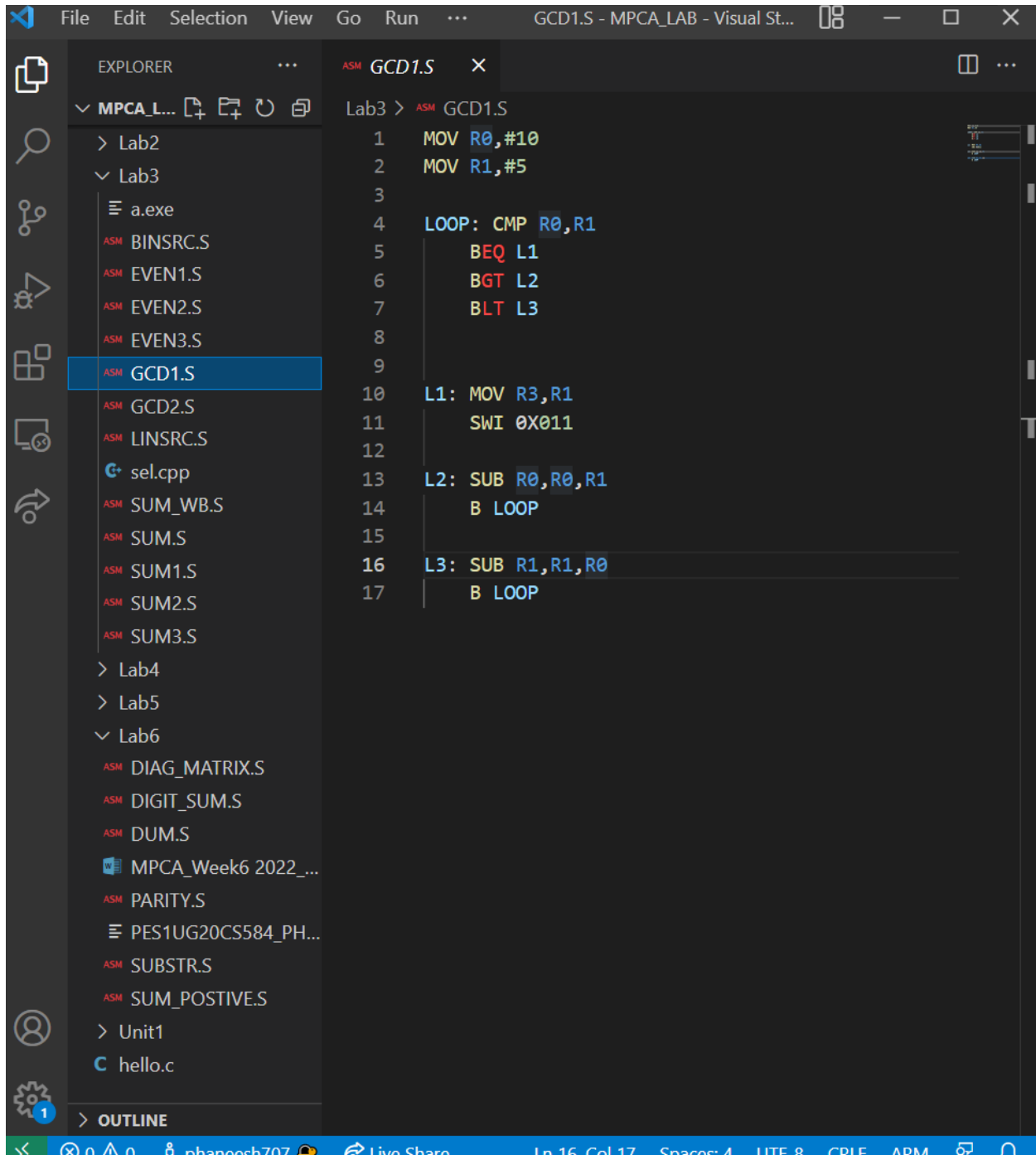


**ARM7TDMI  
PROGRAMMING  
WEEK-3**

**PHANEESH  
PES1UG20CS584  
J SECTION**

1. Write a program in ARM7TDMI-ISA to find GCD of two numbers.

a. Assume operands to be in the CPU registers.



The screenshot shows the Visual Studio Code editor with a project named 'MPCA\_LAB'. The Explorer pane on the left shows a directory structure with 'Lab3' expanded, containing several assembly files. 'GCD1.S' is selected and highlighted in blue. The main editor pane displays the contents of 'GCD1.S', which is an ARM assembly program for finding the Greatest Common Divisor (GCD) of two numbers. The program uses registers R0, R1, and R3. It initializes R0 to 10 and R1 to 5, then enters a loop that compares R0 and R1. If they are equal, it jumps to label L1. If R0 is greater than R1, it jumps to label L2. If R0 is less than R1, it jumps to label L3. Label L1 moves the value of R1 into R3 and then swi 0x011. Label L2 subtracts R1 from R0 and branches back to the loop. Label L3 subtracts R0 from R1 and branches back to the loop.

```
Lab3 > ASM GCD1.S
1  MOV R0,#10
2  MOV R1,#5
3
4  LOOP: CMP R0,R1
5          BEQ L1
6          BGT L2
7          BLT L3
8
9
10 L1: MOV R3,R1
11     SWI 0X011
12
13 L2: SUB R0,R0,R1
14     B LOOP
15
16 L3: SUB R1,R1,R0
17     B LOOP
```

