Microprocessor and Computer Architecture Laboratory

UE19CS256

4th Semester, Academic Year 2020-21

Date:11/2/2021

Name: Ramya N Prabhu	SRN:PES1UG19CS380	Section F

Week#____3_____Program Number: ____1___

Write an ALP to add two 64 bit numbers loaded from memory and store the result in memory

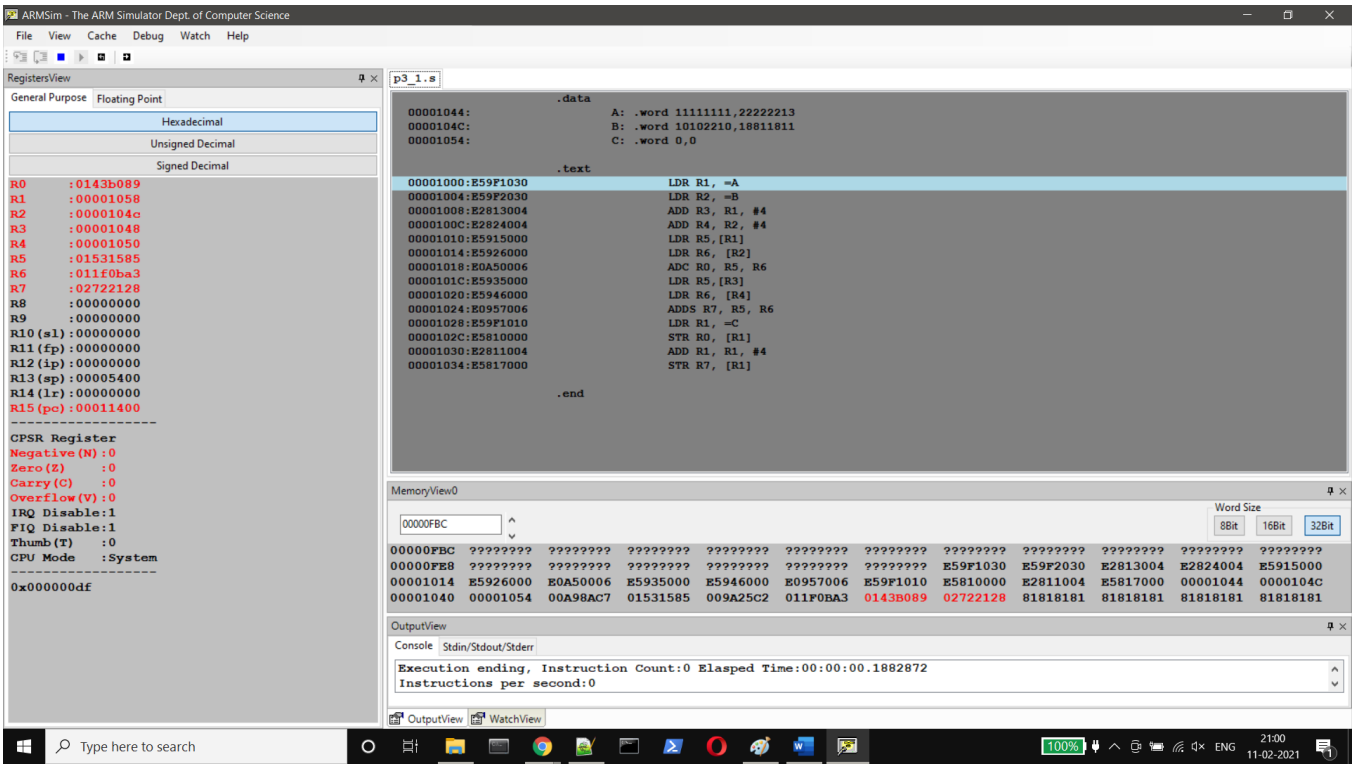
i. ARM Assembly Code for each program

```
.data
        A: .word 11111111,22222213
        B: .word 10102210,18811811
        C: .word 0,0
.text
                LDR R1, =A
                LDR R2, =B
                ADD R3, R1, #4
                ADD R4, R2, #4
                LDR R5,[R1]
                LDR R6, [R2]
                ADC R0, R5, R6
                LDR R5,[R3]
                LDR R6, [R4]
                ADDS R7, R5, R6
                LDR R1, =C
                STR R0, [R1]
                ADD R1, R1, #4
```

STR R7, [R1]

