Ho Chi Minh City University of Technology

Faculty of Computer Science and Engineering
Course: Computer Architecture Lab (CO2008)

Lab 3

Advanced instructions

Ho Chi Minh City, October 23rd 2023

Contents

1 Introduction

- The main purpose of this session is to get familiar with floating-point instructions and read/write file.
- Students must submit their answers to the BKeL system no later than the last period of the lab section. Then, the instructor will evaluate all students' work during the lab section's final period. Please note that we will randomly choose ∼50% of the questions to mark.

2 Exercises

2.1 Exercise 1

Write a MIPS program that calculates and print either the volume or total surface area of a rectangular box, cube, cylinder, pyramid, prism, or sphere. The user is able to choose which metric, shape, and the related parameters to calculate. Note that the parameters can be floating-point numbers.

2.2 Exercise 2

Write a MIPS program to calculate the following integral:

$$f(x) = \int_v^u \frac{ax^4 + bx^3 + cx^2 + d}{e^2} \tag{1}$$

where u, v, a, b, c, d are floating-point numbers chosen by the user, and e is last digit of your student ID (for example, if your student ID is 1234567 then e is 7) For example, if you have the ID 1234567, and user inserted **a=1, b=2, c=3, d=4, u=5, and v=6** then the result should be: **-194.39**

2.3 Exercise 3

To allocate memory, please refer to the following syscall:

```
1    li $v0, 9 # system call code for dynamic allocation
2    li $a0, 24 # $a0 contains number of bytes to allocate
```

After the above system call, $v0 contains the first address in heap memory that is allocated. Then, accessing the allocated memory can be done by lw/sw, for example:

```
    # Trying to write to allocated space
    addi $t0, $zero, 2021
    sw $t0, 0($v0)
```

The followings are instructions used to access a file (open/close/read/write):

```
    # Sample MIPS program that writes to a new file.
    # by Kenneth Vollmar and Pete Sanderson
    .data
    fout: .asciiz "testout.txt" # filename for output
    msg1: .asciiz "Before read: "
    msg2: .asciiz "After read: "
    buffer_write: .asciiz "The quick brown fox jumps over the lazy dog.\n"
    buffer_read: .asciiz "----------\n"
    .text
    ##############################################################
    # Open (for writing) a file that does not exist
    li $v0, 13 # system call for open file
    la $a0, fout # output file name
    li $a1, 1 # Open for writing (flags are 0: read, 1: write)
    li $a2, 0 # mode is ignored
    syscall # open a file (file descriptor returned in $v0)
    move $s6, $v0 # save the file descriptor
    ##############################################################
    # Write to file just opened
    li $v0, 15 # system call for write to file
    move $a0, $s6 # file descriptor
    la $a1, buffer_write # address of buffer from which to write
    li $a2, 44 # hardcoded buffer length
    syscall # write to file
    ##############################################################
    # Close the file
    li $v0, 16 # system call for close file
    move $a0, $s6 # file descriptor to close
    syscall # close file
    ##############################################################
    ##############################################################
    # Open (for reading) a file
    li $v0, 13 # system call for open file
    la $a0, fout # input file name
    li $a1, 0 # Open for reading (flags are 0: read, 1: write)
    li $a2, 0 # mode is ignored
    syscall # open a file (file descriptor returned in $v0)
    move $s6, $v0 # save the file descriptor
    ##############################################################
    # Read from file
    li $v0, 14 # system call for read
```

```
42      move $a0, $s6 # file descriptor
43      la $a1, buffer_read # address of buffer read
44      li $a2, 44 # hardcoded buffer length
45      syscall # read file
```

Please do the followings:

1. Manually create a line of text in format <id>, <name>, <weight >, <height>, <medical history>in a text file by using a text editor (students define the structure of the file themselves). The content on each line should be student ID, name, height, and weight respectively.
2. Open the file to read the text you inserted.
3. Declare a string in the heap memory with dynamically allocated memory. The size of the string must be large enough to store the text in the text file.
4. Copy the line from the text file to the string in the memory.
5. Print the string to the terminal in the following format:
   (a) Student medical information
   (b) Name: <name>
   (c) ID: <ID>
   (d) Weight: <weight>
   (e) Height: <height>
   (f) Medical history: <medical history>