Matthew Michael Sherlin
Dr. Augustine Samba
Computer Organization
December 3, 2020

Assembly Code File:

```
# Starter code for reversing the case of a 30 character input.
# Put in comments your name and date please.  You will be
# revising this code.
#
# Created by Dianne Foreback
# Students should modify this code
#
# Matthew Michael Sherlin
# Dr. Augustine Samba
# Computer Organization
# November 6, 2020
#
#
# This code displays the authors name (you must change
# outpAuth to display YourFirstName and YourLastName".
#
# The code then prompts the user for input
# stores the user input into memory "varStr"
# then displays the users input that is stored in"varStr"
#
# You will need to write code per the specs for
# procedures main, revCase and function findMin.
#
# revCase will to reverse the case of the characters
# in varStr.  You must use a loop to do this.  Another buffer
# varStrRev is created to hold the reversed case string.
#
# Please refer to the specs for this project, this is just starter code.
#
# In MARS, make certain in "Settings" to check
# "popup dialog for input syscalls 5,6,7,8,12"
#
        .data     # data segment
            .align 2   # align the next string on a word boundary
outpAuth:   .asciiz  "This is Matthew Sherlin presenting revCaseMin.\n"
outpPrompt: .asciiz  "Please enter 30 characters (upper/lower case mixed):\n"
            .align 2   #align next prompt on a word boundary
outpStr:    .asciiz  "You entered the string: "
        .align 2   # align users input on a word boundary
varStr:     .space 32  # will hold the user's input string thestring of 20 bytes
                # last two chars are \n\0  (a new line and null char)
```

```
                # If user enters 31 characters then clicks "enter" or hits the
                # enter key, the \n will not be inserted into the 21st element
                # (the actual users character is placed in 31st element).  the
                # 32nd element will hold the \0 character.
                # .byte 32 will also work instead of .space 32
        .align 2   # align next prompt on word boundary
outpStrRev: .asciiz   "Your string in reverse case is: "
        .align 2   # align the output on word boundary
varStrRev:  .space 32  # reserve 32 characters for the reverse case string
            .align 2   # align  on a word boundary
outpStrMin: .asciiz    "\nThe min ASCII character after reversal is: "
#
        .text     # code section begins
        .globl      main
main:
#
# system call to display the author of this code
#
          la $a0,outpAuth           # system call 4 for print string needs address of string in $a0
          li $v0,4            # system call 4 for print string needs 4 in $v0
          syscall

#
# system call to prompt user for input
#
          la $a0,outpPrompt       # system call 4 for print string needs address of string in $a0
          li $v0,4            # system call 4 for print string needs 4 in $v0
          syscall
#
# system call to store user input into string thestring
#
         li $v0,8             # system call 8 for read string needs its call number 8 in $v0
                              # get return values
         la $a0,varStr        # put the address of thestring buffer in $t0
         li $a1,32                 # maximum length of string to load, null char always at end
                                   # but note, the \n is also included providing total len < 22
     syscall
     #move $t0,$v0                 # save string to address in $t0; i.e. into "thestring"
#
# system call to display "You entered the string: "
#
          la $a0,outpStr  # system call 4 for print string needs address of string in $a0
          li $v0,4            # system call 4 for print string needs 4 in $v0
          syscall
#
# system call to display user input that is saved in "varStr" buffer
#
          la $a0,varStr     # system call 4 for print string needs address of string in $a0
```

```
        li $v0,4            # system call 4 for print string needs 4 in $v0
        syscall
#
# Your code to invoke revCase goes next
#

jal revCase

# Exit gracefully from main()
      li   $v0, 10      # system call for exit
      syscall          # close file

#####################################################################
# revCase() procedure can go next
#####################################################################
#  Write code to reverse the case of the string.  The base address of the
# string should be in $a0 and placed there by main().  main() should also place into
# $a1 the number of characters in the string.
# You will want to have a label that main() will use in its jal
# instruction to invoke revCase, perhaps revCase:
#
revCase:
addiu $sp,$sp,-4
sw $ra,0($sp)
li $t3,0
li $t2,97

loop:
beq $t3,31,print
lb $t1,($a0)
bgt $t2,$t1,Label1
subiu $t1,$t1,32
sb $t1,varStrRev($t3)
addiu $a0,$a0,1
addiu $t3,$t3,1
j loop

Label1:
addiu $t1,$t1,32
sb $t1,varStrRev($t3)
addiu $t3,$t3,1
addiu $a0,$a0,1
j loop

print:
#
# After reversing the string, you may print it with the following code.
# This is the system call to display "Your string in reverse case is: "
```

