## Interpreter:

Most of these changes are inspired by the idea that each function should have a specific purpose. Many of the new functions contain a subset of the functionality described by a single function from the design document.

- Interpreter Class

* + Variables:
    - Added:
      * currentelement: a package private integer which contains the index of the current word in the forthWords list.
      * robot: A private reference to the Before this was only stored in InterpreterFunctions but it was also included here to simplify refferences to it.
      * acceptedWords: A private dictionary of the ForthWords as defined by the Forth Language document. This will allow us to check for the correctness of code, as well as easily reffering to which function each ForthWord should correspond to.
  + Functions:
    - Added:
      * - loadWords: A helper function to the play function which will deal with all Forth code parsing.
      * -loadDict: A function to load the dictionary of accepted Forth words from words.txt

- InterpreterFunctions Class

* + Variables:
    - Added:
      * loopStack: Stack - A stack to store loop structs, which contain all necessary information for loop execution.
      * conditionalStack: Stack - A stack to store boolean which will be used to determine which part of the conditional to run.
      * Interpreter: Intepreter - The interpreter is stored in this variable so that the inteprter can be accessed from this class.
  + Functions:
    - Added:
      * +store(): void - Stores a value in the given variable
      * +load(): void - Loads a value from the given variable and pushes it to the stack
      * +begin(): void - Initializes a while loop struct and pushes it to the loopStack
      * +until(): void - Pops a boolean and decides whether or not to loop
      * +do(): void - Initializes a loop struct with a for loop and push it to loopStack
      * +loop(): void - Decide whether or not to loop based on the top element on the loopStack
      * +ifCondition(): void - Add a boolean to the conditionalStack and call seekTo to skip to the else branch if that boolean was false
      * +elseCondition(): void - Remove a boolean to the conditionalStack and call seekTo to skip to the then branch if that boolean was true
      * -checkBool(String element): void - throws an exception if the given element is not a boolean.
      * -checkInt(String element): void - throws an exception if the given element is not an int
      * -seekTo(String element): void - changes Intepreter.currentElement so that it points to the next occurence of the given String
  + New Internal Class:
    - loopStruct class
      * Variables:
        + - start : int - The value at which the loop will start
        + -end: int - The value at which the loop will end
        + -i : int - The loop’s current increment
        + -startIndex: int - The index of the word which begins the loop. This is the only field relevant to a guarded loop