

Department of Computer Science
Ashoka University
Computer Organization and Systems
Assignment 4

Name: Rushil Gupta and Dhruvan Gupta

Problem 1: Sum of Numbers from 1 to N

Pseudocode

```
Prompt user for a positive integer N (<1000)
Read N
Set sum = 0, i = 1
While i <= N:
    sum = sum + i
    i = i + 1
Print "The Sum (S) is equal to: " and sum
```

MIPS Code

```
.data
prompt:      .asciiz "Enter a positive integer (<1000): "
resultStr:   .asciiz "The Sum (S) is equal to: "

.text
main:
    # Print prompt message
    li    $v0, 4
    la    $a0, prompt
    syscall

    # Read integer input into register $t0 (N)
    li    $v0, 5 # We load into $v0 since it is a special register
    syscall
    move  $t0, $v0      # $t0 holds N

    # Initialize sum = 0 and counter i = 1
    li    $t1, 0        # $t1 will hold the sum
    li    $t2, 1        # $t2 is our loop counter i

sum_loop:
    bgt   $t2, $t0, sum_done # If i = $t2 > $t1 = N, then we are done with the loop
    add   $t1, $t1, $t2
    addi  $t2, $t2, 1
    j     sum_loop

sum_done:
    li    $v0, 4
    la    $a0, resultStr # Load the address of the result string
    syscall

    li    $v0, 1
    move  $a0, $t1 # Load the sum into $a0
    syscall
```

Output

```
Enter a positive integer (<1000): 5
The Sum (S) is equal to: 15
-- program is finished running (dropped off bottom) --
```

Problem 2: GCD by Repeated Subtraction

Pseudocode

```
Prompt user for two positive integers A and B
Read A and B
If A <= 0 or B <= 0, then print an error message and exit
While A != B:
    If A > B, then A = A - B
    Else, B = B - A
GCD is A # (or B since A = B)
Print "The GCD is: " and the result
```

MIPS Code

```
.data
promptA:    .asciiz "Enter the first positive integer: "
promptB:    .asciiz "Enter the second positive integer: "
error_msg:  .asciiz "Please enter a positive number"
gcdStr:     .asciiz "The GCD is: "

.text
main:
    # $v0 = 4 is for printing a string, here we ask the user for the first integer
    li    $v0, 4
    la    $a0, promptA
    syscall

    # $v0 = 5 is for reading an integer, here we read the first integer
    li    $v0, 5
    syscall
    move  $t0, $v0      # A

    blez  $t0, error    # If A <= 0, then we print an error message

    # Similar to the above, we ask the user for the second integer
    li    $v0, 4
    la    $a0, promptB
    syscall

    # Similar to the above, we read the second integer
    li    $v0, 5
    syscall
    move  $t1, $v0      # B

    blez  $t1, error    # If B <= 0, then we print an error message

gcd_loop:
    beq  $t0, $t1, gcd_done # If A = B, then we are done
    bgt  $t0, $t1, sub_A    # If A > B, then we subtract B from A and go top of loop
    sub  $t1, $t1, $t0      # If B > A, then we subtract A from B
    j    gcd_loop

sub_A:
    sub  $t0, $t0, $t1
    j    gcd_loop

error:
    # Print the string "Please enter a positive number"
    li    $v0, 4
    la    $a0, error_msg
    syscall
```

```

    # Exit the program
    li $v0, 10
    syscall

gcd_done:
    # Only here when A = B = gcd(A, B)

    # Print the string "The GCD is: "
    li $v0, 4
    la $a0, gcdStr
    syscall

    # Print the GCD
    li $v0, 1
    move $a0, $t0
    syscall

```

Output

```

Enter the first positive integer: 89
Enter the second positive integer: 13
The GCD is: 1
-- program is finished running (dropped off bottom) --

Enter the first positive integer: 105
Enter the second positive integer: 35
The GCD is: 35
-- program is finished running (dropped off bottom) --

Enter the first positive integer: 2
Enter the second positive integer: -3
Please enter a positive number
-- program is finished running --

```

Problem 3: N-th Fibonacci Number

Pseudocode

```
Prompt user for a positive integer N (>2)
Read N
If N < 2, print an error message and exit
Initialize f0 = 0, f1 = 1, and counter i = 3
While i <= N:
    f2 = f0 + f1
    Set f0 = f1 and f1 = f2
    Increment i by 1
Print "The Fibonacci number is: " and f1
```