ARMSim - The ARM Simulator Dept. of Computer Science

FileViewCacheDebugWatchHelp

RegistersView

General Purpose

Hexadecimal

Unsigned Decimal

Signed Decimal

R0 : 5

R1 : 5

R2 : 0

R3 : 5

R4 : 0

R5 : 0

R6 : 0

R7 : 0

R8 : 0

R9 : 0

R10 (s1) : 0

R11 (fp) : 0

R12 (ip) : 0

R13 (sp) : 21504

R14 (lr) : 0

R15 (pc) : 4124

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 : Sy

0x600000df

GCD1.S

00001000:E3A0000A MOV R0,#10

00001004:E3A01005 MOV R1,#5

00001008:E1500001 LOOP: CMP R0,R1

0000100C:0A000001 BEQ L1

00001010:CA000002 BGT L2

00001014:BA000003 BLT L3

00001018:E1A03001 L1: MOV R3,R1

0000101C:EF000011 SWI 0X011

00001020:E0400001 L2: SUB R0,R0,R1

00001024:EAF00007 B LOOP

00001028:E0411000 L3: SUB R1,R1,R0

0000102C:EAF00005 B LOOP

MemoryView0

Word Size

8Bit

16Bit

32Bit

00001050 81818181 81818181 81818181 81818181 81818181

00001064 81818181 81818181 81818181 81818181 81818181

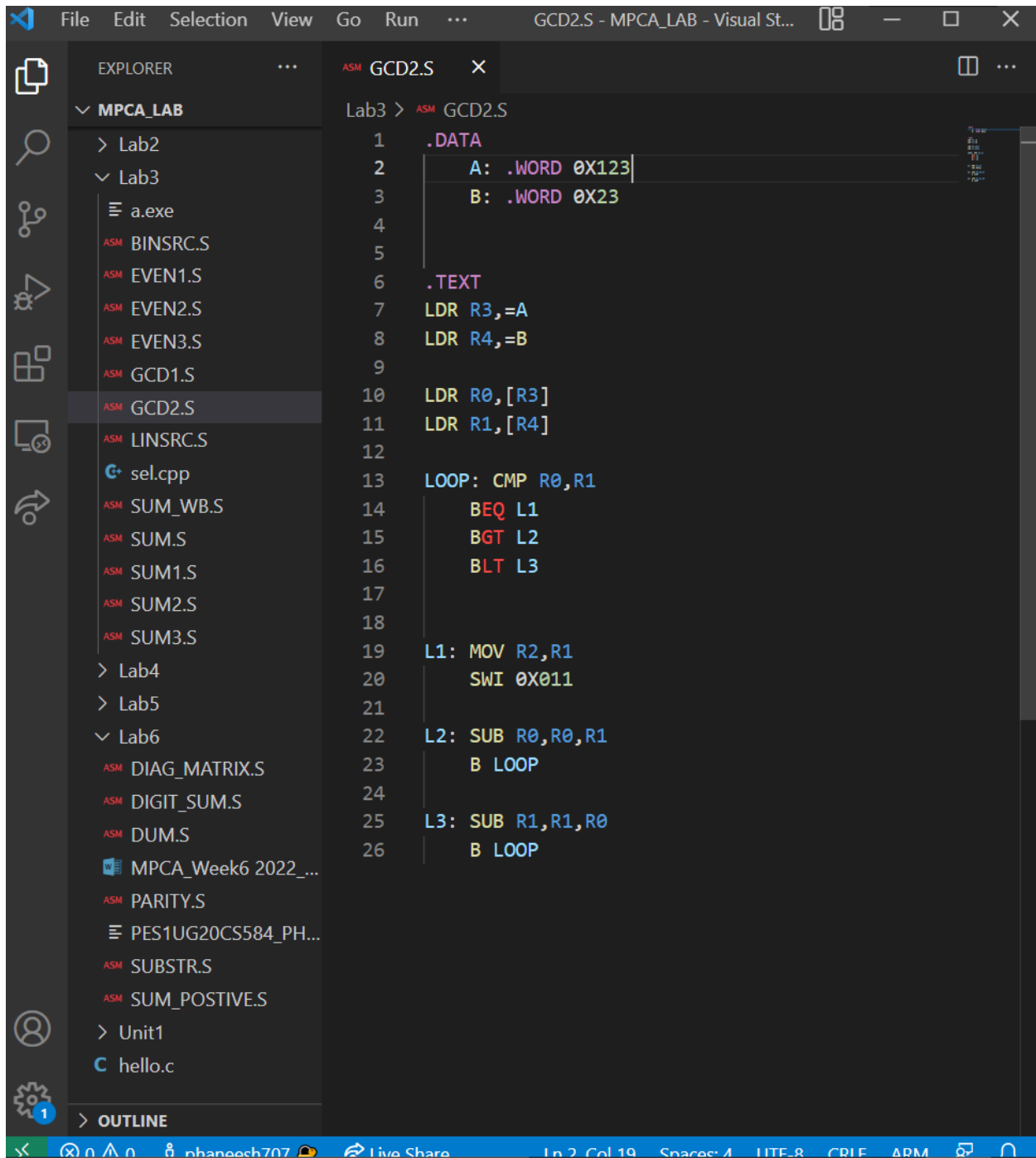
00001078 81818181 81818181 81818181 81818181 81818181

0000108C 81818181 81818181 81818181 81818181 81818181

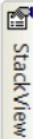
000010A0 81818181 81818181 81818181 81818181 81818181

OutputView

b. Assume operands in the memory locations.

