Lab 1

```
	.data
Z: .word 0
value: .word 12

	.text
	.globl main

main:

	li $t2, 25 # Load immediate value (25)

	lw $t3, value # Load the word stored in value (see above)

	add $t4, $t2, $t3 # Add
	sub $t5, $t2, $t3 # Subtract

	sw $t5, Z #Store the answer in Z (declared at the above)



	li $v0, 4 # Print integer
	move $a0, $t4 # Move the value to be printed into $a0
	syscall # Print the value

	li $v0, 10 # Sets $v0 to "10" to select exit syscall
	syscall # Exit
```

Lab 2.1

```
	.data
prompt:	.asciiz	"\n\n Please Input a value for N = "
result: .asciiz	" The sum of the integers from 1 to N is : "
bye:	.asciiz	"\n\n Adios Amigo! Have a nice day. \n\n"

	.globl main

	.text
main:
	li	$v0, 4	#System call code for print_str
	la	$a0, prompt	#Load address of prompt into $a0
	syscall		#Print the prompt


	li	$v0, 5	#System call code for read_int
	syscall		#Read the integer into $v0


	blez	$v0, done # If ( v0 <= 0 ) go to done
	li	$t0, 0	# clear $t0 to zero


loop:	add	$t0, $t0, $v0 # sum of integers in register $t0

		addi	$v0, $v0, -1 # summing in reverse order
```

```
            bnez   $v0, loop # branch to loop if $v0 is != 0

zero:   li          $v0, 4 # system call code for print_str
            la           $a0, result # load address of result into $a0
            syscall                          # print the string

            li           $v0, 1       # system call code for print_int
            move   $a0, $t0     # a0 = $t0
            syscall

            b            main


done:  li           $v0, 4       # system call code for print_str
            la           $a0, bye     # load address of msg. into $a0
            syscall                          # print the string

            li           $v0, 10             # terminate program
            syscall                          # return control to system
system:
```

Lab 2.2

```
.data
array:  .word 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
result: .asciiz "The sum of number in array is "

.globl main
.text
main:
    li $t0, 0         # $t0 - loop counter
    li $t1, 0         # $t1 - sum

    # Print a newline
    li $v0, 4           # System call code for printing a string
    la $a0, result     # Load the address of the newline string
    syscall

    # Loop to sum the array elements
    loop:
        lw $t2, array($t0)    # Load array element at index $t0 into $t2

        add $t1, $t1, $t2     # Add the element to the sum

        addi $t0, $t0, 4      # Move to the next array element (each element is 4
bytes)

        # Check if the end of the array is reached
        blt $t0, 40, loop     # Branch to loop if $t0 < 40

    # Print the result
    move $a0, $t1       # Set $a0 to the sum
    li $v0, 1           # System call code for printing an integer
    syscall

    # Exit program
    li $v0, 10          # System call code for program exit
    syscall
```

Lab 3-0

```
       .data
array: .word -4, 5, 8, -1
msg1: .asciiz "\n The sum of the positive values = "
msg2: .asciiz "\n The sum of the negative values = "
       .globl main
       .text
main:

       li $v0, 4 # system call code for print_str
       la $a0, msg1 # load address of msg1. into $a0
       syscall # print the string

       la $a0, array # Initialize address Parameter
       li $a1, 4 # Initialize length Parameter
       jal sum # Call sum

       move $a0, $v0 # move value to be printed to $a0
       li $v0, 1 # system call code for print_int
       syscall # print sum of Pos:

       li $v0, 4 # system call code for print_str
       la $a0, msg2 # load address of msg2. into $a0
       syscall # print the string

       li $v0, 1 # system call code for print_int
       move $a0, $v1 # move value to be printed to $a0
       syscall # print sum of neg

       li $v0, 10 # terminate program run and
       syscall # return control to system

sum:   li $v0, 0
              li $v1, 0 # Initialize v0 and v1 to zero
loop:
              blez $a1, retzz # If (a1 <= 0) Branch to Return
              addi $a1, $a1, -1 # Decrement loop count
              lw $t0, 0($a0) # Get a value from the array
              addi $a0, $a0, 4 # Increment array pointer to next word
              bltz $t0, negg # If value is negative Branch to negg
              add $v0, $v0, $t0 # Add to the positive sum
              b loop # Branch around the next two instructions
negg:
              add $v1, $v1, $t0 # Add to the negative sum
              b loop # Branch to loop
retzz: jr $ra # Return
```

