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1

161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168

168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308

305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1

14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1

193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
199: Operation: GotoF 22001 -1 203
203: Operation: Goto -1 -1 269
269: Operation: Era -1 -1 -1
270: Operation: Call 157 -1 -1
157: Operation: Render 5009 -1 -1
158: Operation: Render 5000 -1 -1
159: Operation: Render 6000 -1 -1
160: Operation: Render 8027 -1 -1
161: Operation: Render 8000 -1 -1
162: Operation: Equal 3001 33001 22000
163: Operation: GotoF 22000 -1 165
165: Operation: Equal 3000 33001 22001
166: Operation: GotoF 22001 -1 168
168: Operation: Ret -1 -1 -1
271: Operation: Update -1 -1 -1
272: Operation: Ret -1 -1 -1
307: Operation: Goto -1 -1 304
304: Operation: GotoF 32001 -1 308
305: Operation: Era -1 -1 -1
306: Operation: Call 190 -1 -1
190: Operation: Clear -1 -1 -1
191: Operation: Era -1 -1 -1
192: Operation: Call 14 -1 -1
14: Operation: KeyPressed 34001 -1 22000
15: Operation: GotoF 22000 -1 20
20: Operation: KeyPressed 34002 -1 22002
21: Operation: GotoF 22002 -1 26
26: Operation: Ret -1 -1 -1
193: Operation: Era -1 -1 -1
194: Operation: Call 27 -1 -1
27: Operation: KeyPressed 34003 -1 22000
28: Operation: GotoF 22000 -1 33
33: Operation: KeyPressed 34004 -1 22002

```
34: Operation: GotoF 22002 -1 39
39: Operation: Ret -1 -1 -1
195: Operation: Equal 2001 32000 22000
196: Operation: GotoF 22000 -1 269
197: Operation: GotoF 2000 -1 204
198: Operation: KeyPressed 34008 -1 22001
exit status 3221225786
```

Otros Links

Video Tutorial

Repositorio del Compilador

Manual de Usuario