.end

ii. Output Screenshot



Week#____3_____Program Number: ____2___

Write an ALP to copy n numbers from Memory Location A to Memory Location B

i. ARM Assembly Code for each program

.data

a: .word 1,2,3,4,5,5

b: .word 0,0,0,0,0,0

.text

LDR R0, =a

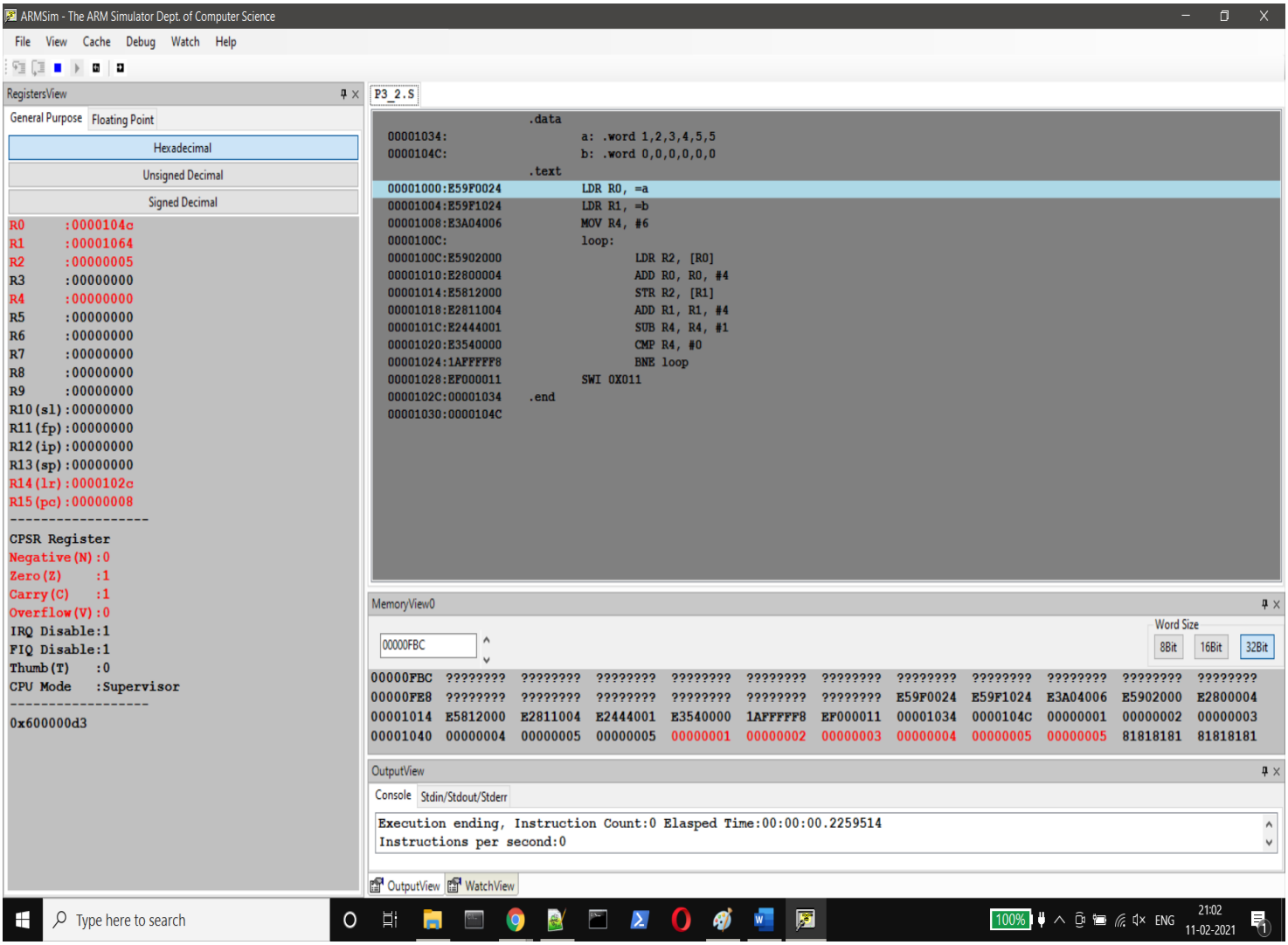
LDR R1, =b

MOV R4, #6

loop:

```
        LDR R2, [R0]
        ADD R0, R0, #4
        STR R2, [R1]
        ADD R1, R1, #4
        SUB R4, R4, #1
        CMP R4, #0
        BNE loop
    SWI 0X011
.end
```

