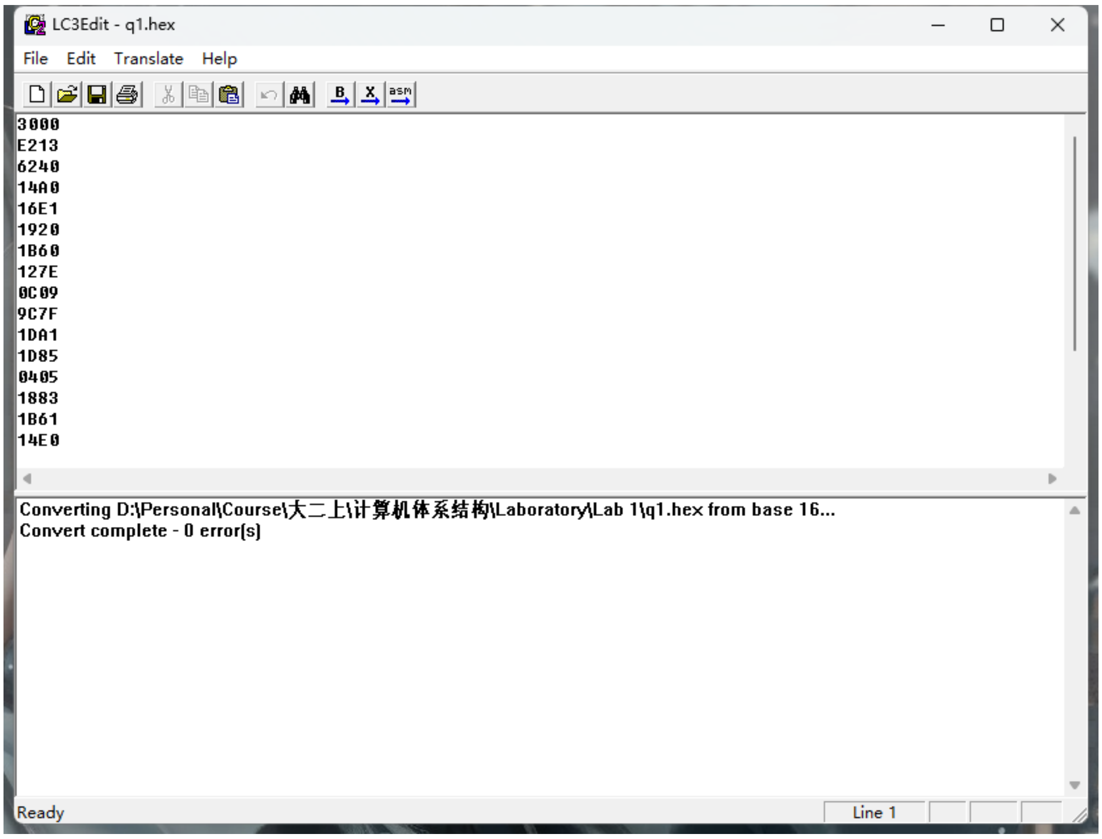
Question 1

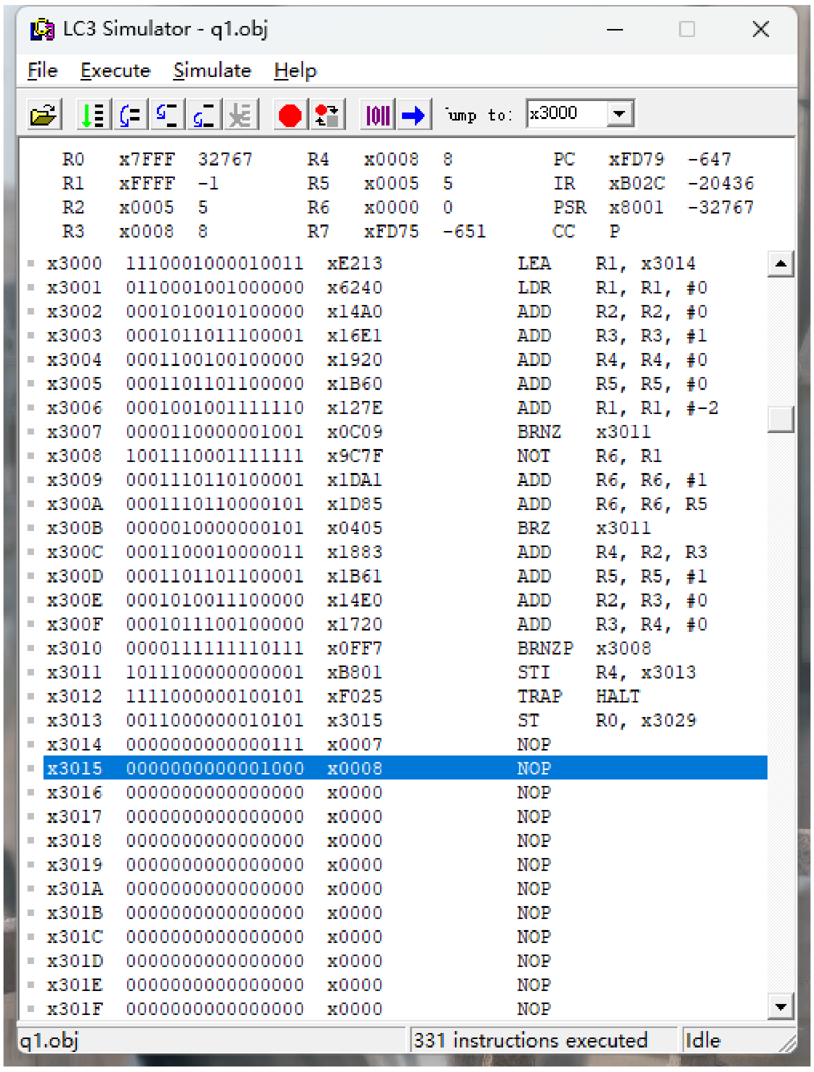
1. Copy the machine code to LC3Edit and convert from base 16, then .obj file is generated.



```
3000
E213
6240
14A0
16E1
1920
1B60
127E
0C09
9C7F
1DA1
1D85
0405
1883
1B61
14E0
```

```
Converting D:\Personal\Course\大二上\计算机体系结构\Laboratory\Lab 1\q1.hex from base 16...
Convert complete - 0 error(s)
```

Open .obj file in LC3 Simulator and set the value of 0x3014 to x0007, which is the last digit of my student ID(521030910387). After running the program, the result at memory 0x3015 is x0008.



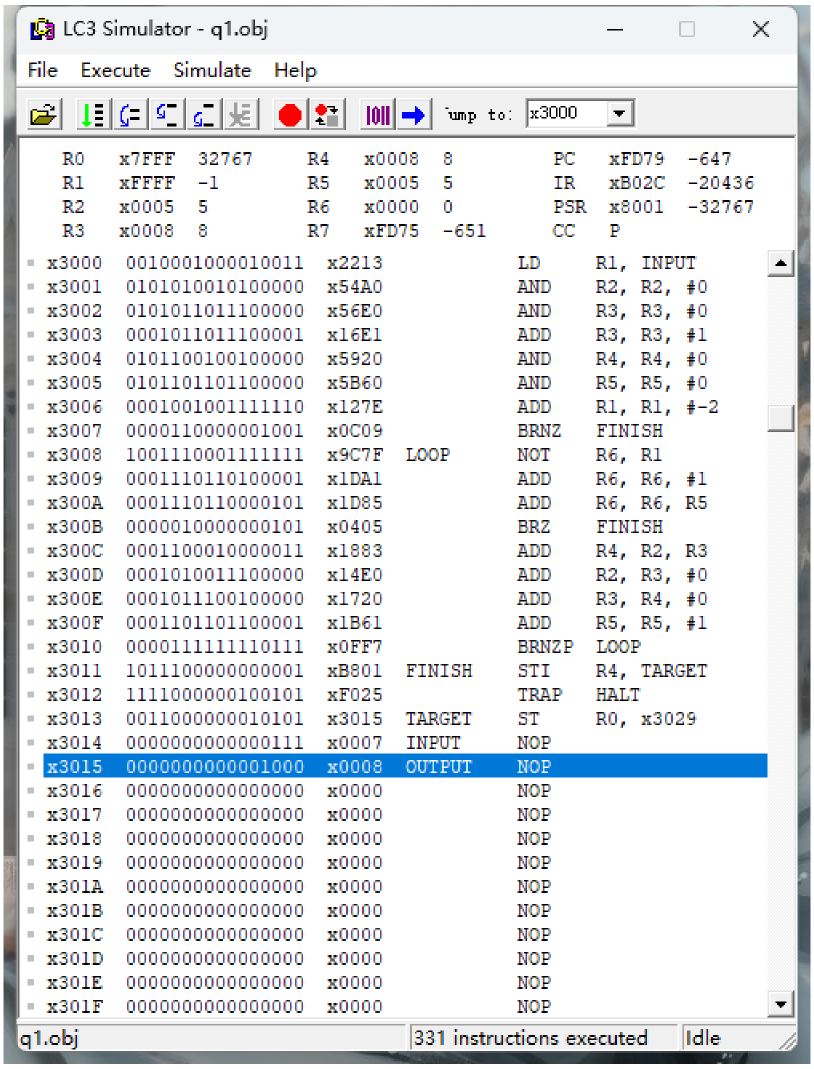
```
R0   x7FFF  32767     R4   x0008  8        PC   xFD79  -647
R1   xFFFF  -1        R5   x0005  5        IR   xB02C  -20436
R2   x0005  5         R6   x0000  0        PSR  x8001  -32767
R3   x0008  8         R7   xFD75  -651     CC   P

x3000  1110001000010011  xE213     LEA    R1, x3014
x3001  0110001001000000  x6240     LDR    R1, R1, #0
x3002  0001010010100000  x14A0     ADD    R2, R2, #0
x3003  0001011011100001  x16E1     ADD    R3, R3, #1
x3004  0001100100100000  x1920     ADD    R4, R4, #0
x3005  0001101101100000  x1B60     ADD    R5, R5, #0
x3006  0001001001111110  x127E     ADD    R1, R1, #-2
x3007  0000110000001001  x0C09     BRNZ   x3011
x3008  1001110001111111  x9C7F     NOT    R6, R1
x3009  0001110110100001  x1DA1     ADD    R6, R6, #1
x300A  0001110110000101  x1D85     ADD    R6, R6, R5
x300B  0000010000000101  x0405     BRZ    x3011
