```
; recursively sum up the integers from 1 to N
; assume that input (N) is in $1
; output is returned in $3

recSum:
  ; Save registers on the stack
  sw $1, -4($30)
  sw $2, -8($30)
  sw $4, -12($30)
  sw $31, -16($30)
  lis $4
  .word 16
  sub $30, $30, $4

  ; Initialize sum so far
  add $3, $0, $0

  ; Check to see if we are in the base case (N=0)
  beq $1, $0, done

  ; Otherwise, we must compute the sum of the current N
  ; and the sum of the rest
  ; keep a copy of the current value of N
  add $2, $1, $0

  ; put N-1 into register $1
  lis $4
  .word 1
  sub $1, $1, $4
  ; get ready to call routine (i.e., ourself)
  lis $4
  .word recSum
  jalr $4
  ; add the value we got back to the current value of N (in $2)
  add $3, $3, $2

done:
  ; restore registers
  lis $4
  .word 16
  add $30, $30, $4
  lw $1, -4($30)
  lw $2, -8($30)
  lw $4, -12($30)
```

```
lw $31, -16($30)
jr $31
```