Proyecto 1 organizacioni de lenguajes y compiladores 2

Jossie Bismarck Castrillo Fajardo 201313692

Manual técnico

Contenido

[Descripción del lenguaje 2](#_Toc43314166)

[Sintaxis del lenguaje 3](#_Toc43314167)

[Definición de variables 3](#_Toc43314168)

[Instrucciones simploes y unarias 3](#_Toc43314169)

[Conversiones 4](#_Toc43314170)

[Instrucciones aritméticas 4](#_Toc43314171)

[Instrucciones lógicas 4](#_Toc43314172)

[Instrucciones bit a bit 5](#_Toc43314173)

[Instrucciones relacionales 5](#_Toc43314174)

[Arreglos y Structs 5](#_Toc43314175)

[Instrucciones de control 5](#_Toc43314176)

[Flujo de la aplicación 6](#_Toc43314177)

[Diagrama de clases 7](#_Toc43314178)

[Librerías utilizadas 8](#_Toc43314179)

# Descripción del lenguaje

Augus es un lenguaje de programación que se basa en PHP y MIPS. Su principal funcionalidad es ser un lenguaje intermedio, no es de alto nivel como PHP ni de bajo nivel como el lenguaje ensamblador MIPS.

El lenguaje tiene dos restricciones: la primera, es que cada instrucción es una operación simple; y la segunda, es que en cada instrucción hay un máximo de dos operandos y su asignación (si la hubiera).

Es un lenguaje débilmente tipado, sin embargo, si se reconocen cuatro tipos de datos no explícitos: entero, punto flotante, cadena de caracteres y arreglo.

Para manejar el flujo de control se proporciona la declaración de etiquetas, sin tener palabras reservadas para ese uso. Es decir, no hay ciclos for, while, ni do-while.

# Sintaxis del lenguaje

## Definición de variables

|  |  |
| --- | --- |
| $t0..$tn | Temporales |
| $a0..$an | Parámetros |
| $v0..$vn | Valores devueltos por funciones |
| $ra | Simulador de dirección de retorno por nivel |
| $s0..$sn | Pilas |
| $sp | Puntero de la pila |

## Instrucciones simploes y unarias

|  |  |
| --- | --- |
| main: | Inicio del programa. (obligatorio) |
| label: | Definición del inicio de una etiqueta. |
| goto label; | Salto incondicional hacia una etiqueta. |
| $t1 = 10; | Asignación numérica. |
| $t1 = 'hola';  $t1 = "hola"; | Asignación de una cadena de caracteres. |
| $t1 = $t2; | Copia simple. |
| $t1 = - $t2; | Negativo. |
| $t1 = &$t2; | $t1 es un puntero a la dirección de $t2. |
| unset($t1); | Destruye la variable $t1. |
| print($t1); | Imprime en pantalla el contenido de $t1.  Solo se pueden imprimir registros y cadenas, no operaciones. |
| $t1 = read(); | Lee la entrada del teclado queda en $t1.  Debe reconocer si es un entero, punto flotante o cadena. |
| #comment | Comentario de una sola línea. |
| exit; | Finaliza la ejecución. (opcional) |

## Conversiones

|  |  |
| --- | --- |
| $t1 = (int) $t2; | Si $t2 tiene decimales son eliminados.  Si $t2 es un carácter se toma su código ASCII.  Si $t2 es una cadena se toma el ASCII del primer carácter.  Si $t2 es un arreglo se aplica al primer elemento las reglas anteriores. |
| $t1 = (float) $t2; | Si $t2 es entero se agrega “.0”.  Si $t2 es un carácter se toma su código ASCII con decimal.  Si $t2 es una cadena se toma el ASCII del primer carácter más el decimal.  Si $t2 es un arreglo se aplica al primer elemento las reglas anteriores. |
| $t1 = (char) $t2; | Si $t2 es un número de 0 a 255 se convierte en el carácter basado en ASCII.  Si $t2 es un número mayor a 255 entonces se aplica el módulo 256 para extraer el ASCII.  Si $t2 es un decimal, se quitan los decimales y se aplican las reglas anteriores.  Si $t2 es una cadena, se almacena solo el primer carácter.  Si $t2 es un arreglo, se almacena solo el primer valor aplicando las reglas anteriores. |

## Instrucciones aritméticas

|  |  |
| --- | --- |
| $t1 = $t2 + $t3; | Suma.  Solamente si $t2 y $t3 son cadenas entonces se concatena. |
| $t1 = $t2 - $t3; | Resta. |
| $t1 = $t2 \* $t3; | Multiplicación. |
| $t1 = $t2 / $t3; | División. |
| $t1 = $t2 % $t3 | Residuo. |
| $t1 = abs($t2); | Valor absoluto. |

## Instrucciones lógicas

|  |  |
| --- | --- |
| $t1 = !$t2; | Not, si $t2 es 0 $t1 es 1, si $t2 es 1 $t1 es 0. |
| $t1 = $t2 && $t3; | And, 1 para verdadero, 0 para falso. |
| $t1 = $t2 || $t3; | Or, 1 para verdadero, 0 para falso. |
| $t1 = $t2 xor $t3; | Xor, 1 para verdadero, 0 para falso. |