x300C  0001100010000011  x1883     ADD    R4, R2, R3
x300D  0001101101100001  x1B61     ADD    R5, R5, #1
x300E  0001010011100000  x14E0     ADD    R2, R3, #0
x300F  0001011100100000  x1720     ADD    R3, R4, #0
x3010  0000111111110111  x0FF7     BRNZP  x3008
x3011  1011100000000001  xB801     STI    R4, x3013
x3012  1111000000100101  xF025     TRAP   HALT
x3013  0011000000010101  x3015     ST     R0, x3029
x3014  0000000000000111  x0007     NOP
x3015  0000000000001000  x0008     NOP
x3016  0000000000000000  x0000     NOP
x3017  0000000000000000  x0000     NOP
x3018  0000000000000000  x0000     NOP
x3019  0000000000000000  x0000     NOP
x301A  0000000000000000  x0000     NOP
x301B  0000000000000000  x0000     NOP
x301C  0000000000000000  x0000     NOP
x301D  0000000000000000  x0000     NOP
x301E  0000000000000000  x0000     NOP
x301F  0000000000000000  x0000     NOP

q1.obj                    331 instructions executed    Idle
```

2. Observing the program in LC3 Simulator, we can see an iterative structure. In each iteration, the values in R2 and R3 are added and the result is stored in R4, then values of R3 and R4 are correspondingly moved back to R2 and R3. Therefore, this program functions as a calculator to figure out a specific term in a Fibonacci-like series where $a_0 = 0, a_1 = 0, a_2 = 0, a_3 = 1, a_4 = 2, a_5 = 3, a_6 = 5, a_7 = 8, a_8 = 13, a_9 = 21, \ldots$

