CS 224
Section No.: 2
Spring 2019
Lab 03
Berrak Taşkınsu / 21602054

CS 224 – Spring 2019 – Lab #2

(Relatively Big) MIPS Assembly Language Programming
Recursion, Floating Point Numbers, Linked Lists

Preliminary Design Report

Berrak Taşkınsu / 21602054

Section No.: 2

CS 224
Section No.: 2
Spring 2019
Lab 03
Berrak Taşkınsu / 21602054

Part 1. Preliminary Work / Preliminary Design Report

1. (10 points) recursiveDivision: Write a recursive MIPS subprogram that performs integer division of two positive numbers by successive subtractions. As a challenge you may want to implement it to return the quotient in $a0 and remainder in $a1. You may assume that inputs are OK, i.e. two positive numbers as expected. Likewise in the following programs also assume that input is OK.

```
      .text
main:
      li $v0, 5
      syscall
      move $a0, $v0
      li $v0, 5
      syscall
      move $a1, $v0
      jal recursiveDivision
      move $a0, $v0
      li $v0, 1
      syscall
      li $v0, 10
      syscall

recursiveDivision:
      addi  $sp, $sp, -12
      sw    $a0, 8($sp) # $a0:21
      sw    $a1, 4($sp) # $a1:5
      sw    $ra, 0($sp)
      bge   $a0, $a1, else
      addi  $sp, $sp, 12
      addi  $v0, $zero, 0
      jr    $ra
      else:
      sub   $a0, $a0, $a1
      jal recursiveDivision
      lw    $ra, 0($sp)
      lw    $a1, 4($sp)
      lw    $a0, 8($sp)
      addi  $sp, $sp, 12
      addi  $v0, $v0, 1
      jr    $ra


      .data
prom1:     .asciiz "Enter the number: "
prom2:     .asciiz "Enter divider: "
```

CS 224
Section No.: 2
Spring 2019
Lab 03
Berrak Taşkınsu / 21602054

2. (10 points) multiplyDigits: Write a recursive MIPS subprogram that finds the multiplication of the digits of a positive integer. For example for 127 it returns 14.

```
      .text
main:
      li $v0, 5
      syscall
      move $a0, $v0
      li $a1, 1
      jal multiplyDigits
      move $a0, $v0
      li $v0, 1
      syscall
      li $v0, 10
      syscall

multiplyDigits:
      addi  $sp, $sp, -12
      sw    $a0, 8($sp) # $a0:127 ( quotient )
      sw    $a1, 4($sp) # $a1:0 ( remainder )
      sw    $ra, 0($sp)
      li $t0, 10
      bgt   $a0, $zero, else
      addi  $sp, $sp, 12
      add   $v0, $zero, $a1
      jr    $ra
      else:
      rem $a1, $a0, $t0
      div $a0, $a0, $t0
      jal multiplyDigits
      lw    $ra, 0($sp)
      lw    $a1, 4($sp)
      lw    $a0, 8($sp)
      addi  $sp, $sp, 12
      mul   $v0, $v0, $a1
      jr    $ra


      .data
prom1:     .asciiz "Enter the number: "
prom2:     .asciiz "Enter divider: "
```

CS 224
Section No.: 2
Spring 2019
Lab 03
Berrak Taşkınsu / 21602054

3. (10 points) Delete_x: Study the linked list program provided and the linked list explanation provided below in Part 2. Delete all elements from the linked list with the value x: the pointer to the linked list is passed in $a0, and the integer value of the element to be deleted is given in $a1. Return the number of counted nodes in $v0. Return the list head pointer in $v1. Are you able to return the deleted node(s) back to the heap? If not include a comment in the program to explain why.

```
Delete_x:

   addi $sp, $sp, -12
   sw $s1, 8($sp)
   sw $s0, 4($sp)
   sw $ra, 0($sp)
   move $s1, $zero
   move $s0, $zero
   move $v0, $zero
   move $v1, $a0
   next:
        lw $s1, 4($v1)
        bne $s1, $a1, skip2
        lw $v1, 0($v1)
        b next
   skip2: # $v1: head pointer
   # We now know the first non-x element
   move $a0, $v1
   loop2:
        lw $s0, 0($a0)
        beq $s0, $zero, end
        lw $s1, 4($s0)
        beq $s1, $a1, delete2
        move $a0, $s0
        beq $s0, $zero, end
        b loop2
   delete2:

   lw $t0, 0($s0)
   sw $t0, 0($a0)
   beq $s0, $zero, end
   b loop2

   end:
   li $v0, 0
   lw $s1, 8($sp)
   lw $s0, 4($sp)
   lw $ra, 0($sp)
   addi $sp, $sp, 12
   jr $ra
```