Here are some fairly straightforward modifications that can be made to your Jack compiler.

All of these mods are strictly extensions of the book specified Jack language. Jack programs that compile with the supplied JackCompiler will be compatible with your extended compiler, except for identifier name clashes with new keywords.

**Easy**

These projects are basically cookbook recipes if you wrote a recursive descent compiler following the modules and methods recommended in projects 10 and 11.

[Making "let" optional in the Jack Compiler](http://nand2tetris-questions-and-answers-forum.32033.n3.nabble.com/Making-let-optional-in-the-Jack-Compiler-tp4030849.html)

[Removing the Requirement for Braces in Jack's "if" and "while" Statements](http://nand2tetris-questions-and-answers-forum.32033.n3.nabble.com/Removing-the-Requirement-for-Braces-in-Jack-s-if-and-while-Statements-tp4030847.html)

[Adding Hexadecimal and Character Constants to Jack](http://nand2tetris-questions-and-answers-forum.32033.n3.nabble.com/Adding-Hexadecimal-and-Character-Constants-to-Jack-tp4030848.html)

**Medium**

These projects are more narrative in nature. They require more extensive changes to the compiler and have less detailed implementation information.

[Adding Standard Operator Precedence to the Jack Compiler](http://nand2tetris-questions-and-answers-forum.32033.n3.nabble.com/Adding-Standard-Operator-Precedence-to-the-Jack-Compiler-tp4028444.html)

[Adding "break" and "continue" to Jack's while loops](http://nand2tetris-questions-and-answers-forum.32033.n3.nabble.com/Adding-break-and-continue-to-Jack-s-while-loops-tp4028649.html)

[Adding Named Constants to Jack](http://nand2tetris-questions-and-answers-forum.32033.n3.nabble.com/Adding-Named-Constants-to-Jack-tp4027626.html)

[Add missing compare, logical, and augmentation ops to Jack](http://nand2tetris-questions-and-answers-forum.32033.n3.nabble.com/Add-missing-compare-logical-and-augmentation-ops-to-Jack-tp4031914.html)

[Adding "switch/case" and "for" statements to Jack](http://nand2tetris-questions-and-answers-forum.32033.n3.nabble.com/Adding-switch-case-and-for-statements-to-Jack-tp4031937.html)

**Complex**

These projects require extensive changes to underlying objects like SymbolTable and VmWriter, multiple source code passes through multiple .jack files, or the implementation may require some non-standard techniques to work around limitations in the VmEmulator.

[Adding Nested Variable Scopes to the Jack Compiler](http://nand2tetris-questions-and-answers-forum.32033.n3.nabble.com/Adding-Nested-Variable-Scopes-to-the-Jack-Compiler-tp4030944.html)

**Other**

These are discussions and not proposals for projects that can be reasonably added to the recursive descent compiler described in the book.

[Adding symbol resolving, type checking, inheritance and polymorphism](http://nand2tetris-questions-and-answers-forum.32033.n3.nabble.com/Adding-symbol-resolving-type-checking-inheritance-and-polymorphism-tp4028523.html)

[Writing a Jack debugger](http://nand2tetris-questions-and-answers-forum.32033.n3.nabble.com/Writing-a-Jack-debugger-tp4028595.html)

["Improving" evaluation of [boolean] expressions?](http://nand2tetris-questions-and-answers-forum.32033.n3.nabble.com/Improving-evaluation-of-expressions-tp4029034.html)

[Some fun with SSA](http://nand2tetris-questions-and-answers-forum.32033.n3.nabble.com/Some-fun-with-SSA-tp4027631.html)

Jump in and try your hand at language design and new feature implementation.

Post your proposals here and include how the grammar needs to be changed to include your change, and a rough idea of what changes need to be made in the compiler.

We can discuss the ideas and help you implement your feature.

Notes:

1. Proposed features **do not** need to be compatible with any other complier mods, but do need to be strict extension to the book spec.
2. Features that need a change in the VM language to support will restrict your audience to students who are willing to modify their VM translator, and will only be testable in the CPU emulator.
3. Features need to keep the grammar LL(1) so that the simple recursive descent design can handle it.