Test Data for Development

In this section, I will run somewhat informally through the testing procedures I will use during development. I will split this document into iterations and further into modules (i.e. compiler, interpreter and assembler). I will give a series of sample input, the expected result and other info.

# Iteration 1

## Assembler

|  |  |  |
| --- | --- | --- |
| Input | Expected result | Description |
| mov eax 5 mov ebx 7 add eax ebx  mov eax [ecx] ; Comment # Comment on its own line | The first section of the program successfully removes all of the whitespace | This is the contents of testing/for\_normalisation.asm, and includes everything that the part of the program corresponding to section 1 of decomposition should deal with |
| section.meta mem\_amt 4 name Hello  section.data i VAR 0 j VAR 1  section.text ; Commands here | The program manages to turn this into a dict object, with the three sections as keys and a string of their contents as values | This tests section 2 of decomposition, splitting the document into sections, and is stored in for\_section\_split.asm. |
| section.meta mem\_amt=2  section.data temp VAR ushort 0 i VAR char 50  section.text MOV eax 1 MOV ebx 1 loop MOV temp eax ADD eax ebx ADD ebx temp  SUB i 1 MOV out eax CMP i 0 JNE loop | Section 3 of the program creates a list of commands corresponding to the data and text sections, section 4 creates a mem\_table dict containing the memory addresses of all variables and labels, and section 5 produces a byte string which correctly encodes the document. | As described. While conforming to these expected items does not guarantee the program works (since the bytecode in the end could be wrong), it does show that the assembler can take an assembly file and fully process it into bytecode. |
| The binary output from the previous test (run through the disassembler) | A correct breakdown which, upon inspection, is equivalent to the initial assembly code (though some data, like variable names, is gone) | If this runs correctly, then it proves that the output of the program was indeed correct. |