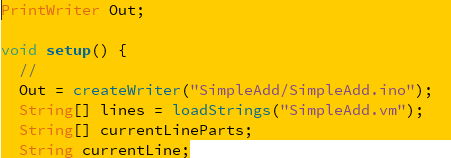
Documento De Funcionamiento Para Simple Add

Este trabajo consiste en leer un archivo .VM y hacer una traducción por medio de processing para que el archivo pueda ser leído en la plataforma de Arduino.

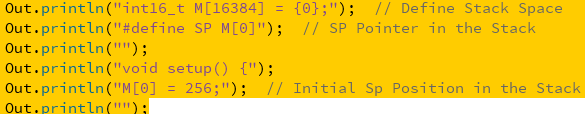
Para esto empezamos a utilizar algunas funciones específicas del lenguaje de processing( Java) para facilitar el proceso de lectura y traducción del archivo



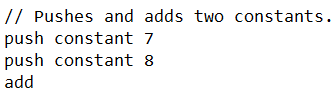
Primero creamos una variable PrintWriter que nos permitirá escribir dentro de un archivo de salida, Luego en el Setup del programa definimos el nombre del archivo de salida y la dirección, en este caso el archivo de salida llevará extensión. ino y se almacenará en dentro de la carpeta “SimpleAdd” con el nombre de SimpleAdd.ino

Luego se crea un arreglo llamado lines que se encargara de leer cada una de las líneas de el archivo .vm que se encuentra dentro de la carpeta donde está ubicado el programa de modo tal que tendríamos algo así :

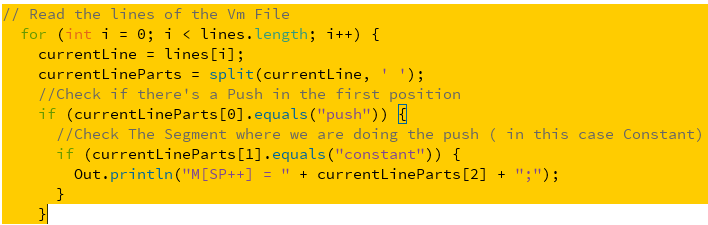




Antes de iniciar la traducción nos aseguramos de declarar las variables que necesitaremos en Arduino para el correcto funcionamiento del programa, en este caso necesitamos definir el tamaño del Stack, el Puntero de Sp en el stack, y la posición inicial de este puntero en el Stack.

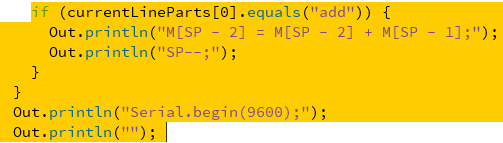


Para este ejemplo especifico debemos hacerle Push a 2 diferentes constantes en el stack y luego hacer un add entre ambas.

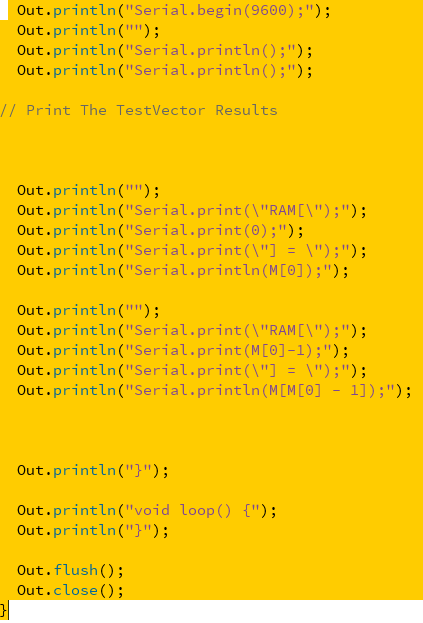


Para esto tendremos el arreglo currentLineParts que nos servirá para almacenar por aparte cada parte de las diferentes líneas del archivo.Vm utilizando la función Split que se encargara de separar las cadenas cada vez que exista un espacio y la string, currentLine almacenará las diferentes posiciones de las lines del archivo.vm.

El código anterior se encarga de leer las diferentes líneas del archivo y si encuentra un push en la posición 0, y luego un constant en la posición 1 , creará una posición después del Sp en donde estará un carácter que se encuentra en la segunda posición del archivo, de modo que si tenemos un push constant 7, Añadiera un 7 despues del SP y asi sucesivamente con la cantidad de Push que el archivo.vm contenga.



Luego verificaremos si en el archivo hay un Add y en caso de que exista lo que haremos será sumar los valores que estaban en las 2 posiciones anteriores al stack , lo almacenaremos en la posición anterior al stack y finalmente reduciremos el stack.



Finalmente inicializaremos el serial de Arduino para poder visualizar los resultados del vector de prueba y compararlos con los de el archivo.vm , abriremos y cerraremos la función loop para un correcto funcionamiento dentro del IDE de Arduino, y utilizaremos la función Flush para asegurarnos que no quede ningún dato en el buffer y la función close para cerrar el archivo una vez todo el proceso halla finalizado,