3. Actually, this program always works correctly at first time. However, when a second run happens, the result becomes unpredictable. That is because the initialization part is wrong. Instead of ADD instruction, AND instruction is supposed to be used for initialization. Just refactor the initialization part and the problem can be fixed. A reasonable version can be written in assembly language as follow:

```
        .ORIG   x3000
        LD      R1, INPUT
        AND     R2, R2, #0
        AND     R3, R3, #0
        ADD     R3, R3, #1
        AND     R4, R4, #0
        AND     R5, R5, #0
        ADD     R1, R1, #-2
        BRNZ    FINISH
LOOP    NOT     R6, R1
        ADD     R6, R6, #1
        ADD     R6, R6, R5
        BRZ     FINISH
        ADD     R4, R2, R3
        ADD     R2, R3, #0
        ADD     R3, R4, #0
        ADD     R5, R5, #1
        BRNZP   LOOP
FINISH  STI     R4, TARGET
        HALT
TARGET  .FILL   OUTPUT
INPUT   NOP
OUTPUT  NOP
        .END
```

Convert this program to .obj and simulate in LC3 Simulator. No matter what initial state of the machine is, this program always works well.



```
R0   x7FFF  32767     R4   x0008  8        PC   xFD79  -647
R1   xFFFF  -1        R5   x0005  5        IR   xB02C  -20436
R2   x0005  5         R6   x0000  0        PSR  x8001  -32767
R3   x0008  8         R7   xFD75  -651     CC   P

x3000  0010001000010011  x2213           LD     R1, INPUT
x3001  0101010010100000  x54A0           AND    R2, R2, #0
x3002  0101011011100000  x56E0           AND    R3, R3, #0
x3003  0001011011100001  x16E1           ADD    R3, R3, #1
