**Acoustatic - Manual de Usuario**

TIPOS DE VARIABLES

Existen distintos tipos de variables los cuales este lenguaje puede manejar. A continuación te indicamos su naturaleza y asignación

|  |  |
| --- | --- |
| Enteros | int entero = 1; |
| Flotantes | fl flotante = 1.0; |
| Strings | string texto = “uno”; |
| Booleanos | boolean booleano = true; |
| Sonidos | Sound Sonido {tone: 1; volume: 1; intensity: 1; pitch : 1} |
| Instrumentos | Instrument Instrumento { Sonido; } |
| Pistas | Track Pista { Instrumento; } |

PROCEDIMIENTOS

Existen varios tipos de procedimientos que pueden aplicarse a las variables antes vistas. A continuación se muestran así también como una breve explicación de qué hace cada acción

|  |  |
| --- | --- |
| Suma | Utiliza el símbolo ‘+’. Regresa la sumatoria de dos valores |
| Resta | Utiliza el símbolo ‘-’. Regresa la resta de dos valores |
| Multiplicación | Utiliza el símbolo ‘\*’. Regresa el producto de dos valores |
| División | Utiliza el símbolo ‘/’. Regresa la división de dos valores |
| Remitente | Utiliza el símbolo ‘%’. Regresa el remitente de dos valores |
| Mayor que | Utiliza el símbolo ‘>’. Regresa la comparación entre el primer número y el segundo. |
| Mayor igual que | Utiliza el símbolo ‘>=’. Regresa la comparación entre el primer número y el segundo. |
| Menor que | Utiliza el símbolo ‘<’. Regresa la comparación entre el primer número y el segundo. |
| Menor igual que | Utiliza el símbolo ‘<=’. Regresa la comparación entre el primer número y el segundo. |
| Igual igual | Utiliza el símbolo ‘==’. Regresa la comparación entre el primer número y el segundo. |
| Igual | Utiliza el símbolo ‘=’. Asigna un valor a una variable. |
| Y | Utiliza el símbolo ‘&&’. Verifica que dos valores sean verdaderos. |
| O | Utiliza el símbolo ‘||’. Verifica que uno de dos valores sea verdadero. |
| No | Utiliza el símbolo ‘!!’. Regresa lo contrario de una expresión boleana |
| No es Igual | Utiliza el símbolo ‘!=’. Verifica que dos valores sean distintos. |
| Print | Utiliza la palabra reservada “print” . Imprime en pantalla la variable o constante dada. |

Los procedimientos a continuación se realizan para cualquier tipo de variables con las restricciones siguientes:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **OP1** | **OP2** | **+** | **-** | **\*** | **/** | **%** | **<** | **<=** | **>** | **>=** | **<>** | **!!** | **&&** | **||** | **=** | **==** |
| **Int** | **int** | I | I | I | F | I | B | B | B | B | B | - | - | - | - | B |
| **Int** | **float** | F | F | F | F | F | B | B | B | B | B | - | - | - | - | B |
| **Int** | **bool** | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| **Int** | **String** | S | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| **float** | **int** | F | F | F | F | F | B | B | B | B | B | - | - | - | - | B |
| **float** | **float** | F | F | F | F | F | B | B | B | B | B | - | - | - | - | B |
| **float** | **bool** | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| **Float** | **String** | S | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| **bool** | **int** | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| **bool** | **float** | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| **bool** | **bool** | - | - | - | - | - | - | - | - | - | - | - | B | B | - | - |
| **Bool** | **Str** | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| **Str** | **int** | S | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| **Str** | **float** | S | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| **Str** | **bool** | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| **Str** | **str** | S | - | - | - | - | - | - | - | - | B | - | - | - | - | B |
| **Int** | **null** | - | - | - | - | - | - | - | - | - | - | - | - | - | I | - |
| **Float** | **null** | - | - | - | - | - | - | - | - | - | - | - | - | - | F | - |
| **Bool** | **null** | - | - | - | - | - | - | - | - | - | - | B | - | - | B | - |

METODOS

Las instrucciones vistas anteriormente pueden ser agrupadas de una forma especial. Este tipo de agrupación se le conoce como métodos o procedimientos y son la base para la creación de acciones complejas dentro de cualquier lenguaje de programación.

Los métodos están caracterizados por poseer una sintaxis y forma diferentes a una variable convencional. Dicha sintaxis empieza con varios elementos:

un tipo de variable, identificador, parámetros y llaves ({})

Estos elementos son básicos a la hora de crear un procedimiento. A continuación se planteará de forma rápida las posibilidades

Uno de los usos más comunes de los métodos, es el de utilizar métodos para obtener un resultado o realizar una operación. Esto se puede conseguir de dos formas:

**Métodos que Regresan Variable**

(int, float, string, boolean, ...) ID(int param1)

{

int variable = 0;

[AQUI VA TODO EL PROCEDIMIENTO]

return variable

}

**Métodos que NO Regresan Variable**

void ID(int param1)

{

[AQUI VA TODAS TUS INSTRUCCIONES]

}

**Método play**

El método play es un método especial, este sólo hace llamadas a otros métodos y prepara para la ejecución de la pieza musical a corresponder. A continuación se adjunta un ejemplo que asume que de declaración las variables adecuadas\*.

play{

9000:12000

{

seccionTambor; //no hace loop, se caya cuando acaban sus elementos

//manejo de if, si haces :duration en un instrumento toma la mayor duracion

//activa la condicion loops si se cumple el estatuto

seccionFlauta:loops if(seccionTambor:duration > seccionFlauta:flauta\_3:duration)

}

modifier1; //ejecuta este metodo

setTime:seccionTambor:1000

12500:50000

{

}

}

\*El código ejemplo es una demostración de la estructura básica de play, se requiere depuración, declaracion de variables y corrección sintáctica para poder ser utilizado

**Método Main**

El método main, corresponde a la pieza principal de ejecución del programa. Haciendo uso de este, uno dicta el orden de los demás procedimientos, el manejo de variables y los resultados obtenidos.

Main siempre es de tipo void ya que no posee metodo al cual regresarle un tipo de dato

**Código Ejemplo**

Como referencia adicional, adjuntamos un codigo ejemplo funcional:

|  |  |
| --- | --- |
| //Manejo de Arreglos  acoustatic piece Sorting  {  int nada = 0;  new instruments{  something  }    function int fibonacci(int n){  int x = 0;  }    play{  nullmethod(0);  }  function void main(){  int osom = 1330 + 7;  int arr[5];  arr[0] = 9;  arr[1] = 1;  arr[2] = 3;  arr[3] = 8;  arr[4] = 4;  arr[3] = 5;    boolean sorted = false;  boolean comparator = true;  int tama = 5;  int iterator = 0;  print(arr[0]);  print(arr[1]);  print(arr[2]);  print(arr[3]);  print(arr[4]); | while(sorted <> comparator)  {  int i = 0-1;  int iMasUno = 0;  sorted = true;  int elemAntes = 0;  int elemDespues = 0;  while(i < tama-2)  {  i = i+1;  iMasUno = i+1;    elemAntes = arr[i];  elemDespues= arr[iMasUno];  if( elemAntes > elemDespues)  {  sorted = false;    int iNormal = arr[i];  int iMas = arr[iMasUno];    arr[iMasUno] = iNormal;  arr[i] = iMas;  }    }  }  print(arr[0]);  print(arr[1]);  print(arr[2]);  print(arr[3]);  print(arr[4]);  }  } |