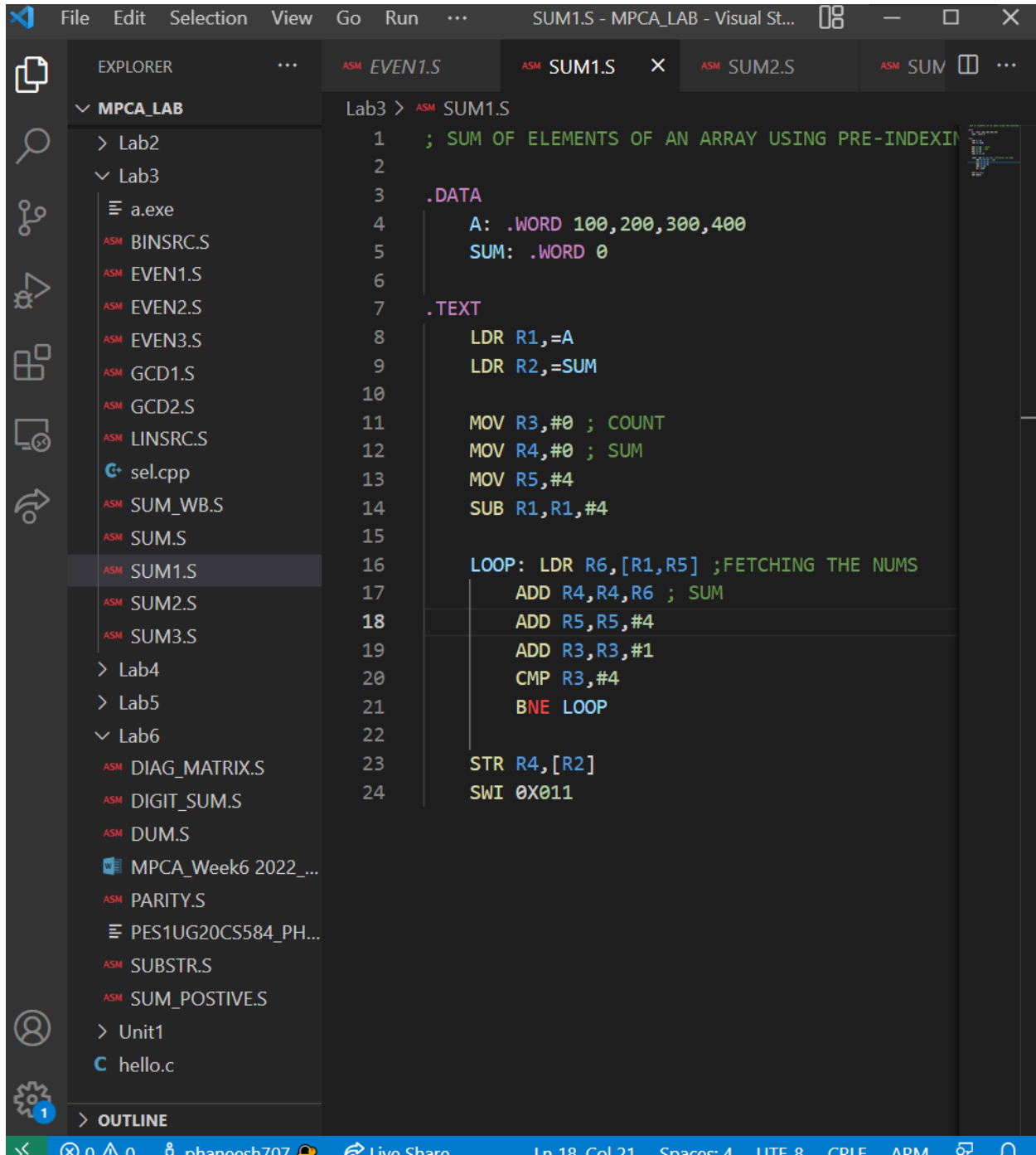


```
1  .DATA
2  A: .WORD 0X123
3  B: .WORD 0X23
4
5
6  .TEXT
7  LDR R3,=A
8  LDR R4,=B
9
10 LDR R0,[R3]
11 LDR R1,[R4]
12
13 LOOP: CMP R0,R1
14      BEQ L1
15      BGT L2
16      BLT L3
17
18
19 L1: MOV R2,R1
20     SWI 0X011
21
22 L2: SUB R0,R0,R1
23     B LOOP
24
25 L3: SUB R1,R1,R0
26     B LOOP
```



2. Write a program in ARM7TDMI-ISA to find the sum of N data items in the memory. Store the result in the memory location.

a. Use Pre-indexing addressing mode.



The screenshot shows the Visual Studio Code editor with the file explorer on the left and the assembly code editor in the center. The file explorer shows a project named 'MPCA\_LAB' with several subfolders and files. The assembly code editor shows the following code:

```
1 ; SUM OF ELEMENTS OF AN ARRAY USING PRE-INDEXING
2
3 .DATA
4     A: .WORD 100,200,300,400
5     SUM: .WORD 0
6
7 .TEXT
8     LDR R1,=A
9     LDR R2,=SUM
10
11     MOV R3,#0 ; COUNT
12     MOV R4,#0 ; SUM
13     MOV R5,#4
14     SUB R1,R1,#4
15
16     LOOP: LDR R6,[R1,R5] ;FETCHING THE NUMS
17           ADD R4,R4,R6 ; SUM
18           ADD R5,R5,#4
19           ADD R3,R3,#1
20           CMP R3,#4
21           BNE LOOP
22
23     STR R4,[R2]
24     SWI 0X011
```

