## Submitted Files

Results | Code

### Abs/Abs.vm

Download

```
// Calculates y = |x|
// x & y are static variables

// Put your code here.
push static 0   // push x
push constant 0 // push 0
lt              // check if x < 0
if-goto NEG     // if true, jump to NEG

// Case 1: x >= 0
push static 0   // push x
pop static 1    // y = x
goto END        // skip negative branch

// Case 2: x < 0
label NEG
push static 0   // push x
neg             // x → -x
pop static 1    // y = -x

// End of program
label END
```

### Abs/Abs00.cmp

Download

```
|   sp    | local   | argument |   this  |  that   |RAM[16] |RAM[17] |RAM[18] | local[0] | local[1] | local[2] | argument[0] | argument[1] | argument[2] |
|    256  |    300  |      400 |   3000  |   3010  |     -1 |      1 |     -3 |      -10 |      -20 |      -30 |        -100 |        -200 |        -300 |

```

### Abs/Abs00.out

Download

```
|   sp    | local   | argument |   this  |  that   |RAM[16] |RAM[17] |RAM[18] | local[0] | local[1] | local[2] | argument[0] | argument[1] | argument[2] |
|    256  |    300  |      400 |   3000  |   3010  |     -1 |      1 |     -3 |      -10 |      -20 |      -30 |        -100 |        -200 |        -300 |

```

**Abs/Abs00.tst** Download

```
// Sample Test file for Abs.vm
// Follows the Test Scripting Language format described in
// Appendix B of the book "The Elements of Computing Systems"

load Abs.vm,
output-file Abs00.out,
compare-to Abs00.cmp,
output-list sp%D1.6.1 local%D1.6.1 argument%D1.8.1 this%D1.6.1 that%D1.6.1
            RAM[16]%D1.6.1 RAM[17]%D1.6.1 RAM[18]%D1.6.1
            local[0]%D1.8.1 local[1]%D1.8.1 local[2]%D1.8.1
            argument[0]%D1.11.1 argument[1]%D1.11.1 argument[2]%D1.11.1;

set sp 256,         // stack pointer
set local 300,      // base address of the local segment
set argument 400,   // base address of the argument segment
set this 3000,      // base address of the this segment
set that 3010,      // base address of the that segment

set RAM[16] -1,   // static 0
set RAM[17] -2,   // static 1
set RAM[18] -3,   // static 2

set local[0] -10,   // local 0
set local[1] -20,   // local 1
set local[2] -30,   // local 2

set argument[0] -100,   // argument 0
set argument[1] -200,   // argument 1
set argument[2] -300;   // argument 2

repeat 7 {        // Change this number to cover the number of instructions in the VM test file
  vmstep;
}
output;

```

**Abs/Abs01.cmp** Download

```
| RAM[16]  | RAM[17]  |
|       7  |       7  |

```

**Abs/Abs01.out** Download

```
| RAM[16]  | RAM[17]  |
|       7  |       7  |

```

**Abs/Abs01.tst** Download

```
// Test 2: x = 7 → y = |x| = 7
load Abs.vm,
output-file Abs01.out,
compare-to Abs01.cmp,
output-list RAM[16]%D2.6.2 RAM[17]%D2.6.2;

set sp 256,
set local 300,
set argument 400,
set this 3000,
set that 3010,

set RAM[16] 7,    // static 0 (x)
set RAM[17] 0;    // static 1 (y)

repeat 7 {
  vmstep;
}
output;

```

**AddSub/AddSub.vm** Download

```
// Calculates x = (a + b) - x
// a & b are local variables
// x is a static variable

// Put your code here.
push local 0   // push a
push local 1   // push b
add            // compute (a + b)
push static 0  // push x
sub            // compute (a + b) - x
pop static 0   // store result back to x
return         // end function
```

**AddSub/AddSub00.cmp** Download

```
|   sp    | local   | argument |   this   |   that   |RAM[16] |RAM[17] |RAM[18] | local[0] | local[1] | local[2] | argument[0] | argument[1] | argument[2] |
|    256 |    300 |      400 |   3000 |   3010 |     29 |      2 |      3 |       10 |       20 |       30 |         100 |         200 |         300 |

```

**AddSub/AddSub00.out** Download

```
|   sp    | local   | argument |  this   |  that   |RAM[16] |RAM[17] |RAM[18] | local[0] | local[1] | local[2] | argument[0] | argument[1] | argument[2] |
|    256 |    300 |      400 |   3000 |   3010 |     29 |      2 |      3 |       10 |       20 |       30 |         100 |         200 |         300 |

```

**AddSub/AddSub00.tst** Download

```
// Sample Test file for AddSub.vm
// Follows the Test Scripting Language format described in
// Appendix B of the book "The Elements of Computing Systems"

load AddSub.vm,
output-file AddSub00.out,
compare-to AddSub00.cmp,
output-list sp%D1.6.1 local%D1.6.1 argument%D1.8.1 this%D1.6.1 that%D1.6.1
            RAM[16]%D1.6.1 RAM[17]%D1.6.1 RAM[18]%D1.6.1
            local[0]%D1.8.1 local[1]%D1.8.1 local[2]%D1.8.1
            argument[0]%D1.11.1 argument[1]%D1.11.1 argument[2]%D1.11.1;

set sp 256,         // stack pointer
set local 300,      // base address of the local segment
set argument 400,   // base address of the argument segment
set this 3000,      // base address of the this segment
set that 3010,      // base address of the that segment

set RAM[16] 1,   // static 0
set RAM[17] 2,   // static 1
set RAM[18] 3,   // static 2

set local[0] 10,   // local 0
set local[1] 20,   // local 1
set local[2] 30,   // local 2

set argument[0] 100,   // argument 0
set argument[1] 200,   // argument 1
set argument[2] 300;   // argument 2

repeat 6 {      // Run enough steps to complete the function
  vmstep;
}

output;

```

**AddSub/AddSub01.cmp** Download

```
| RAM[16]  |
|      15  |

```

**AddSub/AddSub01.out** Download

```
| RAM[16]  |
|      15  |

```

**AddSub/AddSub01.tst** Download

```
// Test 2: a=8, b=12, x=5 → result = (8+12)-5 = 15
load AddSub.vm,
output-file AddSub01.out,
compare-to AddSub01.cmp,
output-list RAM[16]%D2.6.2;

set sp 256,
set local 300,
set argument 400,
set this 3000,
set that 3010,

set RAM[16] 5,      // static 0 (x)
set local[0] 8,     // local 0 (a)
set local[1] 12,    // local 1 (b)

repeat 7 {
  vmstep;
}
output;

```

## Assignment 6

● Ungraded

**Student**

Xinze Zhang

**Total Points**

**- / 80 pts**

**Autograder Score**
**0.0 / 80.0**

**Failed Tests**

1.1 Add and Subtract (0/3)
2.1 Absolute Value (0/7)
2.2 Multiply (0/10)
3.1 Fibonacci (0/12)
3.2 Array Largest (0/16)
4.0 Load/Compile Implementation

**Passed Tests**

Submitted on Time
Your Test Cases (0/0)