ii. Output Screenshot



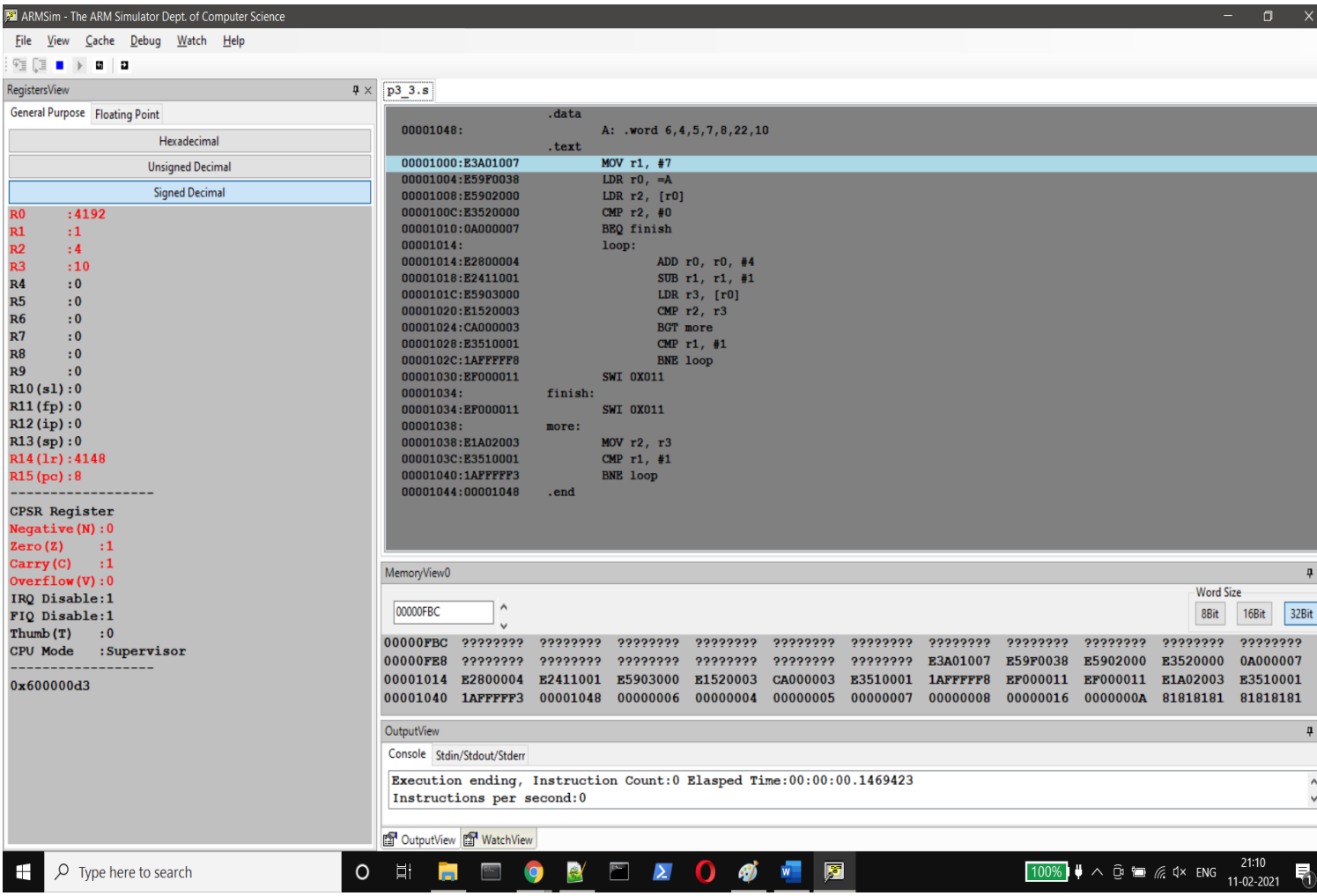
Week#____3_____Program Number: ____3____

Write an ALP to find smallest number in an array of n - 32 bit numbers

i. ARM Assembly Code for each program

```
.data
        A: .word 6,4,5,7,8,22,10
.text
        MOV r1, #7
        LDR r0, =A
        LDR r2, [r0]
        CMP r2, #0
        BEQ finish
        loop:
                ADD r0, r0, #4
                SUB r1, r1, #1
                LDR r3, [r0]
                CMP r2, r3
                BGT more
                CMP r1, #1
                BNE loop
        SWI 0X011
finish:
        SWI 0X011
more:
        MOV r2, r3
        CMP r1, #1
        BNE loop
.end            ;r2 contains the value of the smallest array element
```