ARMSim - The ARM Simulator Dept. of Computer Science

File

View

Cache

Debug

Watch

Help

RegistersView

General Purpose

Hexadecimal

Unsigned Decimal

Signed Decimal

R0 : 0

R1 : 4156

R2 : 4176

R3 : 4

R4 : 1000

R5 : 20

R6 : 400

R7 : 0

R8 : 0

R9 : 0

R10 (s1) : 0

R11 (fp) : 0

R12 (ip) : 0

R13 (sp) : 21504

R14 (lr) : 0

R15 (pc) : 4148

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 : Sy

0x600000df

SUM1.S

; SUM OF ELEMENTS OF AN ARRAY USING PRE-INDEXING

.DATA

00001040: A: .WORD 100,200,300,400

00001050: SUM: .WORD 0

.TEXT

00001000:E3A01D41 LDR R1,=A

00001004:E59F202C LDR R2,=SUM

00001008:E3A03000 MOV R3,#0 ; COUNT

0000100C:E3A04000 MOV R4,#0 ; SUM

00001010:E3A05004 MOV R5,#4

00001014:E2411004 SUB R1,R1,#4

00001018:E7916005 LOOP: LDR R6,[R1,R5] ;FETCHING THE NUMS

0000101C:E0844006 ADD R4,R4,R6 ; SUM

00001020:E2855004 ADD R5,R5,#4

00001024:E2833001 ADD R3,R3,#1

00001028:E3530004 CMP R3,#4

0000102C:1AFFFFF9 BNE LOOP

00001030:E5824000 STR R4,[R2]

SWI 0X011

MemoryView0

0000104C

Word Size

8Bit

16Bit

32Bit

0000104C	00000190	000003E8	81818181	81818181	81818181
00001060	81818181	81818181	81818181	81818181	81818181
00001074	81818181	81818181	81818181	81818181	81818181
00001088	81818181	81818181	81818181	81818181	81818181
0000109C	81818181	81818181	81818181	81818181	81818181

OutputView

b. Use Post-indexing addressing mode.

```
File Edit Selection View Go Run ... SUM2.S - MPCA_LAB - Visual St...  
EXPLORER  
MPCA_LAB  
  Lab2  
  Lab3  
    a.exe  
    BINSRC.S  
    EVEN1.S  
    EVEN2.S  
    EVEN3.S  
    GCD1.S  
    GCD2.S  
    LINSRC.S  
    sel.cpp  
    SUM_WB.S  
    SUM.S  
    SUM1.S  
    SUM2.S  
    SUM3.S  
  Lab4  
  Lab5  
  Lab6  
    DIAG_MATRIX.S  
    DIGIT_SUM.S  
    DUM.S  
    MPCA_Week6 2022_...  
    PARITY.S  
    PES1UG20CS584_PH...  
    SUBSTR.S  
    SUM_POSTIVE.S  
  Unit1  
  hello.c  
  OUTLINE  
Lab3 > ASM SUM2.S  
1 ; SUM OF ELEMENTS OF AN ARRAY USING POST-INDEXING  
2  
3 .DATA  
4     A: .WORD 100,200,300,400,500,600  
5     SUM: .WORD 0  
6  
7 .TEXT  
8     LDR R1,=A  
9     LDR R2,=SUM  
10  
11     MOV R3,#0 ; COUNT  
12     MOV R4,#0 ; SUM  
13     MOV R5,#4  
14  
15     LOOP: LDR R6,[R1],R5;FETCHING THE NUMS  
16           ADD R4,R4,R6 ; SUM  
17           ADD R3,R3,#1  
18           CMP R3,#6  
19           BNE LOOP  
20  
21     STR R4,[R2]  
22     SWI 0X011
```

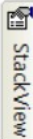




c. Use Auto-indexing addressing mode.

