```
##############################################
#The Fibonacci numbers are the numbers in the following integer sequence.
# 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ...
# F(0)=0 , F(1) =1
#int fib(int n) {
#  if (n < 2) return n;
#  else return fib(n - 1) + fib(n - 2);
#}
##############################################
 .data
n: .word  5
sonuc: .space 4

 .text
 .globl main

main: la   $t0,n
 lw   $t0,($t0)
 addi $sp,$sp,-4
 sw   $t0, 0($sp)        # n  is our argument
      jal  fib                # Call the fib procedure
 addi $sp, $sp,   4      # Pop off the argument
 la   $s0,sonuc
 sw   $v0,0($s0)  # result is in $v0
 li $v0, 10   # code for program end
 syscall

fib:  addi $sp, $sp,   -8     # Entry code
      sw   $ra, 0($sp)
      sw   $fp, 4($sp)
      add  $fp, $sp,   $zero  # End of entry code

      # Compare n with 2
      lw   $t0, 8($fp)        # $t0 holds the argument n
      slti $t1, $t0,   2      # if $t0 < 2 ...
      beq  $t1, $zero, over   # ... skip the next two instructions

      # n < 2


      slti $t1,$t0,1
      beq  $t1,$zero,next
      add $v0,$zero,$zero
      j    exit
next: addi $v0, $zero, 1      # We're done with the recursion
      j    exit               # Jump to the exit code

over: # n >= 2

      # Calculate fib(n - 1)
      addi $t0, $t0,   -1     # Calculate n - 1

      # Set up to call fib with argument n - 1
                              # No registers need to be saved
      addi $sp, $sp,   -4     # Allocate space for arguments
      sw   $t0, 0($sp)        # n - 1 is our argument
      jal  fib                # Call the fib procedure

      # Clean up after calling fib with argument n - 1
      addi $sp, $sp,   4      # Pop off the argument
                              # No registers need to be restored

      # $v0 holds the result of fib(n - 1)
      add  $t1, $v0,   $zero  # Put the result into $t1

      # Calculate fib(n - 2)
      lw   $t0, 8($fp)        # $t0 holds the argument n
      addi $t0, $t0,   -2     # Calculate n - 2

      # Set up to call fib with argument n - 2
      addi $sp, $sp,   -4     # Allocate space for saved register
      sw   $t1, 0($sp)        # Save $t1 (the result of fib(n - 1))
      addi $sp, $sp,   -4     # Allocate space for arguments
      sw   $t0, 0($sp)        # n - 2 is our argument
      jal  fib                # Call the fib procedure

      # Clean up after calling fib with argument n - 2
      addi $sp, $sp,   4      # Pop off the argument
      lw   $t1, 0($sp)        # Restore $t1 (the result of fib(n - 1))
      addi $sp, $sp,   4      # Deallocate space for saved register

      # $v0 holds the result of fib(n - 2)
      add  $v0, $t1,   $v0    # Result is fib(n - 1) + fib(n - 2)

exit: lw   $ra, 0($sp)        # Exit code
      lw   $fp, 4($sp)
      addi $sp, $sp,   8
      jr   $ra                # End of exit code
```