ii. Output Screenshot



Week#____3_____Program Number: ____4,a____

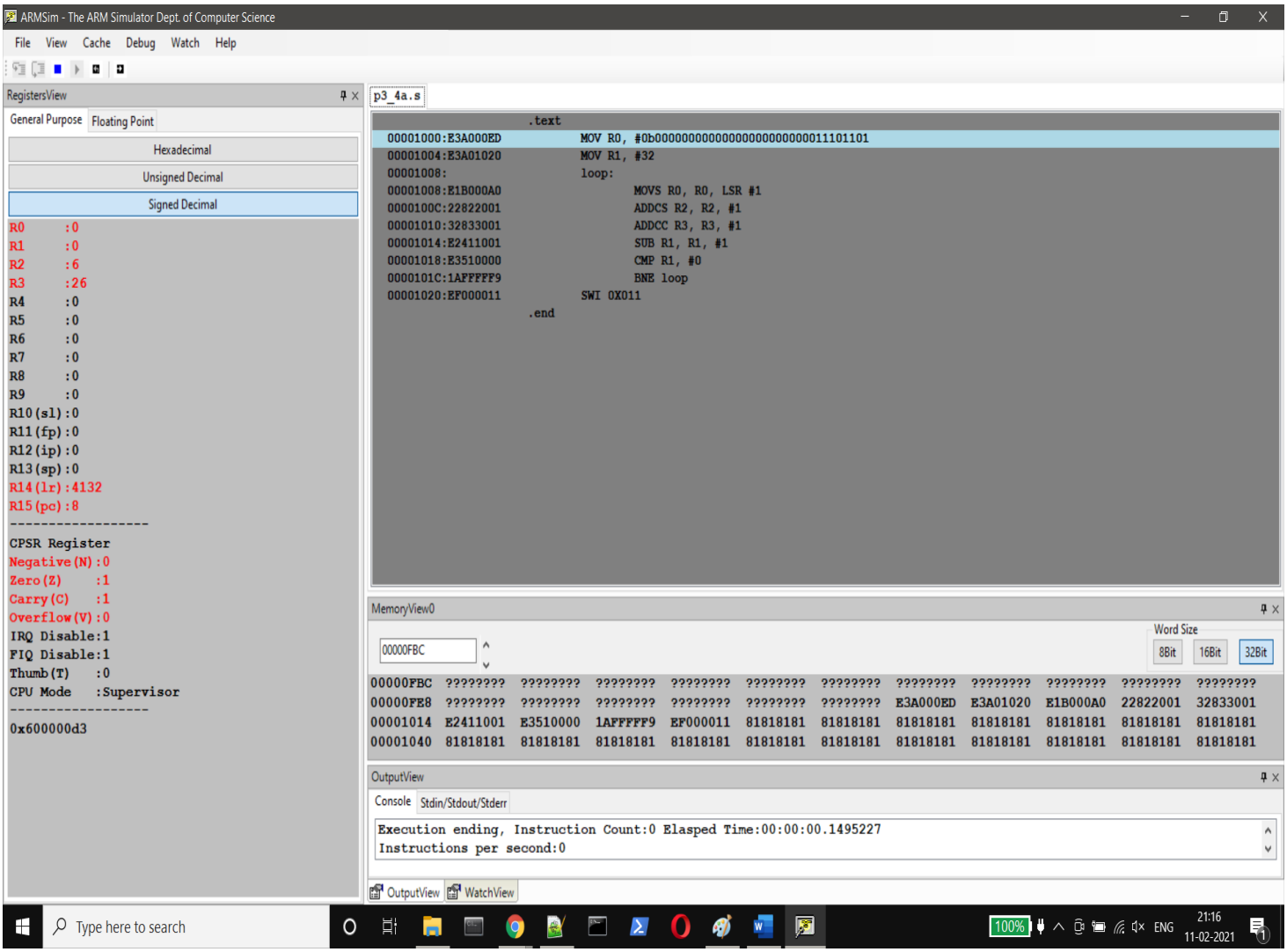
Write an ALP to count the number of 1's and 0's in a given 32 bit number.

i. ARM Assembly code for each program

```
.text
        MOV R0, #0b00000000000000000000000011101101
        MOV R1, #32
        loop:
                MOVS R0, R0, LSR #1
                ADDCS R2, R2, #1
```

```
        ADDCC R3, R3, #1
        SUB R1, R1, #1
        CMP R1, #0
        BNE loop
    SWI 0X011
.end
```