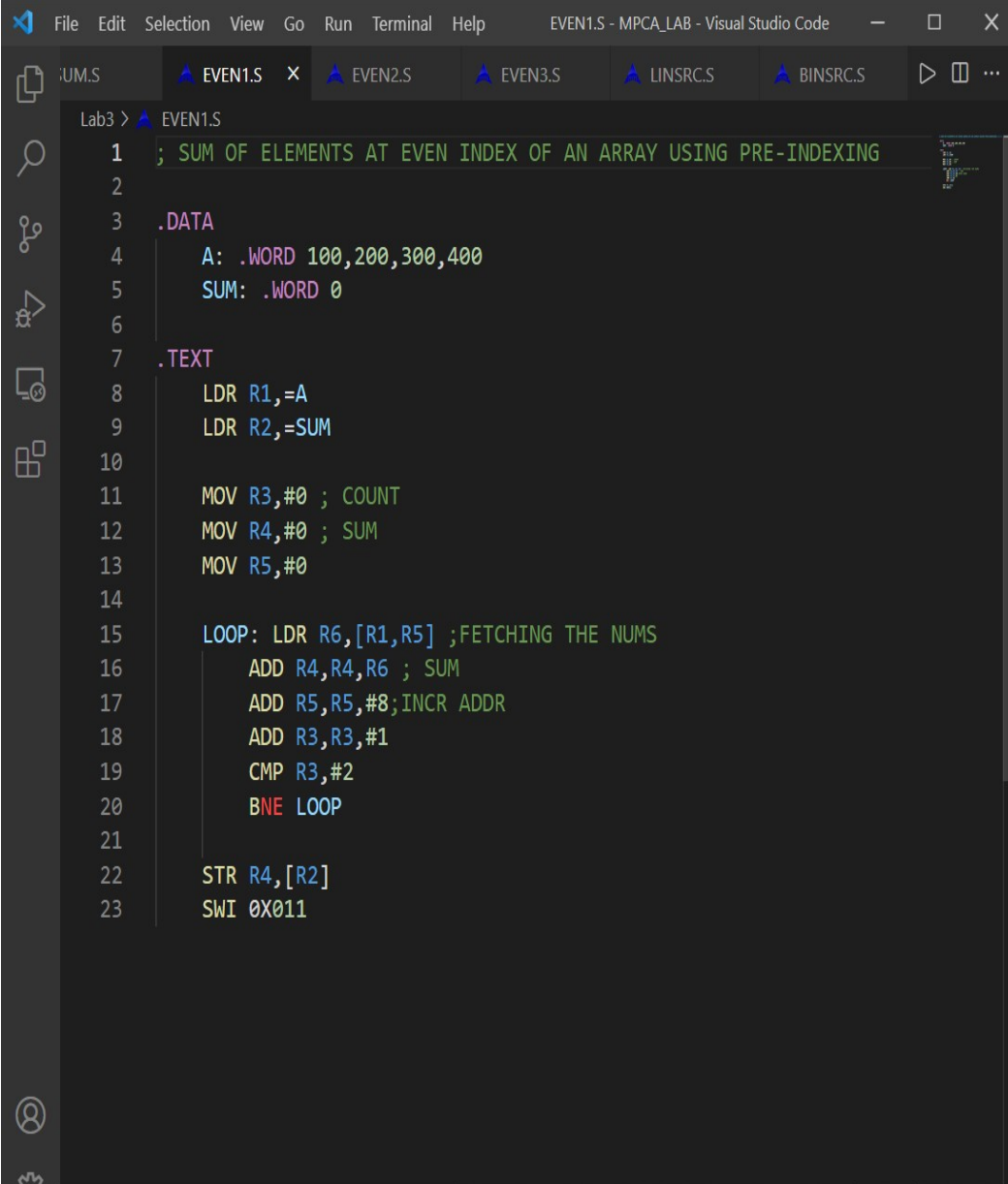
```
File Edit Selection View Go Run ... SUM3.S - MPCA_LAB - Visual St...
EXPLORER
MPCA_LAB
  Lab2
  Lab3
    a.exe
    BINSRC.S
    EVEN1.S
    EVEN2.S
    EVEN3.S
    GCD1.S
    GCD2.S
    LINSRC.S
    sel.cpp
    SUM_WB.S
    SUM.S
    SUM1.S
    SUM2.S
    SUM3.S
  Lab4
  Lab5
  Lab6
    DIAG_MATRIX.S
    DIGIT_SUM.S
    DUM.S
    MPCA_Week6 2022_...
    PARITY.S
    PES1UG20CS584_PH...
    SUBSTR.S
    SUM_POSTIVE.S
  Unit1
  hello.c
  OUTLINE

Lab3 > SUM3.S
1  ; SUM OF ELEMENTS OF AN ARRAY USING PRE-INDEXING
2
3  .DATA
4      A: .WORD 100,200,300,400,500,600
5      SUM: .WORD 0
6
7  .TEXT
8      LDR R1,=A
9      LDR R2,=SUM
10
11     MOV R3,#0 ; COUNT
12     MOV R4,#0 ; SUM
13     MOV R5,#4
14     SUB R1,R1,#4
15
16     LOOP: LDR R6,[R1,R5]!;FETCHING THE NUMS
17           ADD R4,R4,R6 ; SUM
18           ADD R3,R3,#1
19           CMP R3,#6
20           BNE LOOP
21
22     STR R4,[R2]
23     SWI 0X011
```

