Universidad Tecnológica Centroamericana

Compiladores 1

Catedrático: Ing. Carlos Vallejo

Manual de Usuario

Estudiantes:

Andrea Escobar - 11641363

Eduardo Guevara - 11711078

Tegucigalpa, M. D. C.

29 de septiembre del 2020

Lenguaje: WAP

1. Identificadores:

a. Todos los identificadores de cualquier tipo comienzan con underscore(_).
b. Ejemplos:
   i. int _variable
   ii. char _variableDeChar
   iii. bool _valorBooleano
   iv. func _nombreDeFuncion

2. Estructura de las funciones:

a. Todas las funciones deben ir antes del main.
b. Todas las funciones comienzan con la palabra reservada “func”.
c. Después de la palabra reservada func va el tipo de valor de retorno.
d. En tercer lugar va el identificador de la función.
e. Por último va los paréntesis, y dentro de los paréntesis van los parámetros que la función recibe.
f. Los parámetros dentro del paréntesis en caso de ser varios van separados por coma.
g. Ejemplo:
   i.
```
func int _funcion () begin
        return 1;
end func
```
   ii.
```
func bool _funcionBooleana (bool _t) begin
        return _t;
end func
```

3. Bloque Condicional (IF):

a. Todos los bloques condicionales comienzan con la palabra reservada “if”, seguido por la condición entre paréntesis.
b. Ejemplo:
   i.
```
if (_a > _b) begin
        print (“A es mayor que B”);
end if
```

4. Bloque Condicional (ELSE):

a. Todos los bloques condicionales del tipo else comienzan con la palabra reservada “else”, este puede ir seguido de una palabra reservada “if” y luego seguido por la condición entre paréntesis.
b. No es necesario incluir la palabra “if” luego de “else”, si no se utiliza, simplemente no lleva condición.
c. Ejemplo:
    i. else if (_a > _b) begin
        print (“A es mayor que B”);
    end else
    ii. else begin
        print (“B es Mayor que A”);
    end else

5. Estructura de Ciclo (FOR):

a. Todos los ciclos del tipo for comienzan con la palabra reservada “for”, seguido de la palabra reservada “int”, luego una variable de control para el ciclo y un operador hasta “->” para indicar la cantidad de veces que este va a realizar el ciclo, luego de eso va la variable seguida de un “++” que significa un incremento de uno en uno, todo esto entre paréntesis.
b. El operador hasta es inclusivo, es decir, la parte va desde la parte izquierda (tomándolo en cuenta, siempre es 0) hasta el lado derecho (tomando en cuenta).
c. Ejemplo:
    i. for (int _i -> 2; _i++) begin
        print (_i);
    end for

6. Ciclo con Condición (Loop):

a. Todos los ciclos con condición comienzan con la palabra reservada “loop” y terminan con un end loop.
b. Ejemplo:
    i. loop (_boolean & true == true) begin

        _boolean = false;

        print(_matriz(0,0));

    end loop

7. Estructura Multi Condición (Options):

a. Todas estas estructuras comienzan con la palabra reservada “options”, seguido de una variable a evaluar y el tipo de esa variable.
b. Cada opción de esta estructura es denominada “opt” y a esta palabra la siguen corchetes encerrando el valor a comparar.
c. Toda instrucción Options debe tener una opción por defecto, denominada opt_def.
d. Ejemplo:
    i. options (_i , int) begin

```
opt [1]{

    _i = 2;

    print(true);

}

opt_def {

    _j = 3;

}
```

end options

8. Salida y Entrada de Datos:

a. Para la salida de datos se utiliza la instrucción “print”.
b. Para la entrada de datos se utiliza la instrucción “read”.
c. Ejemplos:
    i. print (“Hola”);
    ii. print(_variable);
    iii. read _i;
    iv. read _variable;

9. MAIN:

a. El main lleva al igual que una función normal, empieza con la palabra reservada “func”.
b. Hace uso de la palabra reservada “int_main()”.
c. Ejemplo:
    i. func int_main() begin

```
    return 0;
```

end main

10. Comentarios:

a. Los comentarios se hacen por medio de /* para abrir y */ para cerrarlos.
b. Ejemplo:
    i. /* Este es un comentario */

11. Fin de Línea:

a. Todas las instrucciones simples (No estructuras) terminan con un “;”.
b. Ejemplos:
    i. int _entero;
    ii. _entero = 5;
    iii. print (_entero);
    iv. read _entero;

