CPE 333 Lab 0: Matrix Multiplication

Contributions:

- Wyatt Tack: Wrote multiplication and loop algorithms
- Justin Rosu: Verification
- Brayden Daly: Verification
- Tyler Hamilton: Helped with algorithm for matrix cell addressing

Multiplication verification:

```
ARRAY A:
     0, 3, 2, 0, 3, 1, 0, 3, 2, 3,
     2, 0, 3, 3, 1, 2, 3, 0, 0, 1,
     1, 1, 2, 3, 1, 2, 3, 1, 1, 3,
     2, 2, 0, 1, 3, 2, 2, 2, 0, 0,
     1, 0, 1, 3, 3, 0, 3, 3, 3, 3,
     0, 3, 2, 1, 2, 2, 0, 0, 3, 0,
     1, 1, 0, 3, 3, 1, 2, 3, 3, 0,
     1, 2, 1, 0, 1, 2, 2, 1, 0, 3,
     1, 0, 2, 2, 1, 1, 1, 1, 1, 1,
     2, 0, 3, 1, 1, 2, 2, 3, 3, 1
ARRAY B:
     1, 1, 0, 3, 1, 2, 0, 0, 0, 0,
     0, 2, 1, 2, 3, 0, 0, 3, 3, 2,
     2, 1, 2, 3, 3, 0, 2, 2, 1, 1,
     2, 2, 0, 2, 2, 1, 2, 3, 2, 2,
     3, 3, 2, 2, 1, 1, 1, 1, 2, 1,
     2, 2, 3, 3, 3, 0, 0, 3, 2, 3,
     2, 3, 1, 2, 1, 1, 2, 2, 0, 1,
     0, 3, 2, 1, 1, 1, 2, 0, 1, 2,
     2, 0, 2, 1, 3, 3, 2, 3, 2, 0,
     3, 1, 3, 3, 2, 0, 1, 0, 1, 1
```

Online Verification:

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}
1	28	31	35	35	36	12	20	25	29	23
2	30	28	20	38	29	11	20	28	16	20
3	35	33	29	42	36	13	24	32	23	25
4	21	33	20	30	23	12	13	22	20	21
5	39	38	32	39	34	23	32	29	25	22
6	22	20	23	27	34	12	14	33	27	18
7	28	35	24	30	30	22	25	31	26	22
8	23	25	25	32	26	6	12	19	17	19
9	21	19	17	25	22	10	16	19	14	14
10	30	30	31	38	36	20	26	29	21	21

Rars verification (cell 1,1 at 25376, in order)

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+12)	Value (+16)	Value (+20)	Value (+24)	Value (+28)
25344	3	3	2	0	1	0	1	1
25376	28	31	35	35	36	12	20	25
25408	29	23	30	28	20	38	29	11
25440	20	28	16	20	35	33	29	42
25472	36	13	24	32	23	25	21	33
25504	20	30	23	12	13	22	20	21
25536	39	38	32	39	34	23	32	29
25568	25	22	22	20	23	27	34	12
25600	14	33	27	18	28	35	24	30
25632	30	22	25	31	26	22	23	25
25664	25	32	26	6	12	19	17	19
25696	21	19	17	25	22	10	16	19
25728	14	14	30	30	31	38	36	20
25760	26	29	21	21	0	0	0	0
25792	0	0	0	0	0	0	0	0