ii.Output Screenshot



Week#____3_____Program Number: ____4,b____

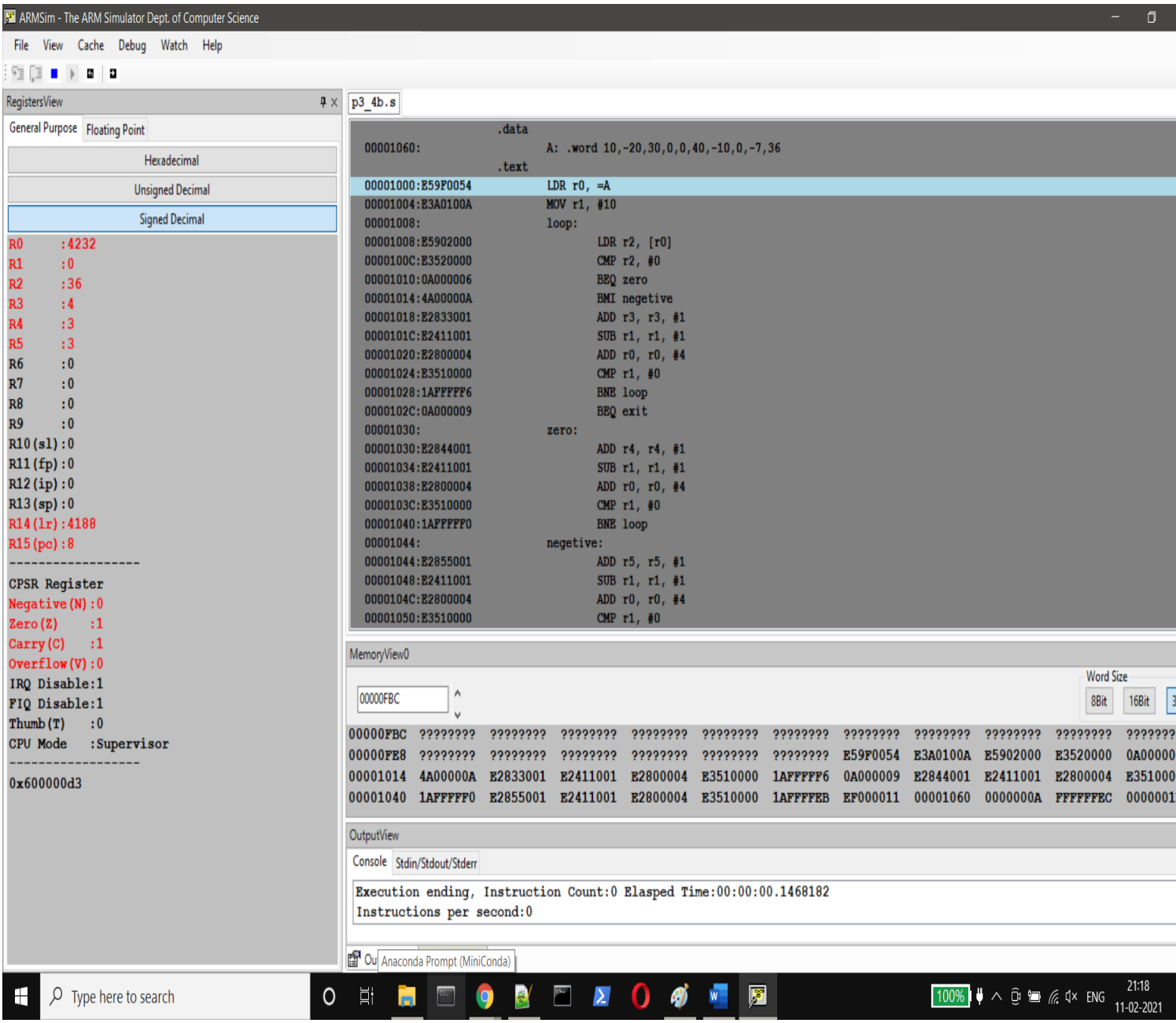
Write an ALP to find the number of zeroes,

positive and negative numbers in a given array

i. ARM Assembly Code

```
.data
        A: .word 10,-20,30,0,0,40,-10,0,-7,36
.text
        LDR r0, =A
        MOV r1, #10
        loop:
                LDR r2, [r0]
                CMP r2, #0
                BEQ zero
                BMI negetive
                ADD r3, r3, #1
                SUB r1, r1, #1
                ADD r0, r0, #4
                CMP r1, #0
                BNE loop
                BEQ exit
        zero:
                ADD r4, r4, #1
                SUB r1, r1, #1
                ADD r0, r0, #4
                CMP r1, #0
                BNE loop
        negetive:
                ADD r5, r5, #1
                SUB r1, r1, #1
                ADD r0, r0, #4
                CMP r1, #0
                BNE loop
        exit:
                SWI 0X011
.end
```