Lab 3-1

```
       .data
chico: .word 3, 5, 2, 7, 8, 9, 1, 4, 6, 10
store: .asciiz "The resulting sum that store in the end of array is : "
       .globl main
       .text
main:         la $t0, chico
                    li $t1, 0 # Sum the zero
                    li $t2, 10 # Length of elements to Sum

loop:         lw $t3, 0($t0) # Load the value in array
                    add $t1, $t1, $t3   # Add to sum
                    addi $t0, $t0 , 4 # Move to the next word in array
                    addi $t2, $t2 , -1 # Decrement
                    bnez $t2, loop

final:       addi $t0, $t0, -4    # Move back to the last element of array chico
             sw $t1, 0($t0)       # Store the sum in the last element of array
chico

             li $v0, 4             # Print string
             la $a0, store
             syscall

             li $v0, 1
             lw $a0, 0($t0)
             syscall
```

Lab 3-2

```
              .data
SRC:   .word 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
DEST:  .space 40
MSG:   .asciiz "Transfer to DESC Array : "
Spacebar: .asciiz "\n"

              .globl main
              .text
main:  li $t0, 10 # counter
              la $t1, SRC # Load base address of SRC
              la $t2, DEST # Load base address of DEST

loop:  lw $t3, 0($t1) # Load word from SRC
              sw $t3, 0($t2) # Store in DEST
              addi $t1, $t1, 4 # Move address in SRC
              addi $t2, $t2, 4 # Move address in DEST
              addi $t0, $t0, -1 # Decrement

              bnez $t0, loop

              li $t0, 10 # counter
              la $t2, DEST # initialize DEST

              li $v0, 4
              la $a0, MSG
              syscall

result_loop: li $v0, 1
                           lw $a0, 0($t2)
                           syscall
```

```
                        li $v0, 4
                        la $a0, Spacebar
                        syscall


                        addi $t2, $t2, 4 # Move to next element
                        addi $t0, $t0, -1 #Decrement
                        bnez $t0, result_loop
```

Lab 3-3

```
        .data
X:      .word  1, 5, 81, 41, 56, 59, 14, 77
N:   .word    8
max_number:   .asciiz       "The max number in array is : "
min_number: .asciiz "\nThe min number in array is : "
        .globl main
        .text

main:  la $t0, X
               lw $t1, N # counter

               jal MinMax

               li $v0, 4
               la $a0, max_number
               syscall

               li $v0, 1
               move $a0, $t2
               syscall

               li $v0, 4
               la $a0, min_number
               syscall

               li $v0, 1
               move $a0, $t3
               syscall

               li $v0, 10
               syscall


MinMax:        lw $t2, 0($t0) # max_value
               move $t3, $t2 # min_value


loop:  lw $t4, 0($t0) # Current number


               # Update Min
               ble $t4, $3, min_update


               # Update max_value
               bge $t4, $t2, max_update

final_check:         addi $t1, $t1, -1 # Decrement
                              addi $t0, $t0, 4 # Next word with 4 bytes
```

```
                                bnez $t1, loop
                                jr $ra

min_update:
       move $t3, $t4
       j final_check

max_update:
       move $t2, $t4
       j final_check
```

Lab 3-4

```
       .data
X: .word 3, 5, 1, 7, 9, 15, 17
input: .asciiz "Please enter your number : "
found_text: .asciiz "Find it at index : "
notfound_text: .asciiz "\nNot found in array."
       .globl main
       .text
main:         li $v0, 4 # System call code for print_str
                     la $a0, input # Load the address of input into $a0
                     syscall # Print the string

                     li     $v0, 5       #System call code for read_int
                     syscall                    #Read the integer into $v0

                     li $t3, 0     # Clear $t3 to zero
                     add $t3, $t3, $v0 # Add to $t3 to find
                     li $t0, 0     # Index
                     la $t1, X     # Load the address of X into $t1

loop:
                     # compare $t3 with the value at $t1
                     lw $t2, 0($t1) # Load the value at $t1 into $t2
                     beq $t3, $t2, found # If $t3 == $t2, jump to found

                     addi $t1, $t1, 4 # next address
                     addi $t0, $t0, 1 # next index
                     blt $t0, 7, loop # If $t0 == 7, jump to notfound
                     j notfound # Jump to loop

found:        li $v0, 4
                     la $a0, found_text
                     syscall
                     li $v0, 1
                     move $a0, $t0
                     syscall
                     j end

end:          li $v0, 10
                     syscall

notfound:     li $v0, 1
                     li $a0, -1
                     syscall

                     li $v0, 4
                     la $a0, notfound_text
```

```
syscall

li $v0, 10
syscall
```