MIPS Code

```
.data
promptFib: .asciiz "Enter a positive integer (>= 2) for Fibonacci: "
fibStr:     .asciiz "The Fibonacci number is: "
errorStr:   .asciiz "Please enter a number >= 2"

.text
main:
    li    $v0, 4
    la    $a0, promptFib
    syscall

    li    $v0, 5
    syscall
    move  $t0, $v0      # N

    blt  $t0, 2, fib_error # If N < 2, then we print an error message
    # Although the assignment says check for N < 0, we asked for N >= 2

    li    $t1, 0        # f0
    li    $t2, 1        # f1
    li    $t3, 3        # counter

fib_loop:
    bgt  $t3, $t0, fib_done # i > N => done
    add  $t4, $t1, $t2    # f2 = f0 + f1
    move $t1, $t2        # f0 = f1
    move $t2, $t4        # f1 = f2
    addi $t3, $t3, 1     # i = i + 1
    j    fib_loop

fib_error:
    li    $v0, 4
    la    $a0, errorStr
    syscall

    li    $v0, 10
    syscall

fib_done:
    li    $v0, 4
    la    $a0, fibStr
    syscall

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

Output

Enter a positive integer (≥ 2) for Fibonacci: -3

Please enter a number ≥ 2

-- program is finished running --

Enter a positive integer (≥ 2) for Fibonacci: 10

The Fibonacci number is: 34

-- program is finished running (dropped off bottom) --

Problem 4: 32-bit 2's Complement Hexadecimal Representation

Pseudocode

```
Prompt user for an integer
Read integer N
Print "The 32-bit hexadecimal representation is: 0x"
Output N in hexadecimal format using syscall 34
```

MIPS Code

```
.data
prompt:      .asciiz "Enter an integer N: "
result_msg:  .asciiz "The 2's compliment Hex of N is: 0x"
err_msg:     .asciiz "Invalid input\n"
hex_lookup:  .ascii  "0123456789ABCDEF"
buffer:      .space 32

.text
.globl main

main:
    li      $v0, 4          # syscall: print string
    la      $a0, prompt
    syscall

    # Read the input
    li      $v0, 8          # syscall: read string
    la      $a0, buffer
    li      $a1, 32
    syscall

    la      $t0, buffer     # start of input string
    li      $t1, 0          # integer value accumulator
    li      $t2, 1          # default sign is +1 (positive)

    # Loading some constants
    lb      $t3, 0($t0)     # current character from buffer
    li      $s0, 45
    li      $s1, 48
    li      $s2, 57
    li      $s3, 10
    li      $s4, 214748364  # limit to check overflow

    # Check if first character is '-'. If so, set sign = -1
    seq     $t2, $t3, $s0
    sub     $t2, $zero, $t2
    ori     $t2, $t2, 1

    # If $t2 < 0, we skip the sign character.
    # If $t2 > 0, we jump to digit conversion.
    blt     $t2, $zero, skip_sign
    bgt     $t2, $zero, convert_digit

skip_sign:
    addi    $t0, $t0, 1     # move pointer to next char

convert_digit:
    lb      $t3, 0($t0)

    # If we reach end of string, jump to hex printing.
    beq     $t3, $zero, start_hex
```

```

    beq    $t3, $s3, skip_sign

    # Check if character is digit (between '0' and '9');
    # otherwise, go to error.
    blt    $t3, $s1, error
    bgt    $t3, $s2, error

    # Convert ASCII digit to integer value
    addi   $t3, $t3, -48

    bgt    $t1, $s4, error
    beq    $t1, $s4, check_last_digit
    j      add_digit

check_last_digit:
    # maximum last digit for 214748364<?>
    li     $t4, 7
    slt    $t5, $t2, $zero
    add    $t4, $t4, $t5

    bgt    $t3, $t4, error

add_digit:
    mul    $t1, $t1, $s3
    addu   $t1, $t1, $t3
    addi   $t0, $t0, 1
    j      convert_digit

start_hex:
    mul    $t1, $t1, $t2
    move   $t2, $t1

    li     $v0, 4                # hex conversion start (0x prefix)
    la     $a0, result_msg
    syscall

    li     $t0, 8                # set count: 8 hex digits to print
    li     $t1, 28               # start shifting at bit position 28

hex_loop:
    srl    $t3, $t2, $t1
    andi   $t3, $t3, 0xF
    la     $t4, hex_lookup
    add    $t3, $t4, $t3
    lbu    $t5, 0($t3)
    move   $a0, $t5
    li     $v0, 11
    syscall

    addi   $t0, $t0, -1
    addi   $t1, $t1, -4
    bgt    $t0, $zero, hex_loop

    j      exit_program

error:
    li     $v0, 4
    la     $a0, err_msg
    syscall
    j      exit_program

exit_program:

```

```
li    $v0, 10
syscall
```

Output

```
The 2's compliment Hex of N is: 10
0x0000000a
-- program is finished running --

The 2's compliment Hex of N is: -15
0xffffffff1
-- program is finished running --

The 2's compliment Hex of N is: 256
0x00000100
-- program is finished running --

The 2's compliment Hex of N is: 1209839013919
Invalid input
-- program is finished running --
```