ii. Output Screenshot



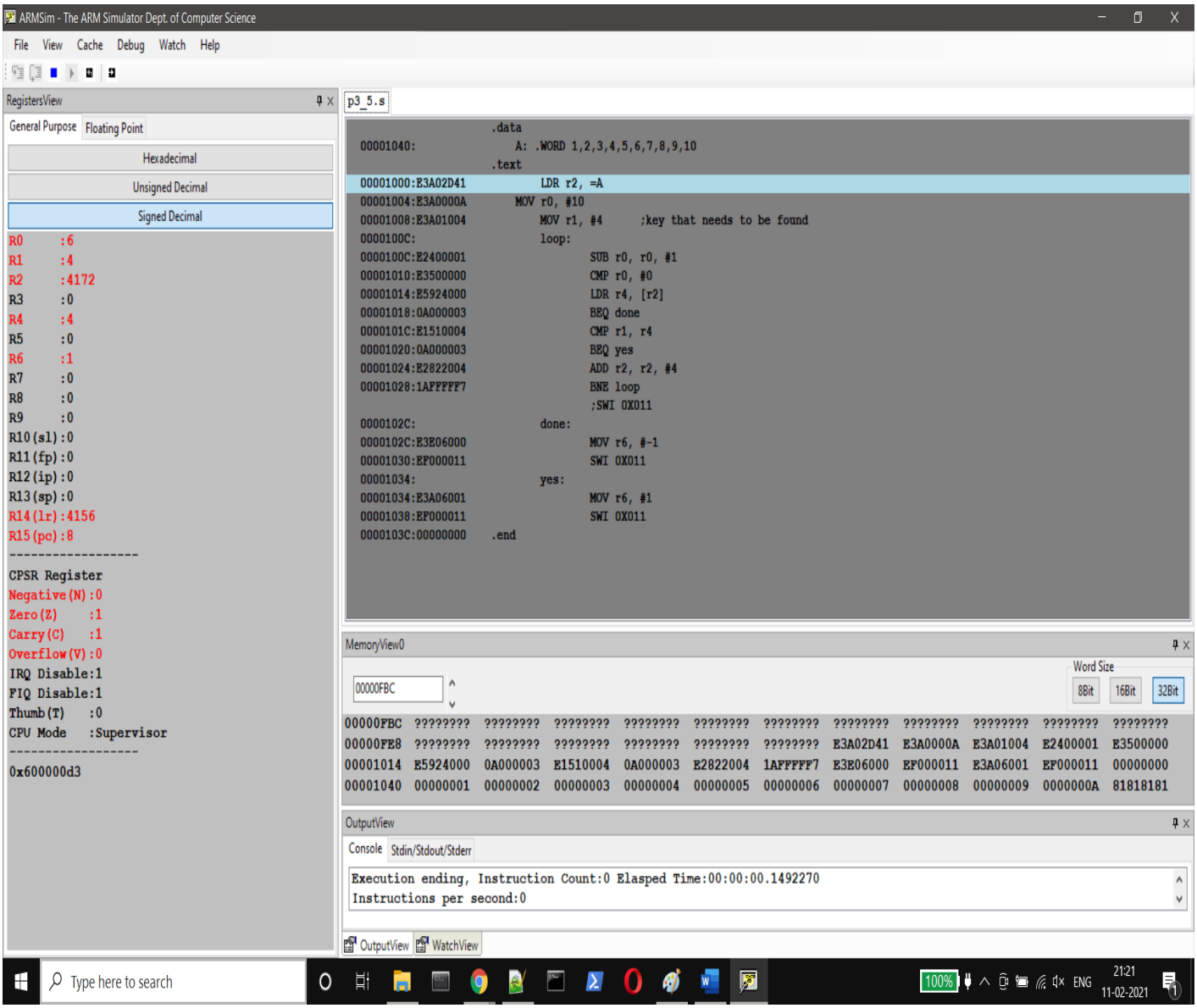
Week#____3_____Program Number: ____5 ____

Write an ALP to check whether a given number is present in array using Linear Search (Without SWI 0x02), if found move +1 to R6 and key position to R7 else move -1 to R6 (if number not found)

