**11 Notas sobre la implementación**

Descripción de la implementación realizada.

**11.1. Descripción de archivos**

Hemos divido el proyecto es 3 paquetes principales:

* **Compilador:** Contiene las clases relativas al traductor de lenguaje.  Este paquete contiene 3 paquetes:

1. Lexico: contiene la clase AnalizadorLexico, que recibe un flujo de entrada y devuelve un ArrayList de Tokens. Este paquete contiene un paquete Tokens con clases que corresponden a cada uno de los tipos de tokens que maneja el Analizador Léxico.
2. TablaSimbolos: Contiene varias clases, la principal es GestorTS. Este gestor es una pila de Tablas de Símbolos(clase TablaSimbolos). A su vez esta tabla de símbolos tiene información como modo, tipo(clase TipoTs), dirección y nivel. El tipo se apoya en otras clases como Campo.
3. Traductor: Contiene las clases necesarias para traducir un ArrayList de Tokens procedente del analizador léxico a un ArrayList de Objetos que contendrá el código binario. La clase que gestiona la traducción es Traductor y existen otras como ErrorTraductor que gestiona los errores.

* **Interfaz:** Contiene las interfaces utilizadas de cara al usuario. Como hemos implementado dos programas separados (compilador e intérprete) tenemos dos interfaces:

1. Compilador: Contiene la interfaz del compilador. Esta interfaz da la opción de introducir el código en la propia interfaz o cargarla desde un fichero, una vez cargado puedes compilar y ver el código pila o ejecutar (en cuyo caso compilará y luego ejecutará el programa) También nos ofrece ejecutar el código en modo Traza (mostrando el contenido de la pila y la memoria en cada instrucción además de las entradas/salidas del programa) o en modo Normal (mostrando únicamente las entradas/salidas del programa)
2. Pila: Contiene una interfaz que hemos utilizado para probar el intérprete a pila. Este panel hace de intermediario entre el bytecode del lenguaje a pila y el explicado en clase (con sentencias alfanuméricas como "apila 3" o "suma"). Al descompilar un archivo en bytecode este se mostrará como cadenas alfanuméricas. Al compilar, el texto escrito será traducido a lenguaje de pila, siempre y cuando su sintaxis sea correcta

* **Pila:** Contiene el intérprete encargado de simular la ejecución del código . Como tenemos dos tipos de código (código P y código J) este paquete contiene dos paquetes:

1. Intérprete: Se encarga de ejecutar el código P generado por el compilador. Contiene 3 paquetes que se encargan de gestionar tanto los datos, como las instrucciones y excepciones que pueden surgir en el código y 3 clases que son las principales (EscritorPila, Interprete y LectorPila) que gestionan la entrada/salida de la ejecución y la propia ejecución. En este cuatrimestre hemos creado dos clases mas (Memoria y Huecos) para gestionar la memoria estática y el heap.