Factorial with Loop:

Write an assembly code for calculating factorial using loop.

Then write the same using the recursion .

1) Create and Use looping function to calculate the factorial
2) Then think how to call the function recursively.

```
.main:
   mov r0, 5          /* n = 5 */
   mov r1, 1          /* result = 1 */

.loop:
   cmp r0, 1
   bgt .multiply
   b .done

.multiply:
   mul r1, r1, r0
   sub r0, r0, 1
   b .loop

.done:
   Nop
```

Factorial with recursion

```
.factorial:

   /* if (n == 1) return 1 */
   cmp r0, 1
   beq .base

   /* allocate 8 bytes on stack */
   sub r14, r14, 8

   /* save n */
   st r0, 0[r14]

   /* save return address */
   st r15, 4[r14]
```

```
    /* compute factorial(n-1) */
    sub r0, r0, 1
    call .factorial

    /* restore n */
    ld r0, 0[r14]

    /* restore return address */
    ld r15, 4[r14]

    /* multiply n * factorial(n-1) */
    mul r1, r0, r1

    /* deallocate stack */
    add r14, r14, 8

    ret


.base:
    mov r1, 1
    ret

.main:
    mov r14, 4000       /* initialize stack pointer (sp = r14) */
    mov r0, 5           /* n = 5 */
    call .factorial
```

Sum of N numbers:

```
/* =================================== */
/*  RECURSIVE SUM OF N NUMBERS       */
/* =================================== */

.sum:

    /* ---- BASE CASE ---- */
    cmp r0, 0
    beq .base

    /* ---- RECURSIVE CASE ---- */
```

```
   sub r14, r14, 8      /* allocate stack space */

   st r0, 0[r14]        /* save n */
   st r15, 4[r14]       /* save return address */

   sub r0, r0, 1        /* n = n - 1 */
   call .sum            /* sum(n-1) */

   ld r0, 0[r14]        /* restore n */
   ld r15, 4[r14]       /* restore return address */

   add r1, r1, r0       /* n + sum(n-1) */

   add r14, r14, 8      /* deallocate stack */

   ret


.base:
   mov r1, 0
   ret


.main:
   mov r14, 4000        /* initialize stack pointer */
   mov r0, 5            /* n = 5 */
   call .sum
```

Q) Write assembly level code for generating Fibonacci sequence.