Assignment Description for Project 7 (Virtual Machine II)

The project corresponds to Chapter 8 of the book. Please read the chapter of the book thoroughly. You will continue to write the compiler from the VM language to Hack assembly.

**Objective:** Get a deep understanding of the operation of the VM and of the process of compiling VM code into assembly code, which essentially corresponds to what is done in the ‘backend’ of a modern compiler.

**Contract:** Complete the VM language parser and the VM to Hack assembly compiler by extending the Python source code or the C++ source code that you worked on for Project 6 (Virtual Machine I). Extend 1 member function of the VMParser class and implement 7 member functions of the CodeWriter class. The functions are the following:

void VMParser::parse() //extend

void VMCompiler::writeInit(bool sysinit); //implement

void VMCompiler::writeLabel(string label);

void VMCompiler::writeGoto(string label);

void VMCompiler::writeIf(string label);

void VMCompiler::writeCall(string funcName, int numArgs);

void VMCompiler::writeReturn();

void VMCompiler::writeFunction(string funcName, int numLocals);

**Resources:** You will continue to work with the code that you worked with for Project 6 (Virtual Machine I).

For Background, Details, and Notes please refer to the description of Project 6.

**Details**

Contrary to Project 6, in Project 7 you will have to use Sys.Init once you have implemented the writeInit() method. You will do so by calling (for Python)

python ./hvm.py source1.VM

or (for C++),

./VM2ASMCompiler target.ASM source1.VM

**Testing of the compiler**

1. We recommend that you first complete the implementation of the parse() function. You can test the implementation by creating a VM code file that contains one of each command type and then by looking at the result of the parsing. The result appears on the screen when running the compiler. If the parser is correct, you can switch off this feature.
2. Once the parser is correct you could continue with the implementation of the next three functions (WriteLabel, WriteGoto, and WriteIf) and test them on the ProgramFlow programs provided for Chapter 8 using the CPU emulator (2 tests).
3. If these three are correct you can then go ahead with the next four functions (WriteFunction, WriteCall, WriteReturn and WriteInit) and then test them using the FunctionCalls programs provided for Chapter 8 (3 tests).
4. You should only use the call to sys.init for the last testing, that is, once you have implemented compilation of function calling.

**Submission**

You should submit your source files (for Python submit hvmParser.py and hcmCodeWriter.py ,for C++ submit VMParser.cpp and CodeWriter.cpp,), which should include your implementation of the above named functions.

In one zip-archive, include the source files and the filled in Excel declaration sheet. Please use the convention *projectMM-familyname-firstname-EP1200.zip* for the filename.