**MANUAL DE USUARIO**

**PseudoCodeTranslate**

**AUTORES**

**SEBASTIÁN CARMONA AGUIAR**

**INSTITUTO TECNOLOGICO METROPOLITANO**

**ITM**

**MEDELLÍN**

**2020**

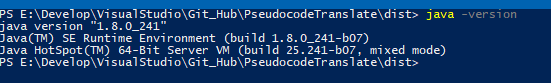
# **Instrucciones para usar programa**

1. Ubicarse en la carpeta donde se encuentra el archivo .jar
2. Verificar que en dicha carpeta se encuentra otra carpeta llamada Tabla, la cual debe contener el archivo TablaConversion.txt
3. Regresar a la carpeta que contiene el archivo .jar y presionar las teclas Shift + clic derecho, lo que abrirá un menú
4. Seleccionar la opción Abrir Ventana de comando, CMD o PowerShell (puede aparecer con estos nombres)
5. Copiar el siguiente comando:

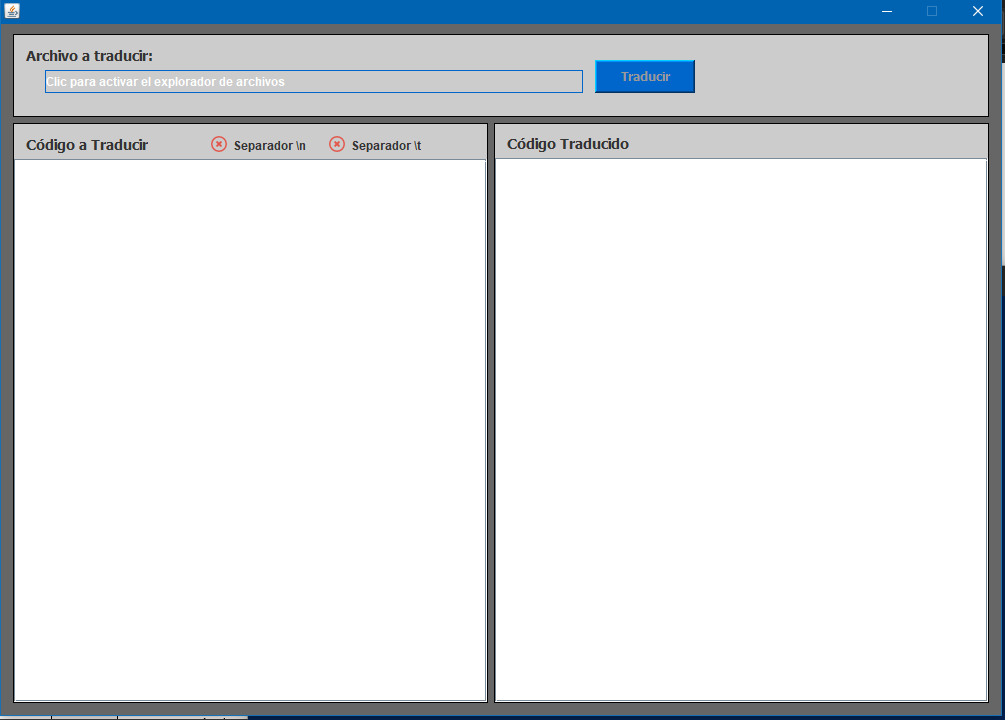
java -jar “PseudocodeTranslate.jar”

**Nota:**

1. el commando java funciona siempre y cuando ya se haya realizado la respectiva configuración de las variables de entono, para confirmar si el comando está configurado usar: java -version y debe recibir una respuesta similar a la siguiente:



1. para pegar comandos basta con copiar el texto, regresar a la ventana de comandos y presionar clic derecho
2. Una vez se haya pegado el comando anterior, presionar Enter, el programa estará esperando alguno de los siguientes valores:
   1. 1 = activa la ventana gráfica del traductor
   2. 2 = mantiene la traducción por ventana de comandos
3. Habitualmente se usará la opción 1, presionar Enter nuevamente y cargará versión gráfica del traductor



**2**

**4**

**3**

**1**

1. Al dar clic abrirá el menú para seleccionar el archivo que contiene el código a ser traducido
2. Se activará una vez se haya seleccionar el archivo que contiene el código a traducir y que este posea el formato correcto (saltos de línea o tabulaciones)
3. Text área donde se presentará el código contenido en el archivo que se desea traducir
4. Text área que presentará el código traducido

# **Reglas de traducción**

El traductor usa como base el archivo llamado TablaConversion.txt, allí se encuentran todas las palabras reservas del lenguaje pseudocodigo, además de las reglas que se aplicarán a cada palabra reserva.

