```
 1  ; FindMax(Square, N, M)
 2  ; Input Parameters
 3  ;    $a0: Square is the address of first element of 2D matrix
 4  ;    $a1: N is the number of rows in Square
 5  ;    $a2: M is the number of columns in Square
 6  ; Return Value:
 7  ;    $v0: value of maximum element in Square
 8  ;
 9  0x1FFF FFB0 FindMax:      li      $v0, -1              # max <-- -1
10  0x1FFF FFB4               move    $t0, $zero           # i <-- 0
11  0x1FFF FFB8 NextRow:      slt     $t7, $a1, $t0        # if N<i then $t7 <-- 1
12  0x1FFF FFBC               bne     $t7, $zero, Return   # if i>=N Return
13  0x1FFF FFC0               move    $t5, $a0             # p <-- Square
14  0x1FFF FFC4               move    $t1, $zero           # j <-- 0
15  0x1FFF FFC8 NextColumn:   slt     $t7, $a2, $t1        # if M<j then $t7 <-- 1
16  0x1FFF FFCC               bne     $t7, $zero, RowDone  # if j>=M RowDone
17  0x1FFF FFD0               mul     $t3, $t0, $a1        # $t3 <-- i*N
18  0x1FFF FFD4               add     $t4, $t3, $t1        # $t4 <-- i*N+j
19  0x1FFF FFD8               sll     $t5, $t4, 2          # $t5 <-- 4*(i*N+j)
20  0x1FFF FFDC               lw      $t6, 0($t5)          # $t6 <-- Square[i][j]
21  0x1FFF FFE0               slt     $t7, $v0, $t6        # if(max < Square[i][j]) then $t7 <-- 1
22  0x1FFF FFE4               beq     $t7, $zero NoChange
23  0x1FFF FFE8               move    $v0, $t6             # max <-- Square[i][j]
24  0x1FFF FFEC NoChange:     addi    $t1, $t1, 1          # j <-- j+1
25  0x1FFF FFF0               j NextColumn
26  0x1FFF FFF4 RowDone:      addi    $t0, $t0, 1          # i <-- i+1
27  0x1FFF FFF8               j NextRow                    # if i != N goto NextRow
28  0x1FFF FFFC Return:       jr      $ra
```

Figure 1: MIPS Assembly code for FindMax procedure.

Question 1 (10 points):   The code for the FindMax procedure shown in Figure 1 is not
optimized and there are several improvements that could be made to improve this code.

a. (**6 points**) What changes would you do to the code to improve its performance. You do not
need to write assembly code — although you may if you wish. You can simply explain how
you would transform the code to make it more efficient.

b. (**4 points**) Given the same invocation of FindMax with N = 10000 and M = 5000 that we
studied in the previous question, would your changes increase or decrease the CPI? Explain
your reasoning.