**Course Name**: Computer Architecture Lab

**Course Number and Section**: **14:332:333:03**

**Experiment**: [Experiment # 5 – RISC-V functions and pointers]

**Lab Instructor**: Christos Mitropoulos

**Date Performed**: November 7 2018

**Date Submitted**: November 28 2018

**Submitted by**: Vancha Verma 173004061

**Course Name**: Computer Architecture Lab

**Course Number and Section**: **14:332:333:03**

**! Important: Please include this page in your report if the submission is a paper submission. For electronic submission (email or Sakai) please omit this page.**

--------------------------For Lab Instructor Use ONLY--------------------------

GRADE: ________________

COMMENTS:

## EXERCISE 1

**The task: read all of the commented lines under the map function in “megalistmanips.s” (before it returns with jr ra), and make sure that the lines do what the comments say.**

I made several changes in the code. First, i changed the line “add t1, s0, x0” to “lw t1, 0(s0)” since add would save the address where as lw would save the contents of s0 in t1. Since la is not a command, i looked at the comment to see the purpose of the time. To execute the comment, i changed la to lw. This caused issues when mul uses t1. Thus, instead of using the t1 register, i changed it to t3 in the code for mapLoop. Since the the offset is supposed to be index multiplied by 4, I added the line “slli t4, t0, 2″ and stored the new offset in t4. All the changes made were in mapLoop and can be seen below. I have also included the output of the fixed code.

```
mapLoop:
     lw t3, 0(s0)          #load the address of the array of current
node into t3
     lw   t2, 4(s0)        # load the size of the node's array into
t2

    slli t4, t0, 2         #the offset is supposed to be index
multiplied by 4
    add   t3, t3, t4       # offset the array address by the count
     lw   a0, 0(t3)        # load the value at that address into a0

     jalr  s1              # call the function on that value.

     sw   a0, 0(t3)        # store the returned value back into the
array
     addi  t0, t0, 1       # increment the count
     bne   t0, t2, mapLoop # repeat if we haven't reached the array
size yet

     lw   a0, 8(s0)        # load the address of the next node into
a0
     add a1, s1, x0        # put the address of the function back
into a1 to prepare for the recursion

     jal   map             # recurse
```

<u>Output:</u>

```
Lists before:
5 2 7 8 1
1 6 3 8 4
5 2 7 4 3
1 2 3 4 7
5 6 7 8 9

Lists after:
30 6 56 72 2
2 42 12 72 20
30 6 56 20 12
2 6 12 20 56
30 42 56 72 90
```

## EXERCISE 2

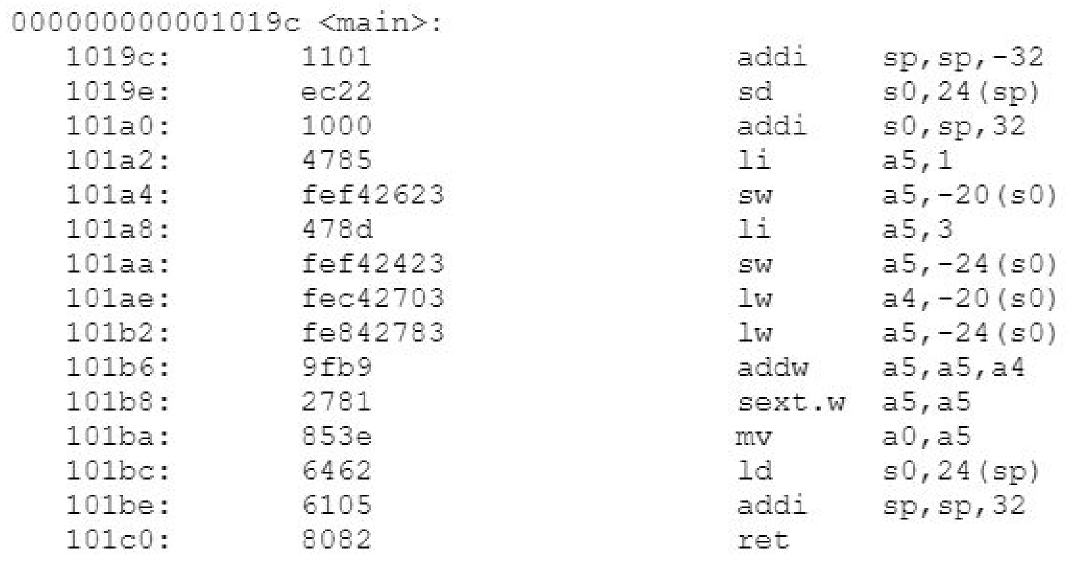
**Consider the discrete-valued function f defined on integers in the set {-3, -2, -1, 0, 1, 2, 3}. Here's the function definition:**
**f(-3) = 6**
**f(-2) = 61**
**f(-1) = 17**
**f(0) = -38**
**f(1) = 19**
**f(2) = 42**
**f(3) = 5**
**Your task is to implement it in RISC-V, with the condition that your code may NOT use any branch instructions!**

```
# YOUR CODE GOES HERE!
addi t1, a0, 3          # changes a0 values to be [0,1,2,3,4,5,6] and
stores it in t1
addi t2, x0, 4
mul t1, t2, t1          # multiplies the offset by 4
add t0, t0, t1          # adds the offset value to the array
lw a0, 0(t0)            # loads the value using the new offset to a0

jr   ra         # Always remember to jr ra after your function!
```

**Notice that there is an array of integers in the .data section of "discrete_fn.s" How can you use this to your advantage and complete this task?**
The .data section is used to see if the printed values match the answer.

**EXERCISE 3**
**Task: Open this file with an editor (e.g., nano dump.txt) and try to find the main function.**



```
000000000001019c <main>:
   1019c:       1101                            addi    sp,sp,-32
   1019e:       ec22                            sd      s0,24(sp)
   101a0:       1000                            addi    s0,sp,32
   101a2:       4785                            li      a5,1
   101a4:       fef42623                        sw      a5,-20(s0)
   101a8:       478d                            li      a5,3
   101aa:       fef42423                        sw      a5,-24(s0)
   101ae:       fec42703                        lw      a4,-20(s0)
   101b2:       fe842783                        lw      a5,-24(s0)
   101b6:       9fb9                            addw    a5,a5,a4
   101b8:       2781                            sext.w  a5,a5
   101ba:       853e                            mv      a0,a5
   101bc:       6462                            ld      s0,24(sp)
   101be:       6105                            addi    sp,sp,32
   101c0:       8082                            ret
```

.