## Instrucciones bit a bit

|  |  |
| --- | --- |
| $t1 = ~$t2; | Not. |
| $t1 = $t2 & $t3; | And. |
| $t1 = $t2 | $t3; | Or. |
| $t1 = $t2 ^ $t3; | Xor. |
| $t1 = $t2 << $t3; | Shift de $t2, $t3 pasos a la izquierda. |
| $t1 = $t2 >> $t3; | Shift de $t2, $t3 pasos a la derecha. |

## Instrucciones relacionales

|  |  |
| --- | --- |
| $t1 = $t2 == $t3; | $t1 = 1 si $t2 es igual a $t3, sino 0. |
| $t1 = $t2 != $t3; | $t1 = 1 si $t2 no es igual a $t3, sino 0. |
| $t1 = $t2 >= $t3; | $t1 = 1 si $t2 es mayor o igual a $t3, sino 0. |
| $t1 = $t2 <= $t3; | $t1 = 1 si $t2 es menor o igual a $t3, sino 0. |
| $t1 = $t2 > $t3; | $t1 = 1 si $t2 es mayor a $t3, sino 0. |
| $t1 = $t2 < $t3; | $t1 = 1 si $t2 es menor a $t3, sino 0. |

## Arreglos y Structs

|  |  |
| --- | --- |
| $t1 = array(); | Define $t1 como un arreglo o un struct, para diferenciarlos se utiliza ya sea el valor numérico o el nombre asociativo. |
| $t1[4] = 1; | Asignación de un valor numérico (1) a un índice del arreglo (4). |
| $t1['nombre'] = 'carlos';  $t1["nombre"] = "carlos"; | Asignación de un valor cadena (carlos) a un componente del struct (nombre). |
| $t1 = $t1[4]; | Acceso a un índice del arreglo. |
| $t1 = $t2['nombre']; | Acceso a un componente del struct. |
| $t1 = 'hola';  print($t1[0]); #imprime h | Acceder a un carácter de una cadena. |

## Instrucciones de control

|  |  |
| --- | --- |
| if ($t1) goto label; | Salto condicional, si $t1 es 1 salto, sino si $t1 es 0 sigue a la siguiente instrucción, $t1 puede ser cualquier tipo de instrucción simple. Las estructuras de control se implementan utilizando este salto condicional. |

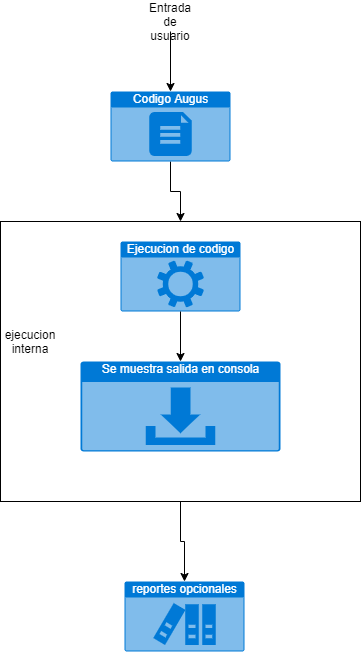
# Flujo de la aplicación

1. Se ingresa el código en Augus ya sea por archivo o editado.

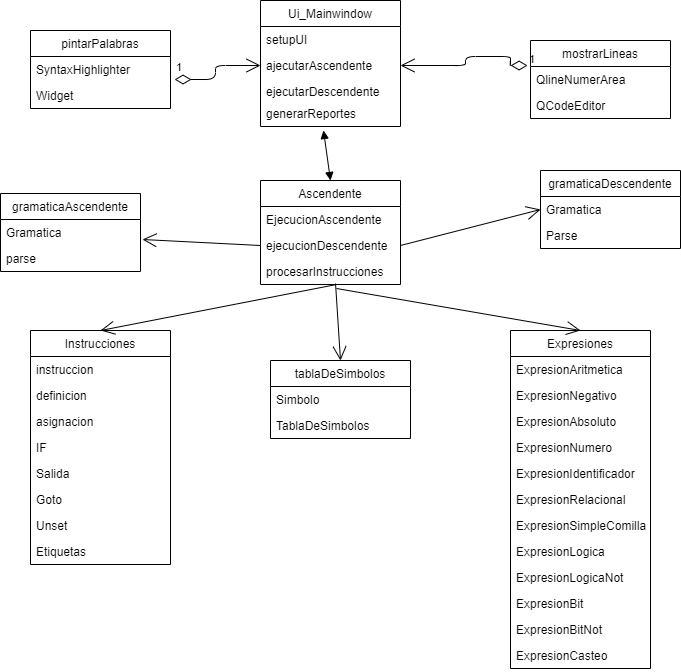
2. Se ejecuta el código con el menú y/o con un ícono (o paso a paso).

3. El resultado de la ejecución se muestra en la consola.

4. Opcionalmente se pueden acceder a los reportes.



# Diagrama de clases



# Librerías utilizadas

* PyQt5
* Graphviz
* Enum
* Random
* Sys
* PLY