```
#Source: https://www.cs.drexel.edu/~jjohnson/fa00/cs570/programs/mips/mmult.asm

# void mmult(double x[][], double y[][], double z[][], int n)
# // Inputs:  x,y,z are n X n matrices.
# // Side-effect:  z is modified, so that z = x * y
#      {
#        int i,j;
#        for (i=0; i != n; i++)
#           for (j=0; j != n; j++)
#               z[i][j] = 0.0;
#              for (k=0; k != n; k++)
#                   z[i][j] = z[i][j] + x[i][k] * y[k][j];
#      }

       .text
       .globl main
main:

       sub    $sp,$sp,4
       sw     $ra,0($sp)
       
       la     $a0,A
       la     $a1,B
       la     $a2,C
       li     $a3,2
       jal    mmult
       li     $s0,1
       li     $v0,4
       la     $a0,str
       syscall
       mul    $t2,$a3,$a3
       li     $s0,0
loop:  sll    $t0,$s0,3     # 8*i
       add    $t1,$t0,$a2   #address of C[i]
       l.d    $f12,0($t1)
       li     $v0,3
       syscall                     # print C[i]
       li     $v0,4
       la     $a0,newline
       syscall
       add    $s0,$s0,1
       bne    $s0,$t2,loop
```

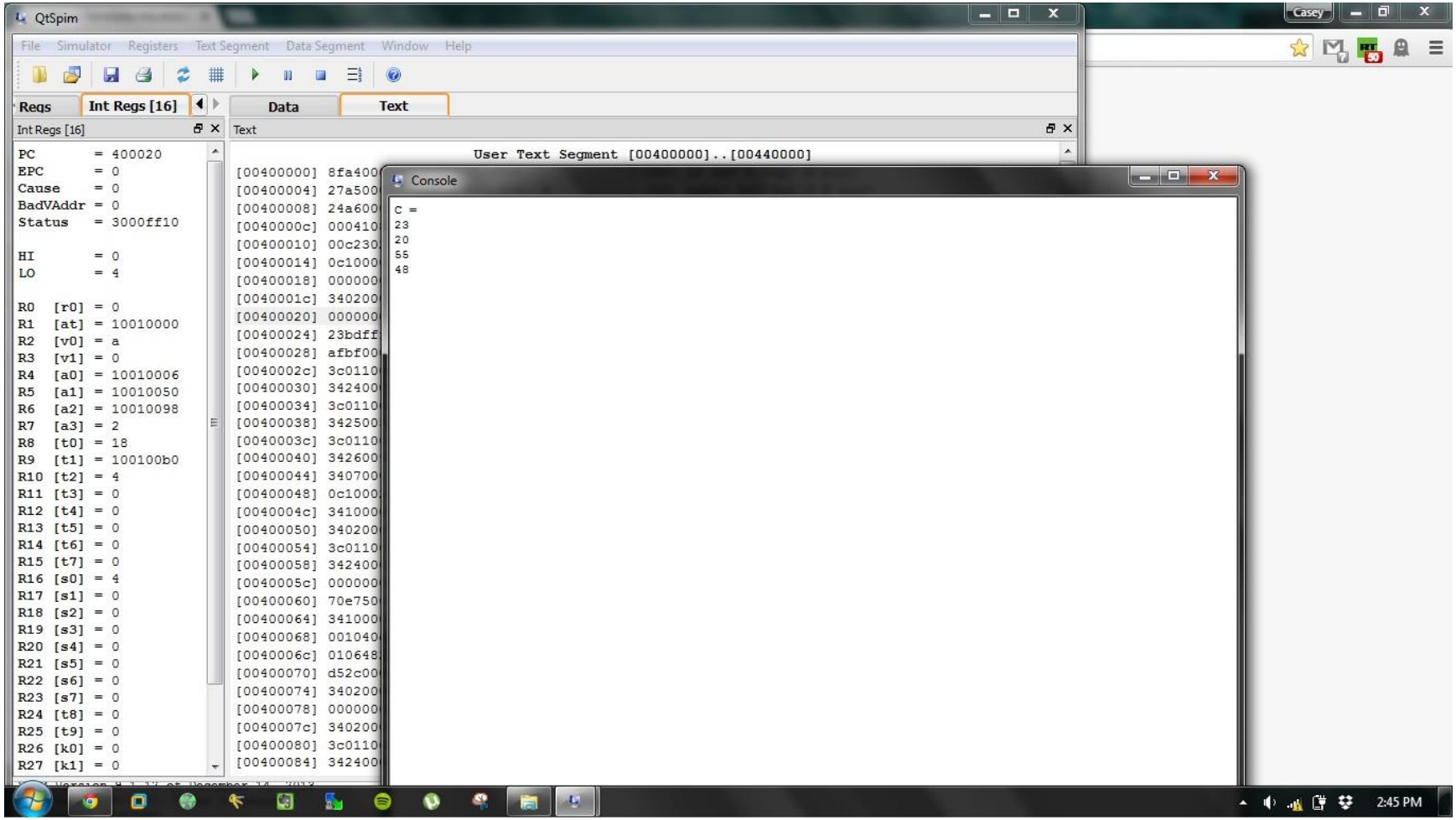
```
        lw      $ra,0($sp)
        add     $sp,$sp,4
        jr      $ra
```

```
        .data
str:    .asciiz         "C = \n"
newline:
        .asciiz         "\n"
        .align 2
A:      .double 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0
B:      .double         9.0, 8.0, 7.0, 6.0, 5.0, 4.0, 3.0, 2.0, 1.0
C:      .space 72       # 9 * 8 bytes
```