x3004  0101100100100000  x5920           AND    R4, R4, #0
x3005  0101101101100000  x5B60           AND    R5, R5, #0
x3006  0001001001111110  x127E           ADD    R1, R1, #-2
x3007  0000110000001001  x0C09           BRNZ   FINISH
x3008  1001110001111111  x9C7F  LOOP     NOT    R6, R1
x3009  0001110110100001  x1DA1           ADD    R6, R6, #1
x300A  0001110110000101  x1D85           ADD    R6, R6, R5
x300B  0000010000000101  x0405           BRZ    FINISH
x300C  0001100010000011  x1883           ADD    R4, R2, R3
x300D  0001010011100000  x14E0           ADD    R2, R3, #0
x300E  0001011100100000  x1720           ADD    R3, R4, #0
x300F  0001101101100001  x1B61           ADD    R5, R5, #1
x3010  0000111111110111  x0FF7           BRNZP  LOOP
x3011  1011100000000001  xB801  FINISH   STI    R4, TARGET
x3012  1111000000100101  xF025           TRAP   HALT
x3013  0011000000010101  x3015  TARGET   ST     R0, x3029
x3014  0000000000000111  x0007  INPUT    NOP
x3015  0000000000001000  x0008  OUTPUT   NOP
x3016  0000000000000000  x0000           NOP
x3017  0000000000000000  x0000           NOP
x3018  0000000000000000  x0000           NOP
x3019  0000000000000000  x0000           NOP
x301A  0000000000000000  x0000           NOP
x301B  0000000000000000  x0000           NOP
x301C  0000000000000000  x0000           NOP
x301D  0000000000000000  x0000           NOP
x301E  0000000000000000  x0000           NOP
x301F  0000000000000000  x0000           NOP

q1.obj                    331 instructions executed    Idle
```