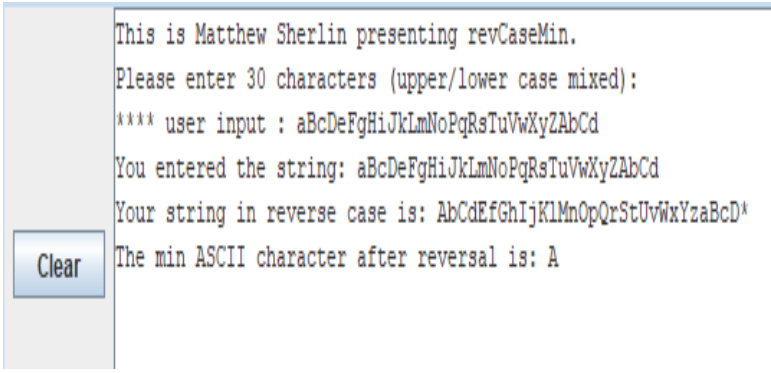
```
        la $a0,outpStrRev       # system call 4 for print string needs address of string in $a0
        li $v0,4            # system call 4 for print string needs 4 in $v0
        syscall
# system call to display the user input that is in reverse case saved in the varRevStr buffer
        la $a0,varStrRev        # system call 4 for print string needs address of string in $a0
        li $v0,4            # system call 4 for print string needs 4 in $v0
        syscall

#
# Your code to invoke findMin() can go next
jal findMin
# Your code to return to the caller main() can go next
jr $ra
#####################################################################
# findMin() function can go next
#####################################################################
#  Write code to find the minimum character in the string.  The base address of the
# string should be in $a0 and placed there by revCase.  revCase() should also place into
# $a1 the number of characters in the string.
#  You will want to have a label that revCase() will use in its jal
# instruction to invoke revCase, perhaps findMin:
#
#
findMin:
# write use a loop and find the minimum character
li $t3, 0
la $t2, varStrRev
lb $t1, 0($t2)
move $t9, $t1

loop2:
beq $t3, $a1, exit
lb $t1, 0($t2)
blt $t1, 64, Label2
blt  $t1, $t9, Label3
j Label2

Label3:
move $t9, $t1
j Label2

Label2:
addi $t2, $t2, 1
addi $t3, $t3, 1
j loop2

exit:
#
```

```
# system call to display "The min ASCII character after reversal is:  "
        la $a0,outpStrMin       # system call 4 for print string needs address of string in $a0
        li $v0,4            # system call 4 for print string needs 4 in $v0
        syscall

# write code for the system call to print the the minimum character

li $v0, 11
move $a0, $t9
syscall

# write code to return to the caller revCase() can go next
jr $ra
```

Project Implementation:

In order to get this program to work, I had to first read through and understand everything that was going on. It was difficult to follow everything but once I got the hang of it I began to work on it. First, I just wrote a basic jal to invoke the revCase function in the beginning. Next, was the writing of the function. This was difficult and gave me a lot of issues. I needed to use my resources for help but eventually I got it. In the beginning I initialize the registers, add space on the stack, and begin the loop. In the loop, I have a couple branch statements to cover some cases, and then the integer is switched and stored to the reversed string. After the revCase function, findMin is invoked. FindMin is much simpler. This function searches through and finds the lowest value and stores it into t9. It will loop through for 32 times and increment it each time through the loop. It stores the lowest value into t9 each iteration and once it hits 32, the loop is exited. Finally, the value in t9 is printed and then the code is returned and exited. There is an issue where the loop doesn't go back and end the program, however the code is still working.

Working Code Screen Print:



Transcription:

This is Matthew Sherlin presenting revCaseMin.
Please enter 30 characters (upper/lower case mixed):
**** user input : aBcDeFgHiJkLmNoPqRsTuVwXyZAbCd
You entered the string: aBcDeFgHiJkLmNoPqRsTuVwXyZAbCd
Your string in reverse case is: AbCdEfGhIjKlMnOpQrStUvWxYzaBcD*
The min ASCII character after reversal is: A

Conclusion:

To conclude, I learned and struggled A LOT during this lab. This was by far the toughest and much more complex than the others. I needed to use the internet a lot to do research and understand how everything works but in the end the program works for the most part. I had issues with the functions but they work now, however the program doesn't loop back and end the program gracefully. There is a loop at the end but I had trouble getting to loop to the end without adding something in the part of the code that I wasn't meant to add. However, in the end the general purpose of the code works, and I hope I can get partial credit for this because of that.