**Manual de Usuario Pug++**

Estructura de un programa:

|  |
| --- |
| program nombrePrograma;  main() {    } |

Declarar variables (int, float, char) y asignarles un valor:

|  |
| --- |
| program nombrePrograma;  var int x;  main() {      var float a,b;      a = 5;      b = 2;  } |

Declarar un Arreglo o Matriz (int, float, char) y asignarles valores:

|  |
| --- |
| program array;  var int array[3];  main(){      var int matrix[2][2];      array[0] = 4;      array[1] = 8;      array[2] = 4;      matrix[0][0] = 3;      matrix[1][0] = 2;      matrix[1][1] = 5;      matrix[0][1] = 4;      print(array[2]);      print(matrix[1][1]);  } |

Declarar y llamar una función:

\*Si la función NO es tipo void debe tener un statement tipo RETURN, si la función es tipo void NO debe tener RETURN.

|  |
| --- |
| program funcion;  function int suma(int x, int y) {      return(x + y);  }  main(){      var int a;      a = suma(5,20);      print(a);  } |

Leer e imprimir un valor de entrada desde la consola:

|  |
| --- |
| program ejemplo;  main(){      var int x;      read(x);      print(x);  } |

Estatutos condicionales y operadores booleanos:

Menor que: <

Mayor que: >

Igual a: ==

Diferente a : <>

AND : &

OR: |

|  |
| --- |
| program ejemplo;  main(){      var int a;      a = 10;      if(a > 5) then {          print("TRUE");      } else{          print("FALSE");      }  } |

Ciclos WHILE y FOR

|  |
| --- |
| program ciclos;  main(){      int i = 0;      while (i < 10){          print(i);          i = i + 1;      }      for i = 0 to i < 10 {          print(i);      }  } |

Operadores especiales de Matrices

Determinante: $

|  |
| --- |
| program determinanteMatriz;  main() {      var float result;          int i, j, matrix[3][3];        %% assign matrix      matrix[0][0] = 1;      matrix[1][0] = 0;      matrix[2][0] = 1;      matrix[0][1] = 8;      matrix[1][1] = 0 - 1;      matrix[2][1] = 2;      matrix[0][2] = 0 - 2;      matrix[1][2] = 8;      matrix[2][2] = 2;      result = matrix$;      print(result);  } |

Inversa: ?

|  |
| --- |
| program inversaMatriz;  main() {      var int i, j, matrix[3][3];          float result[3][3];        %% assign matrix      matrix[0][0] = 1;      matrix[1][0] = 2;      matrix[2][0] = 0 - 1;      matrix[0][1] = 2;      matrix[1][1] = 0 - 3;      matrix[2][1] = 1;      matrix[0][2] = 0 - 1;      matrix[1][2] = 0;      matrix[2][2] = 3;      result = matrix?;      for j = 0 to j < 3 {          for i = 0 to i < 3 {              print(result[i][j]);          }      }  } |

Transpuesta: !

|  |
| --- |
| program transpuestaMatriz;  main() {      var int i, j, matrix[3][3], result[3][3];        %% assign matrix      matrix[0][0] = 1;      matrix[1][0] = 4;      matrix[2][0] = 7;      matrix[0][1] = 2;      matrix[1][1] = 5;      matrix[2][1] = 8;      matrix[0][2] = 3;      matrix[1][2] = 6;      matrix[2][2] = 9;      result = matrix!;      for j = 0 to j < 3 {          for i = 0 to i < 3 {              print(result[i][j]);          }      }  } |

Para ejecución:

`python parser.py <nombre de archivo o ruta de archivo >`