APPENDIX I: MIPS TEST CODE for ASSEMBLER BATCH MODE

```
# The first instruction address in the program is at memory
# location 0x00400000
.text
main:     # label for the first instruction location
          la $a0, array_base  # array base
          la $a1, array_size  # array size
          jal BubbleSort
exit:     j exit

# Subproram: Bubble Sort
# Purpose: Sort data using a Bubble Sort algorithm
# Input Params: $a0 - array
# $a1 - array size
# Register conventions:
# $s0 - array base
# $s1 - array size
# $s2 - outer loop counter
# $s3 - inner loop counter
BubbleSort:
          addi $sp, $sp, -20  # save stack information
          sw $ra, 0($sp)
          sw $s0, 4($sp) # need to keep & restore $s registers
          sw $s1, 8($sp)
          sw $s2, 12($sp)
          sw $s3, 16($sp)
          move $s0, $a0
          lw $s1, 0($a1)
          addi $s2, $zero, 0  # outer loop counter
 OuterLoop:
          addi $t1, $s1, -1
          slt $t0, $s2, $t1
          beq $t0, $zero, EndOuterLoop
          addi $s3, $zero, 0  # inner loop counter
 InnerLoop:
          addi $t1, $s1, -1
          sub $t1, $t1, $s2
          slt $t0, $s3, $t1
          beq $t0, $zero, EndInnerLoop
          sll $t4, $s3, 2     # load data[j] - offset 4 bytes
          add $t5, $s0, $t4
          lw $t2, 0($t5)
          addi $t6, $t5, 4    # load data[j+1]
          lw $t3, 0($t6)
          sgt $t0, $t2, $t3
          beq $t0, $zero, NotGreater
          move $a0, $s0
          move $a1, $s3
          addi $t0, $s3, 1
          move $a2, $t0
          jal Swap         # t5 is &data[j], t6 is &data[j=1]

 NotGreater:
          addi $s3, $s3, 1
          j InnerLoop
 EndInnerLoop:
          addi $s2, $s2, 1
          j OuterLoop
 EndOuterLoop:
          lw $ra, 0($sp)    # restore stack information
          lw $s0, 4($sp)
          lw $s1, 8($sp)
          lw $s2, 12($sp)
          lw $s3, 16($sp)
          addi $sp, $sp, 20
          jr $ra
```

```
# Subprogram: swap
# Purpose: to swap values in an array of integers
# Input parameters: $a0 - the array containing elements to
swap
# $a1 - index of element 1
# $a2 - index of elelemnt 2
# Side Effects: Array is changed to swap element 1 and 2
Swap:
          sll $t0, $a1, 2   # calculate address of element 1
          add $t0, $a0, $t0
          sll $t1, $a2, 2   # calculate address of element 2
          add $t1, $a0, $t1
          lw $t2, 0($t0)    # swap elements
          lw $t3, 0($t1)
          sw $t2, 0($t1)
          sw $t3, 0($t0)
          jr $ra


# The first data address in the program is at memory location
# 0x10010000
.data
array_size:   # label for next data memory location
          .word 10
array_base:   # label for next data memory location
          .word 24
          .word 53
          .word 28
          .word 12
          .word 75
          .word 49
          .word 61
          .word 17
          .word 36
          .word 83
```