A continuación, se detalla la lógica de cada una de las reglas configuradas:

|  |  |  |
| --- | --- | --- |
| **NRO** | **DESCRIPCIÓN** | **EJEMPLO** |
| **0** | 1. Buscar palabra reservada 2. Sustitución directa de la palabra | **Regla:**  "0";"finmetodo";"}"  **Valor original:**  finmetodo  **Paso 2: sustitución**  }  **Resultado:**  } |
| **1** | 1. Buscar palabra reservada 2. Sustitución directa de la palabra 3. Captura de texto de acuerdo con el RegEx 4. El carácter especial .? es reemplazado por el texto capturado | **Regla**:  "1";"clase";"class";"(class\s\w+[^\s])";".?{"  **Valor original:**  publico clase viajando  **Paso 2: sustitución**  publico class viajando  **Paso 3: captura texto**  class viajando (regex)  **Paso 4: reemplaza carácter especial**  Reemplaza **.?** por **class viajando**  **Resultado:**  publico class viajando{ |
| **2** | 1. Buscar palabra reservada 2. Captura de texto de acuerdo con el RegEx 3. Sustitución de la palabra 4. Usar formato y reemplazar carácter especial .?{ por el texto capturado | **Regla:**  "2";"\*\*";"(([a-z0-9])+\\*\\*([a-z0-9])+)";"\*\*";",";"Math.pow(.?)"  **Valor original:**  a = b\*\*c  **Paso 2: captura texto**  b\*\*c  **Paso 3: sustituir \*\* por ,**  b,c  **Paso 4: Usar formato y reemplaza carácter especial**  Math.pow(.?) -> reemplaza .? por a,b  **Resultado:**  Math.pow(b,c) |
| **3** | 1. Buscar palabra reservada 2. Captura de texto de acuerdo con el RegEx 3. Sustitución de la palabra reservada   Nota: se podría usar la regla 0, sin embargo, la palabra reservada mod, podría estar dentro del nombre de una variable, método, etc. Por lo tanto, se requiere de una expresión regular para evitar hacer reemplazos inadecuados | **Regla:**  "3";"mod";"(\smod\s)";"%"  **Valor original:**  a = b mod c  **Paso 2: captura texto**  b mod c  **Paso 3: sustituir mod por %**  b % c  **Resultado:**  a = b % c |
| **4** | **Captura de variables y traducción**   1. Buscar palabra reservada 2. Sustitución de la palabra 3. Captura nombre de variables usando el RegEx y separando por “,”, además se aplican condiciones internas para retirar valor inicial de las variables 4. Valida si el texto cumple la condición anidada para usar segundo parámetro de reemplazo (válida que la variable no sea un parámetro de un método) 5. De aplicar segunda condición, con el RegEx se captura línea completa de variables y valores iniciales, usando la estructura .?; reemplaza valor .? con el texto capturado, logrando colocar ; al final | **Regla:**  "4";"entero";"int";"(int .\*[^\n\t])";",";".?;";"{<>:metodo}"  **Valor original:**  entero n1=0, n2=1  **Paso 2: sustitución**  int n1=0, n2=1  **Paso 3: captura nombre de variables**  n1  n2  **Paso 4 y 5: válida si aplica condición anidada y reemplaza**  .?; -> .? se reemplaza por int n1=0, n2=1  **Resultado:**  int n1=0, n2=1; |
| **5** | **De acuerdo con las variables previamente capturas crea sentencia Scanner**   1. Buscar palabra reservada 2. Captura todo el texto con el RegEx 3. Sustituye la palabra reservada 4. Identifica la variable y el tipo de dato, de acuerdo con la información obtenida en la regla anterior, usando la condición anidada crea sentencia | **Regla:**  "5";"lea ";"";"(lea .\*[^\n\t])";"{int: = new Scanner(System.in).nextInt();,double: = new Scanner(System.in).nextDouble();,boolean: = new Scanner(System.in).nextBoolean();,String: = new Scanner(System.in).nextLine()}"  **Valor original:**  lea pesos  **Paso 2: captura**  lea pesos  **Paso 3: sustituye lea por “ ”**  pesos  **Paso 4 y 5: identifica variable y tipo de dato, usa condición anidada y crea estructura**  pesos = new Scanner(System.in).nextInt();  **Resultado:**  pesos = new Scanner(System.in).nextInt(); |

**Consideraciones:**

* Los valores de texto literal o que se encuentran entre comillas simples (‘’) o dobles (“”), no son alterados por el traductor, aunque su contenido coincida con alguna de las palabras reservadas.
* Las condiciones anidadas se identifican por estar entre corchetes “{…}”; para múltiples condiciones anidadas se usa el separador coma “,”
* Las RegEx (Expresiones Regulares) siempre deben estar encerradas entre paréntesis (…)