
Notes:

- 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 questions to mark.**
- To submit, please make a zip file that contains all sources and supplementary files (such as text files containing matrices) for these questions below, each question should be answered in one source file.

Question 1. Write a MIPS program that allows users to calculate areas of circle, triangle, rectangle or square. At first, users can choose the desired shape. Then, users can input different parameters of the selected shape. Finally, the program prints the area of the shape.

Question 2. Write a MIPS program that helps users calculate the following integral:

$$f(x) = \int_u^v (ax^2 + bx + c)\, dx \tag{1}$$

where u, v, a, b, and c are floating-point numbers given by users.

Question 3. To allocate memory, please refer to the following syscall

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

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)
```

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

---

1. Please create sentence/paragraph in a text file (students define the structure of the file themselves)
2. Open the file to read the sentence/paragraph;
3. Declare a string in the heap memory with dynamically allocated memory. The size of the string must be large enough to store the sentence/paragraph in the text file;
4. Copy the sentence/paragraph from the text file to the string in the memory;
5. Print the string to the terminal.

——————the end——————