

```

.data
prompt:    .asciiz "Enter an integer: "
illegal_msg: .asciiz "Illegal Number!\n"
fib_msg:    .asciiz "First Fibonacci numbers:\n"
space:      .asciiz " "

.text
.globl main

main:
# Ask user for an integer
    li $v0, 4          # print input prompt
    la $a0, prompt
    syscall
    li $v0, 5          # read integer value
    syscall
    move $t0, $v0      # save N in $t0

# Check if N is at least 22
    blez $t0, illegal  # if N <= 0 → invalid
    blt $t0, 22, illegal # if N < 22 → invalid
    j start_fib        # valid input → go compute Fibonacci

illegal:
    li $v0, 4          # service: print string

# Show error message and restart input
    la $a0, illegal_msg
    syscall
    j main             # restart from the beginning

start_fib:
    li $v0, 4          # service: print string

# Print title for Fibonacci sequence and set initial values
    la $a0, fib_msg
    syscall
    li $t1, 0          # first term (F0)
    li $t2, 1          # second term (F1)
    li $t3, 0          # counter for how many terms printed

fib_loop:
# Continue while we still have terms left to display
    bge $t3, $t0, end   # if printed N terms → finish

```

```

    move $a0, $t1      # current Fibonacci term to print
    li $v0, 1
    syscall
    li $v0, 4          # print a separating space

# Print separator and update Fibonacci terms
    la $a0, space
    syscall
    addu $t4, $t1, $t2  # next = previous + current
    move $t1, $t2      # shift current to previous
    move $t2, $t4      # next becomes current

# Increase counter and loop back
    addi $t3, $t3, 1    # counter++
    j fib_loop         # loop again

end:
    li $v0, 10         # syscall to exit
    syscall

```