```
# matrix multiplication.
# leaf procedure
# inputs in $a0 = x, $a1 = y, $a2 = z, $a3 = n
# assume that n > 0
# temporaries used: $s0,$s1,$s2 $t0,$t1,$t2
# $f4, $f6, $f8, $f10 as double precision fp registers
# constants used:       zero
        .text
mmult:
        sub     $sp,$sp,12              # push two words on the stack
        sw      $s0,0($sp)
        sw      $s1,4($sp)
        sw      $s2,8($sp)

        li      $s0,0                   # i = 0
L1:     li      $s1,0                   # j = 0
L2:     li      $s2,0                   # k = 0
        mtc1    $zero,$f10              # z[i][j] = 0.0
        mtc1    $zero,$f11
inner:
        mul     $t0,$s0,$a3             # load x[i][k]
        add     $t1,$t0,$s2             # i*n+k
        sll     $t1,$t1,3               # 8*(i*n+k)
```

```
        add    $t2,$a0,$t1          # address of x[i][k]
        l.d    $f4,0($t2)           # load x[i][k]
        mul    $t0,$s2,$a3          # load y[k][j]
        add    $t1,$t0,$s1          # k*n+j
        sll    $t1,$t1,3            # 8*(k*n+j)
        add    $t2,$t1,$a1          # address of y[k][j]
        l.d    $f6,0($t2)           # load y[k][j]
        mul.d  $f8,$f4,$f6          # x[i][k] * y[k][j]
        add.d  $f10,$f10,$f8 # z[i][j] = z[i][j] + x[i][k] * y[k][j]
        add    $s2,$s2,1            # k++ and test inner loop condition
        bne    $s2,$a3,inner #
        mul    $t0,$s0,$a3          # store z[i][j]
        add    $t1,$t0,$s1          # i*n+j
        sll    $t1,$t1,3            # 8*(i*n+j)
        add    $t2,$t1,$a2          # address of z[i][j]
        s.d    $f10,0($t2)          # store z[i][j]
        add    $s1,$s1,1            # j++ and test loop condition
        bne    $s1,$a3,L2
        add    $s0,$s0,1            # i++ and test loop condition
        bne    $s0,$a3,L1
exit:   lw     $s0,0($sp)           # restore registers $s0,$s1
        lw     $s1,4($sp)
        lw     $s2,8($sp)
        add    $sp,$sp,12           # pop stack
        jr     $ra
```


QtSpim
File Simulator Registers Text Segment Data Segment Window Help
Regs
Int Regs [16]
Data
Text
Int Regs [16]
PC = 400020
EPC = 0
Cause = 0
BadVAddr = 0
Status = 3000ff10
HI = 0
LO = 4
R0 [r0] = 0
R1 [at] = 10010000
R2 [v0] = a
R3 [v1] = 0
R4 [a0] = 10010006
R5 [a1] = 10010050
R6 [a2] = 10010098
R7 [a3] = 2
R8 [t0] = 18
R9 [t1] = 100100b0
R10 [t2] = 4
R11 [t3] = 0
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 4
R17 [s1] = 0
R18 [s2] = 0
R19 [s3] = 0
R20 [s4] = 0
R21 [s5] = 0
R22 [s6] = 0
R23 [s7] = 0
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
Text
User Text Segment [00400000]..[00440000]
[00400000] 8fa400
[00400004] 27a500
[00400008] 24a600
[0040000c] 000410
[00400010] 00c230
[00400014] 0c1000
[00400018] 000000
[0040001c] 340200
[00400020] 000000
[00400024] 23bdff
[00400028] afbf00
[0040002c] 3c0110
[00400030] 342400
[00400034] 3c0110
[00400038] 342500
[0040003c] 3c0110
[00400040] 342600
[00400044] 340700
[00400048] 0c1000
[0040004c] 341000
[00400050] 340200
[00400054] 3c0110
[00400058] 342400
[0040005c] 000000
[00400060] 70e750
[00400064] 341000
[00400068] 001040
[0040006c] 010648
[00400070] d52c00
[00400074] 340200
[00400078] 000000
[0040007c] 340200
[00400080] 3c0110
[00400084] 342400
Console
C =
23
20
55
48
Casey
2:45 PM