12. Tipos:

Tipo	Asignación	Acceder a una posición
Int	5	-
char	‘a’	-
bool	true | false	-
array_int	(1,2,3)	_nombrearray(0)
array_char	(‘a’,’b’,’c’)	_nombrearray(0)
array_bool	(true)	_nombrearray(0)
matrix_int	((1,2), (4, 7))	_nombrematriz(0,0)
matrix_char	((‘a’,’b’), (‘c’,’d’))	_nombrematriz(0,0)
matrix_bool	((true,false),(false,false))	_nombrematriz(0,0)

13. Operadores:

Operadores	Función
=	Asignación
==	Comparación
!=	Distinto
<	Menor que
>	Mayor que
<=	Menor o igual que
>=	Mayor o igual que
+	Suma
-	Resta
*	Multiplicación
/	División
->	Hasta (uso en for)
;	final de línea
++	sumar 1
&	AND
|	OR
!	NOT

Ejemplos de Código Fuente

Funcionales

```
func int _funcion (int _a, char _b, bool _t) begin
        int _i = 3, _j, _c = 2+2;
    matrix_int _k = ((true, false, true));
    read _i;
    array_char _l = ('a'), _h, _g = ('a', 'b', 'c');
        _i = 2 +_b;
    _j = ((true, false, true), (true));
        if ((_i > _j) & true | false) begin
                if(true) begin
                        _i = 2*5/2+3;
                end if
        end if
        else if(true) begin
                for(int _i -> 2, _i ++) begin
                end for
        end else
    else if(false) begin
    end else
    else begin
    end else
    if(false) begin
    end if
    else begin
    end else
    return _i;
end func
func int_main() begin
        loop (true & false | !(_b)) begin
        _funcion(_b, 5+5);
        end loop
        options (_i , int) begin
                opt [1]{
                        _i = 2;
                        print(true);
                }
                opt_def {
                        _j = 3;
                }
        end options
        return 0;
end main
```

```
func int_main() begin
	loop (!(_b)) begin
	_PrintSumar(_b, 5+5);
	end loop
	options (_i , int) begin
		opt [1]{
			_i = 2;
			print(true);
		}
		opt_def {
			_i = 3;
		}
	end options
	return 0;
end main
```

```
func int_main() begin
	int _i;
	read _i;
	print(_i);
	for(int _j -> 10, _i ++) begin
		if (_j >= 5) begin
			print (_j);
		end if
		else begin
			print (“La cantidad de Ciclos realizados es menor a 5);
		end else
	end for
	return 0;
end main
```

No-Funcionales

```
func int _funcion (int _a, char _b, bool _t) begin
	int _i = 3, _j, _c = 2+2;
	matrix_int _k = ((true, false, true));
	read ;
	array_char _l = ('a'), _h, _g = ('a', 'b', 'c');
	_i = 2 +_b
	_j = ((true, false, true), (true));
```

```
        if ((_i >= _j) & true | false) begin
                if(true) begin
                        _i = 2*5/2+3;
                end if
        end if
        else if(true) begin
                for(int _i -> 2, _i ++) begin
                end for
        end else
    else if(false) begin
    end else
    else begin
    end else
    return _i
end func
func int_main() begin
        loop (true & false | !(_b)) begin
        _funcion(_b, 5+5);
        end loop
        options (_i , int) begin
                opt [1]{
                        _i = 2;
                        print(true);
                }
                opt_def {
                        _j = 3;
                }
        end options
        return 0;
end main
```

```
func int_main() begin

        loop (true & false | !(_b)) begin

         _funcion(_b, 5+5);

        end loop

        options (_i , int) begin

                opt [1]{

                        _i = 2
```

```
                print(true;
            }
            opt_def {
                _j = 3;
            }
        end options
        return 0;
end main
```

```
func bool _funcBool() begin ola
        int _i = 3;
    int _b = 2;
        bool _val = false;
        if (_i >= 3) begin
            _val = true;
          end if
    return _val;
end func
/* Final de la Función Booleana _funcBool()*/
func int_main() begin
        loop (_i<10) begin
            _funcBool();
        end loop
        options (_i , int) begin
            opt [1]{
                _i = 2;
                print(true);
            }
            opt_def {
                _j = 3;#
            }
        end options
        return 0;
end main
```