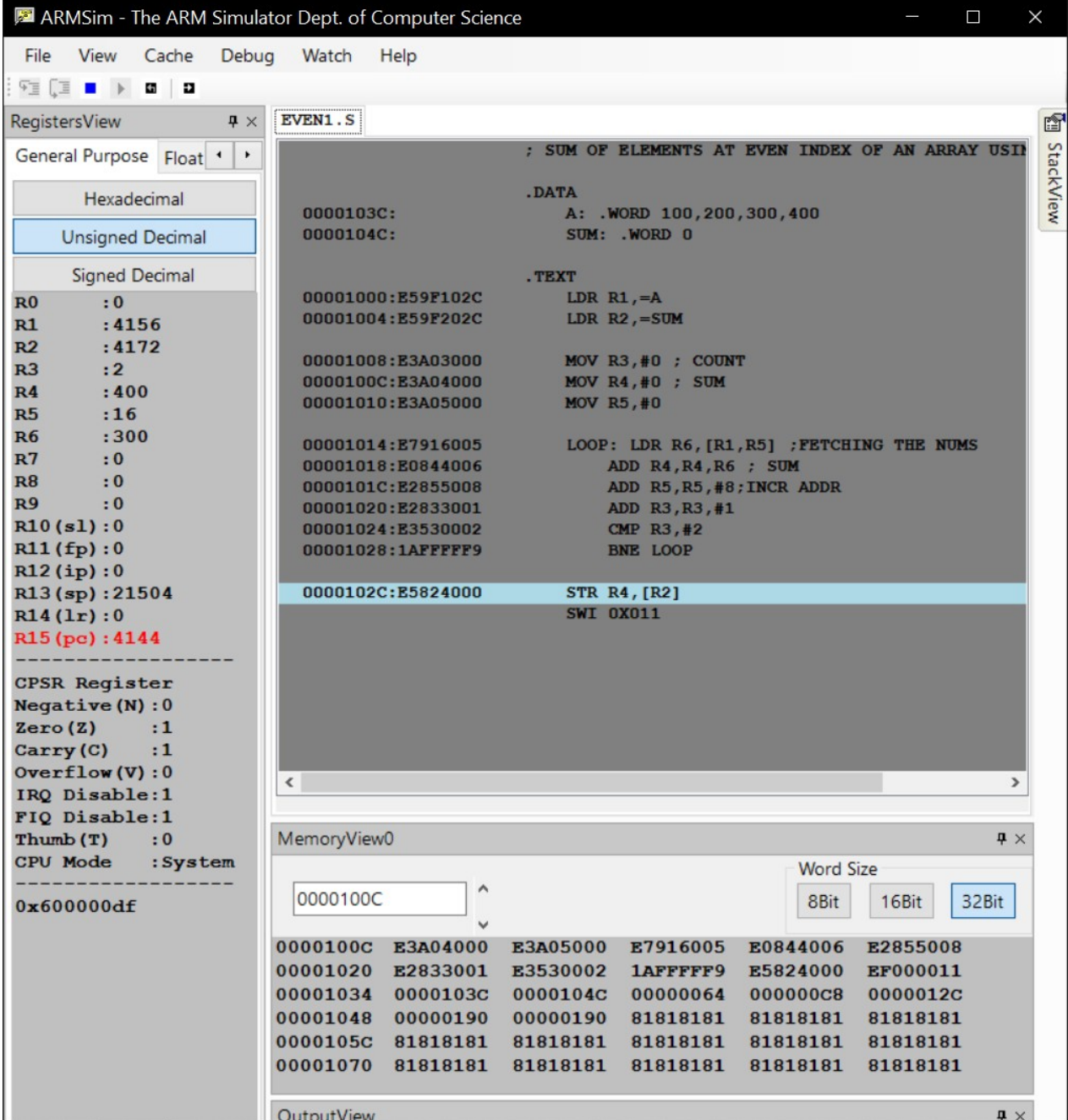


3. Write a program in ARM7TDMI-ISA to find the sum of N data items at alternate [ odd or **even** positions] locations in the memory. Store the result in the memory location.

a. Use Pre-indexing addressing mode

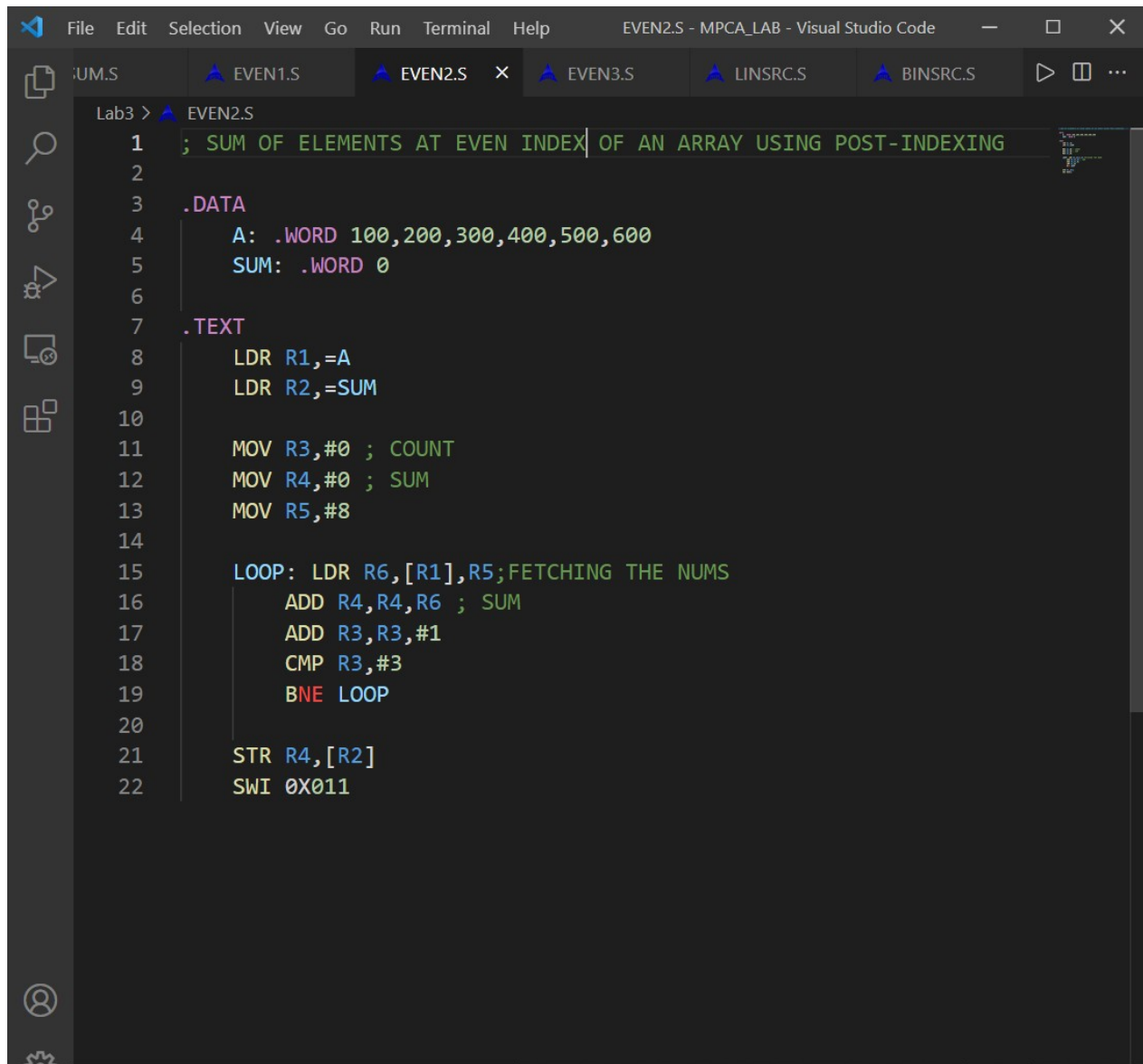


```
File Edit Selection View Go Run Terminal Help EVEN1.S - MPCA_LAB - Visual Studio Code
EVEN1.S x EVEN2.S EVEN3.S LINSRC.S BINSRC.S
Lab3 > EVEN1.S
1 ; SUM OF ELEMENTS AT EVEN INDEX OF AN ARRAY USING PRE-INDEXING
2
3 .DATA
4 A: .WORD 100,200,300,400
5 SUM: .WORD 0
6
7 .TEXT
8 LDR R1,=A
9 LDR R2,=SUM
10
11 MOV R3,#0 ; COUNT
12 MOV R4,#0 ; SUM
13 MOV R5,#0
14
15 LOOP: LDR R6,[R1,R5] ;FETCHING THE NUMS
16 ADD R4,R4,R6 ; SUM
17 ADD R5,R5,#8;INCR ADDR
18 ADD R3,R3,#1
19 CMP R3,#2
20 BNE LOOP
21
22 STR R4,[R2]
23 SWI 0X011
```





