8-bit opcode	24-bit address

Accumulator.load

00000001	24-bit address

Accumulator.store

00000010	24-bit address

Accumulator.add

00000011	24-bit address

Accumulator.multiply

00000100	24-bit address

Accumulator.run

00000101	24-bit address

Accumulator.end

00000110	24-bit address

Stack.push

00000001	24-bit address

Stack.pop

00000010	24-bit address

Stack.add

00000011	24-bit address

Stack.mul

00000100	24-bit address

Stack.end

00000101	24-bit address

```
# Accumulator-based quadratic_eval.s (in binary) (659 Bytes)

.data
000000000000000010000001: 0011 #X: 3
000000000000000010000010: 0111 #A: 7
000000000000000010000011: 0110 #B: 6
000000000000000010000100: 0001 #C: 1

.text
00000001 000000000000000010000001 # LOAD X
00000100 000000000000000010000001 # MUL  X
00000100 000000000000000010000010 # MUL  A
00000010 000000000000000010000010 # STOR A

00000001 000000000000000010000001 # LOAD X
00000100 000000000000000010000011 # MUL  B
00000010 000000000000000010000011 # STOR B

00000001 000000000000000010000100 # LOAD C
00000011 000000000000000010000011 # ADD  B
00000011 000000000000000010000010 # ADD  A

00000110 # END
```

```
# Stack-based quadratic_eval.s (in binary) (556 Bytes)
.data
000000000000000010000001: 0011 #X: 3
000000000000000010000010: 0111 #A: 7
000000000000000010000011: 0110 #B: 6
000000000000000010000100: 0001 #C: 1

.text
00000001 000000000000000010000001 # PUSH X
00000001 000000000000000010000001 # PUSH X
00000001 000000000000000010000010 # PUSH A
00000100 # MUL
00000100 # MUL

00000001 000000000000000010000001 # PUSH X
00000001 000000000000000010000011 # PUSH B
00000100 # MUL

00000001 000000000000000010000100 # PUSH C

00000011 # ADD
00000011 # ADD

00000101 # END
```

```
#  MIPS quadratic_eval.s (in ASCII) (911 Bytes)
# Accumulator-based quadratic_eval.s (in hex)

.data
0x81: 0x3 #X: 3
0x82: 0x7 #A: 7
0x83: 0x6 #B: 6
0x84: 0x1 #C: 1

.text
0x1 0x81 # LOAD X
0x4 0x81 # MUL  X
0x4 0x82 # MUL  A
0x2 0x82 # STOR A

0x1 0x81 # LOAD X
0x4 0x83 # MUL  B
0x2 0x83 # STOR B

0x1 0x84 # LOAD C
0x3 0x83 # ADD  B
0x3 0x82 # ADD  A

0x6 # END
```

```
# Stack-based quadratic_eval.s (in hex)
.data
0x81: 0x3 #X: 3
0x82: 0x7 #A: 7
0x83: 0x6 #B: 6
0x84: 0x1 #C: 1

.text
0x1 0x81 # PUSH X
0x1 0x81 # PUSH X
0x1 0x82 # PUSH A
0x4 # MUL
0x4 # MUL

0x1 0x81 # PUSH X
0x1 0x83 # PUSH B
0x4 # MUL

0x1 0x84 # PUSH C

0x3 # ADD
0x3 # ADD

0x5 # END
```