**8.1**

.data

prompt: .asciiz "Enter an integer to be converted to hexadecimal: "

pre: .asciiz "0x"

digit\_to\_hex: .byte '0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F'

.text

.globl main

main: la $a0, prompt # load address of prompt1 for syscall

li $v0, 4 # specify Print String service

syscall # print the prompt string

li $v0, 5 # specify Read Integer service

syscall # Read the number. After this instruction, the number read is in $v0.

move $a0, $v0 # move the read int to the argument

jal print\_hex # Call print\_hex

li $v0, 10 # Setup Exit Syscall

syscall # Execute syscall

print\_hex: move $t0, $a0

li $t1, 32 # value to srl

la $t2, digit\_to\_hex # ptr to lookup table

# Prints 0x

la $a0, pre # load address of pre for syscall

li $v0, 4 # specify Print String service

syscall

# Prints the rest of the chars

loop: addi $t1, $t1, -4 # decrement srl value by 4

srlv $t3, $t0, $t1 # first time this SRLVs 28, then 24, then 20, 16, 12, 8, 4,0

andi $t4, $t3, 0xF # extract digit. This ANDs the integer byte with all ones, in order to copy it to $t4

add $t4, $t4, $t2 # Adds output to the address in the map

lbu $t4, 0($t4)

add $a0, $t4, $zero # load desired char into $a0

li $v0, 11 # specify Print String service

syscall

bne $t1, $zero,loop # has the loop looped while srl has been 0? If not, then loop again. Otherwise, return.

jr $ra

**8.2**

.data

test: .asciiz "ABCDEGFHIJKL"

.text

.globl main

main: la $a0, test # load address of the original test for syscall

li $v0, 4 # specify Print String service

syscall # print the prompt string

li $v0, 11 # Setup print char syscall

addi $a0, $zero, 10 # Load newline ascii char

syscall # execute syscall

la $a0, test # move the test char pointer to $a0

addi $a1, $zero, 5 # set first index arg to be 5

addi $a2, $zero, 6 # Set second index arg to be 6

jal swap\_mem # Call swap\_mem

la $a0, test # load address of the swapped test for syscall

li $v0, 4 # specify Print String service

syscall # print the prompt string

li $v0, 10 # Setup Exit Syscall

syscall # Execute syscall

swap\_mem: move $t0, $a0 # char pointer

move $t1, $a1 # Index 1

move $t2, $a2 # Index 2

add $t3, $t0, $t1 # Address of char at index 1

add $t4, $t0, $t2 # Address of char at index 2

lbu $t5, 0($t3) # Char at index 1

lbu $t6, 0($t4) # Char at index 2

sb $t5, 0($t4)

sb $t6, 0($t3)

jr $ra

**8.3**

.data

test: .asciiz "123456789"

.text

.globl main

main: la $a0, test # move the test char pointer to $a0

jal string\_length # Call swap\_mem

move $a0, $v0 # load returned length value into $a0

li $v0, 1 # specify Print int service

syscall # print the prompt string

li $v0, 10 # Setup Exit Syscall

syscall # Execute syscall

string\_length: move $t0, $a0 # char pointer

li $t1, 0 # Set i to 0

loop: add $t2, $t0, $t1 # Adds the char pointer to the index

lbu $t3, 0($t2) # gets the ascii value of char into $t3

beq $t3, $zero, end # jump to end if string hits null terminated value

addi $t1, $t1, 1

j loop

end: move $v0, $t1

jr $ra

**8.4**

.data

test: .asciiz "The quick brown fox jumps over the lazy dog."

#test: .asciiz "EDCBA"

.text

.globl main

main: la $a0, test # load address of the original test for syscall

li $v0, 4 # specify Print String service

syscall # print the prompt string

li $v0, 11 # Setup print char syscall

addi $a0, $zero, 10 # Load newline ascii char

syscall # execute syscall

la $a0, test # move the test char pointer to $a0

jal s\_sort # Call s\_sort

la $a0, test # load address of the sorted test for syscall

li $v0, 4 # specify Print String service

syscall # print the prompt string

li $v0, 10 # Setup Exit Syscall

syscall # Execute syscall

s\_sort: move $t0, $a0 # Char pointer is stored in $t0

# CALL STRING\_LENGTH

addi $sp, $sp, 8

sw $ra, 8($sp)

sw $t0, 4($sp)

jal string\_length

lw $t0, 4($sp)

lw $ra, 8($sp)

move $t1, $v0 # $t1 will be length, retrieved from string\_length function

# END CALL STRING\_LENGTH

addi $t2, $zero, 0 # Initialize i to 0, within $t2

for\_1: bge $t2, $t1, end\_for\_1 # If i is greater than or equal to length, then end

add $t3, $t2, $zero # set index\_of\_min to i, stored within $t3

add $t4, $zero, $t2 # Set j equal to i, stored within $t4

for\_2: bge $t4, $t1, end\_for\_2

add $t5, $t0, $t3 # Get's address of message[index\_of\_min]

lbu $t5, 0($t5) # get value at message[index\_of\_min]

add $t6, $t0, $t4 # Get's address of message[j]

lbu $t6, 0($t6) # get value at message[j]

ble $t5, $t6, end\_if

add $t3, $zero, $t4

end\_if:

addi $t4, $t4, 1

j for\_2

end\_for\_2:

# CALL SWAP

addi $sp, $sp, 32

sw $ra, 32($sp)

sw $t0, 28($sp)

sw $t1, 24($sp)

sw $t2, 20($sp)

sw $t3, 16($sp)

sw $t4, 12($sp)

sw $t5, 8($sp)

sw $t6, 4($sp)

move $a0, $t0

move $a1, $t2

move $a2, $t3

jal swap\_mem

lw $t6, 4($sp)

lw $t5, 8($sp)

lw $t4, 12($sp)

lw $t3, 16($sp)

lw $t2, 20($sp)

lw $t1, 24($sp)

lw $t0, 28($sp)

lw $ra, 32($sp)

# END CALL SWAP

addi $t2, $t2, 1

j for\_1

end\_for\_1: jr $ra

swap\_mem: move $t0, $a0 # char pointer

move $t1, $a1 # Index 1

move $t2, $a2 # Index 2

add $t3, $t0, $t1 # Address of char at index 1

add $t4, $t0, $t2 # Address of char at index 2

lb $t5, 0($t3) # Char at index 1

lb $t6, 0($t4) # Char at index 2

sb $t5, 0($t4)

sb $t6, 0($t3)

jr $ra

string\_length: move $t0, $a0 # char pointer

li $t1, 0 # Set i to 0

loop: add $t2, $t0, $t1 # Adds the char pointer to the index

lbu $t3, 0($t2) # gets the ascii value of char into $t3

beq $t3, $zero, end # jump to end if string hits null terminated value

addi $t1, $t1, 1

j loop

end: move $v0, $t1

jr $ra