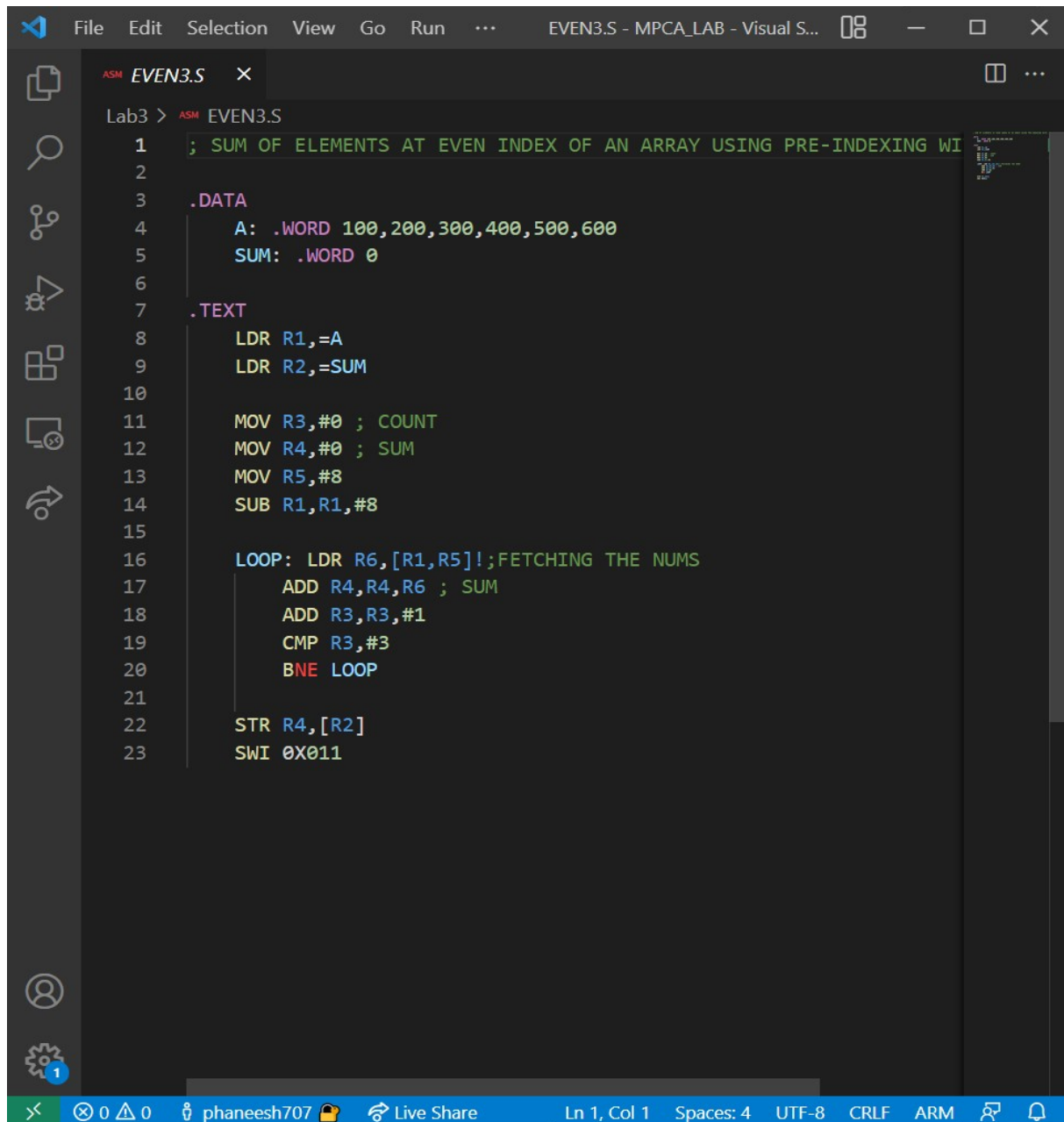
b. Use Post-Indexing addressing mode



```
1 ; SUM OF ELEMENTS AT EVEN INDEX OF AN ARRAY USING POST-INDEXING
2
3 .DATA
4     A: .WORD 100,200,300,400,500,600
5     SUM: .WORD 0
6
7 .TEXT
8     LDR R1,=A
9     LDR R2,=SUM
10
11     MOV R3,#0 ; COUNT
12     MOV R4,#0 ; SUM
13     MOV R5,#8
14
15     LOOP: LDR R6,[R1],R5;FETCHING THE NUMS
16           ADD R4,R4,R6 ; SUM
17           ADD R3,R3,#1
18           CMP R3,#3
19           BNE LOOP
20
21     STR R4,[R2]
22     SWI 0X011
```