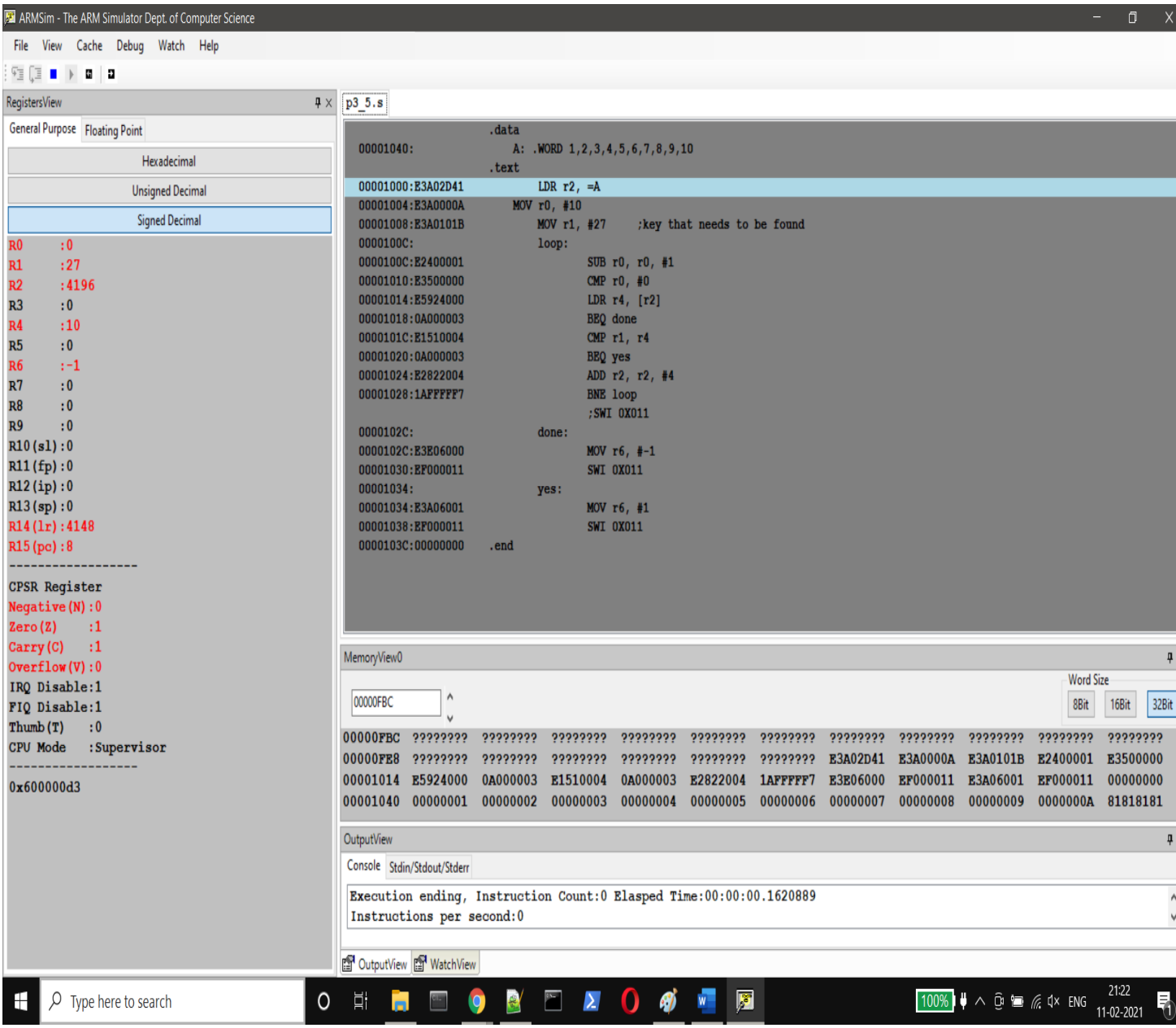
i. ARM Assembly Code

```
.data
   A: .WORD 1,2,3,4,5,6,7,8,9,10
.text
                LDR r2, =A
   MOV r0, #10
                MOV r1, #4          ;key that needs to be found
                loop:
                    SUB r0, r0, #1
                    CMP r0, #0
                    LDR r4, [r2]
                    BEQ done
                    CMP r1, r4
                    BEQ yes
                    ADD r2, r2, #4
                    BNE loop
                    ;SWI 0X011
                done:
                    MOV r6, #-1
                    SWI 0X011
                yes:
                    MOV r6, #1
                    SWI 0X011
.end
```

ii. Output Screenshot



Case num not found:



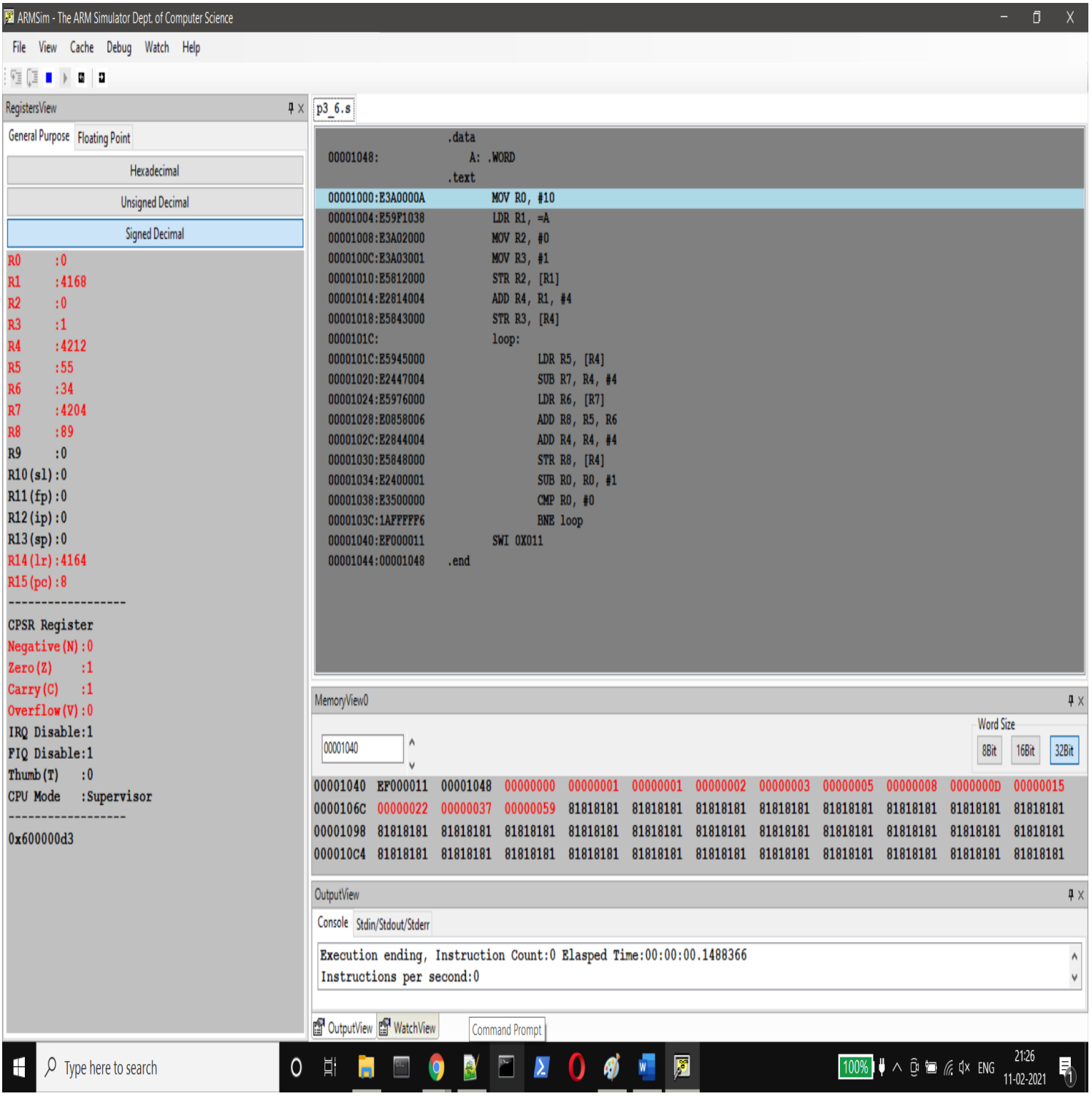
Week#____3_____Program Number: ____6 ____

