Intro to Computer Systems :: Project 8: VM Translator II

Student name: Cham

***Grading method***

In evaluating the second part of your VM Implementation program (AKA Translator) we look into the following things:

 ***Code quality:*** since the software design was provided, your code will be judged mainly for elegance and readability.

 ***Correctness:*** your translator must translate correctly the supplied VM test programs: FibonacciElement and StaticTest. "Correct translation" occurs when the Translator generates Hack code that, when run on the CPU Emulator, generates the desired results.



|  |  |  |
| --- | --- | --- |
| ***Assembler*** |  | ***Comments*** |
| Packaging | 5/ 5 | Directory (folder) with your name on it, containing (1) /src directory of your code, (2) a README file with instructions for compiling and running your program, and comments on things that don’t work and how you tried to fix them. |
| Working? | 40/ 45 | Does the program generate assembly output that will work on all test inputs?  Doesn’t accept directory from command line without typing \*.vm (-5) |
| Well built? | 47/ 50 | 14 points for documentation. But, don't over-­‐document! For every method you write, document what it does, what parameters it takes, and what it returns. Use your judgment to add more documentation when needed.  16 points for a VM translator that produces efficient assembly code. Generally speaking, the fewer the number of generated assembly instructions, the better.  20 points for a good and clean implementation that we can easily read and understand.  This is a lot of code to jam into one file. Try moving each class to new a file to make your code much easier to read (-3) |
| Total | 92/ 100 |  |

Total grade: \_\_\_\_92\_\_\_\_\_