Code:

```
.data
  arra:       #1, 2, 2,
              #2, 2, 1,
              #1, 1, 1,
        0, 3, 2, 0, 3, 1, 0, 3, 2, 3,
        2, 0, 3, 3, 1, 2, 3, 0, 0, 1,
        1, 1, 2, 3, 1, 2, 3, 1, 1, 3,
        2, 2, 0, 1, 3, 2, 2, 2, 0, 0,
        1, 0, 1, 3, 3, 0, 3, 3, 3, 3,
        0, 3, 2, 1, 2, 2, 0, 0, 3, 0,
        1, 1, 0, 3, 3, 1, 2, 3, 3, 0,
        1, 2, 1, 0, 1, 2, 2, 1, 0, 3,
        1, 0, 2, 2, 1, 1, 1, 1, 1, 1,
        2, 0, 3, 1, 1, 2, 2, 3, 3, 1

  arrb:       #3, 2, 1,
              #3, 2, 1,
              #1, 2, 3,
        1, 1, 0, 3, 1, 2, 0, 0, 0, 0,
        0, 2, 1, 2, 3, 0, 0, 3, 3, 2,
        2, 1, 2, 3, 3, 0, 2, 2, 1, 1,
        2, 2, 0, 2, 2, 1, 2, 3, 2, 2,
        3, 3, 2, 2, 1, 1, 1, 1, 2, 1,
        2, 2, 3, 3, 3, 0, 0, 3, 2, 3,
        2, 3, 1, 2, 1, 1, 2, 2, 0, 1,
        0, 3, 2, 1, 1, 1, 2, 0, 1, 2,
        2, 0, 2, 1, 3, 3, 2, 3, 2, 0,
        3, 1, 3, 3, 2, 0, 1, 0, 1, 1

  arrc:       #1, 1, 1,
              #1, 1, 1,
              #1, 1, 1,
.text
li sp, 0x10000      #Stack Pointer
la a1, arra  #Array memory pointers
la a2, arrb
la a0, arrc
li a3, 10    #square matrix size
call matrixmult     #one function
end: j end

#----- Matrix Multiplication:
-------------------------------------------------------------
# Multiply square matrices at memory a1 and a2 with length a3, for product at memory
a0
#using t0,1,2,3,4,5,6, s0,1,2, a4,5,6, ra
matrixmult:
       addi sp, sp, -56
       sw t0, 52(sp)
       sw t1, 48(sp)
       sw t2, 44(sp)
       sw t3, 40(sp)
```

```
        sw t4, 36(sp)
        sw t5, 32(sp)
        sw t6, 28(sp)
        sw s0, 24(sp)
        sw s1, 20(sp)
        sw s2, 16(sp)
        sw a4, 12(sp)
        sw a5, 8(sp)
        sw a6, 4(sp)
        sw ra, (sp)

# start nested for loops
        add t0, x0, x0 #i=0
rloop: bge t0, a3, rfin #loop for all a rows
        add t6, x0, x0 #j=0
cloop: bge t6, a3, cfin #loop for all b columns

#loop for adding products for each resultant cell
        add s0, x0, x0 #reset result cell
        add t2, x0, x0 #k=0
resloop:bge t2, a3, resfin #loop for all b row/a col
# use of temporaries for adding up memory locaton for specific a and b element:
        #   t3        t4     t5
        #addr a = a1 + t0*a3*4 + t2*4
        #lw s1(t3)    <-- s1 = a
        mv a5, t0
        mv a6, a3
        call multiply
        mv t4, a4
        slli t4, t4, 2
        slli t5, t2, 2
        add t3, a1, t4
        add t3, t3, t5
        lw s1, (t3)
        #addr b = a2 + t2*a3*4 + t6*4
        #lw s2(t3)    <-- s2 = b
        mv a5, t2
        mv a6, a3
        call multiply
        mv t4, a4
        slli t4, t4, 2
        slli t5, t6, 2
        add t3, a2, t4
        add t3, t3, t5
        lw s2, (t3)
        # multiply found elements and add to running total for cell
        mv a5 s1
        mv a6 s2
        call multiply
        add s0, s0, a4      #s0 = s1*s2
        addi t2, t2, 1      #k++
        j resloop
resfin:
        #addr c = a0 + t0*a3*4 + t6*4
        #sw s0 (t3)   <-- s0 = result in t3 location of c
```

```
        mv a5, t0
        mv a6, a3
        call multiply
        mv t4, a4
        slli t4, t4, 2
        slli t5, t6, 2
        add t3, a0, t4
        add t3, t3, t5
        sw s0, (t3)
        addi t6, t6, 1      #j++
        j cloop
cfin:   addi t0, t0, 1      #i++
        j rloop
rfin:
        lw ra, (sp)
        lw a6, 4(sp)
        lw a5, 8(sp)
        lw a4, 12(sp)
        lw s2, 16(sp)
        lw s1, 20(sp)
        lw s0, 24(sp)
        lw t6, 28(sp)
        lw t5, 32(sp)
        lw t4, 36(sp)
        lw t3, 40(sp)
        lw t2, 44(sp)
        lw t1, 48(sp)
        lw t0, 52(sp)
        addi sp, sp, 56
        ret
#----- multiply a4 = a5 * a6 -----
multiply:
        addi sp, sp, -4
        sw t0, (sp)
        add t0,x0,x0      #t0 = x
        add a4,x0,x0      #a0=0
muloop:      bge t0, a5, muend #for i>a1
        add a4, a4, a6        #a0+=a2
        addi t0, t0, 1        #i++
        j muloop
muend:  lw t0, (sp)
        addi sp, sp, 4
        ret
```