Write an ALP to generate Fibonacci Series and store them in an array

i. ARM Assembly Code

```
.data
   A: .WORD
.text
                    MOV R0, #10
                    LDR R1, =A
                    MOV R2, #0
                    MOV R3, #1
                    STR R2, [R1]
                    ADD R4, R1, #4
                    STR R3, [R4]
                    loop:
                       LDR R5, [R4]
                       SUB R7, R4, #4
                       LDR R6, [R7]
                       ADD R8, R5, R6
                       ADD R4, R4, #4
                       STR R8, [R4]
                       SUB R0, R0, #1
                       CMP R0, #0
                       BNE loop
                    SWI 0X011
.end
```

ii. Output Screenshot


ARMSim - The ARM Simulator Dept. of Computer Science
File View Cache Debug Watch Help
RegistersView
General Purpose Floating Point
Hexadecimal
Unsigned Decimal
Signed Decimal
R0 :0
R1 :4168
R2 :0
R3 :1
R4 :4212
R5 :55
R6 :34
R7 :4204
R8 :89
R9 :0
R10(sl):0
R11(fp):0
R12(ip):0
R13(sp):0
R14(lr):4164
R15(pc):8
CPSR Register
Negative(N):0
Zero(Z) :1
Carry(C) :1
Overflow(V):0
IRQ Disable:1
FIQ Disable:1
Thumb(T) :0
CPU Mode :Supervisor
0x600000d3
p3_6.s
.data
00001048: A: .WORD
.text
00001000:E3A0000A MOV R0, #10
00001004:E59F1038 LDR R1, =A
00001008:E3A02000 MOV R2, #0
0000100C:E3A03001 MOV R3, #1
00001010:E5812000 STR R2, [R1]
00001014:E2814004 ADD R4, R1, #4
00001018:E5843000 STR R3, [R4]
0000101C: loop:
0000101C:E5945000 LDR R5, [R4]
00001020:E2447004 SUB R7, R4, #4
00001024:E5976000 LDR R6, [R7]
00001028:E0858006 ADD R8, R5, R6
0000102C:E2844004 ADD R4, R4, #4
00001030:E5848000 STR R8, [R4]
00001034:E2400001 SUB R0, R0, #1
00001038:E3500000 CMP R0, #0
0000103C:1AFFFFF6 BNE loop
00001040:EF000011 SWI 0X011
00001044:00001048 .end
MemoryView0
00001040
Word Size
8Bit 16Bit 32Bit
00001040 EF000011 00001048 00000000 00000001 00000001 00000002 00000003 00000005 00000008 0000000D 00000015
0000106C 00000022 00000037 00000059 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
00001098 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010C4 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
OutputView
Console Stdin/Stdout/Stderr
Execution ending, Instruction Count:0 Elasped Time:00:00:00.1488366
Instructions per second:0
OutputView WatchView Command Prompt
Type here to search
100% ENG 21:26 11-02-2021