FIT1008 – Intro to Computer Science
Tutorial 4

Semester 1, 2018

Objectives of this tutorial

- To understand the function calling and returning in MIPS.
- To be able to write simple MIPS functions.
- To understand memory maps.

Exercise 1

Consider the following uncommented MIPS code:

```
function:
        addi $sp, $sp, -8
        sw $ra, 4($sp)
        sw $fp, 0($sp)
        addi $fp, $sp, 0
        addi $sp, $sp, -4

        lw $t0, 8($fp)
        lw $t1, 12($fp)
        blt $t0, $t1, one

        lw $t0, 8($fp)
        sw $t0, -4($fp)
        j end

one:    lw $t0, 12($fp)
        sw $t0, -4($fp)

end:    lw $v0, -4($fp)
        addi $sp, $sp, 4
        lw $fp, 0($sp)
        lw $ra, 4($sp)
        addi $sp, $sp, 8
        jr $ra
```

(i) Comment the code.

(ii) What does this program do?

Exercise 2

Consider the following Python code:

```
def collatz(n):

    ## HERE

    if n % 2 == 0:
       return n/2

    return 3*n + 1

n = int(input("Enter integer: "))

while (n > 1):
   print(n)
   n = collatz(n)
```

(i) Draw a stack diagram at the time `## HERE` is found.

(ii) Translate the above program into MIPS. Try to make your translation as faithful as possible.

Exercise 3

Translate into MIPS the following function:

```
def odd_product(a_list):
    product = 1
    for x in a_list:
        if x%2 !=0:
            product=product*x
    return product
```

Exercise 4

(i) The function calling convention given in lectures typically has functions accessing their first parameter at `8($fp)`, the second at `12($fp)`, the third at `16($fp)`, and so on.

Is this **order** necessary? In other words, would it be possible to have the last parameter at `8($fp)` instead, and the second-last at `12($fp)`, and so on (provided that all functions are changed to agree with this new convention)?

(ii) Why is the memory at address `4($fp)`seldom accessed by a called function?