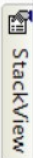
c. Use Auto-indexing addressing mode



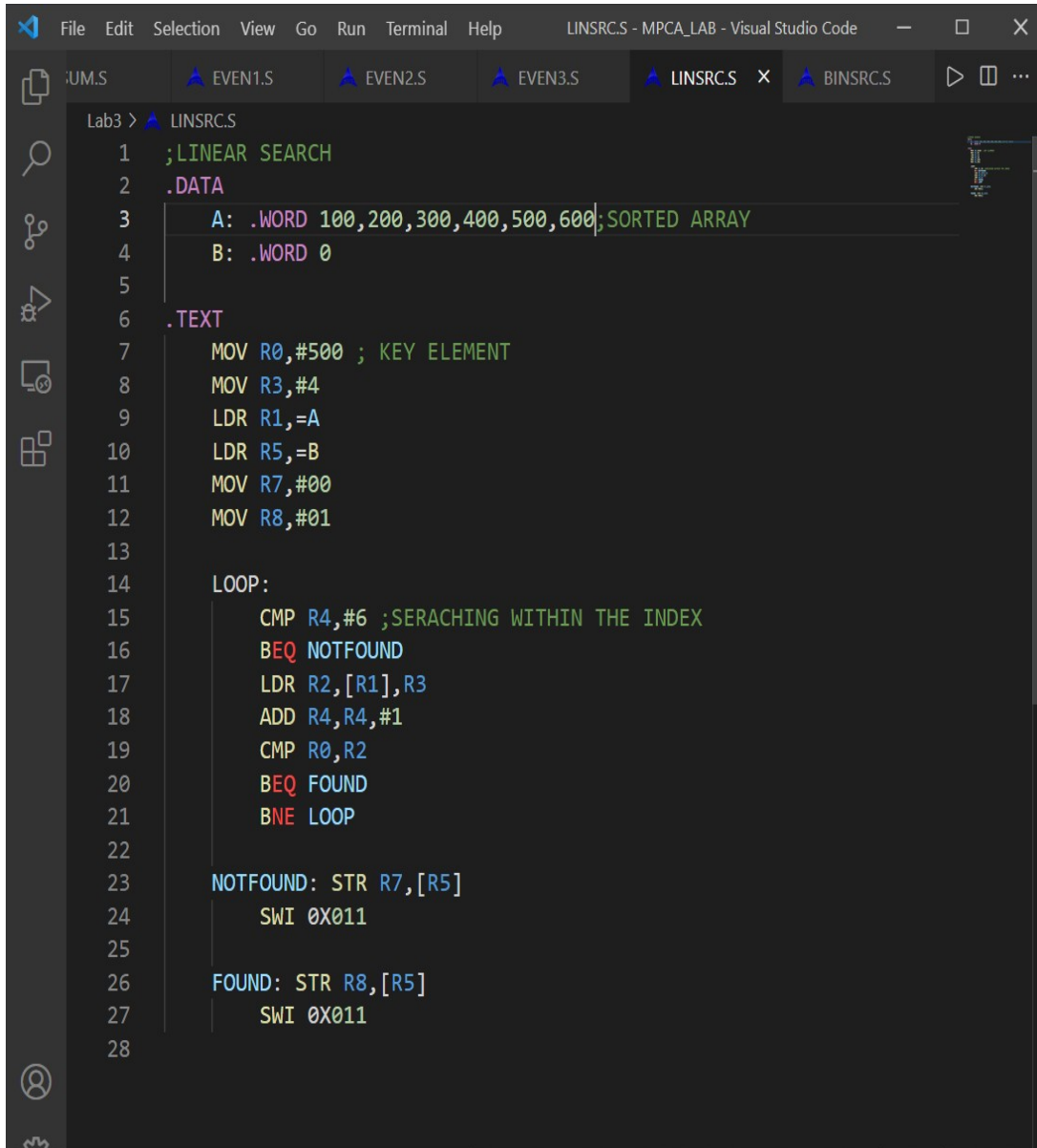
```
1 ; SUM OF ELEMENTS AT EVEN INDEX OF AN ARRAY USING PRE-INDEXING WI
2
3 .DATA
4 A: .WORD 100,200,300,400,500,600
5 SUM: .WORD 0
6
7 .TEXT
8 LDR R1,=A
9 LDR R2,=SUM
10
11 MOV R3,#0 ; COUNT
12 MOV R4,#0 ; SUM
13 MOV R5,#8
14 SUB R1,R1,#8
15
16 LOOP: LDR R6,[R1,R5]!;FETCHING THE NUMS
17 ADD R4,R4,R6 ; SUM
18 ADD R3,R3,#1
19 CMP R3,#3
20 BNE LOOP
21
22 STR R4,[R2]
23 SWI 0X011
```

Visual Studio Code interface showing the file 'EVEN3.S - MPCA\_LAB - Visual S...'. The status bar at the bottom indicates 'Ln 1, Col 1', 'Spaces: 4', 'UTF-8', 'CRLF', 'ARM', and a user profile 'phaneesh707'.





4. Write a program in ARM7TDMI-ISA to search for an element in an array. Store 00 if the search is unsuccessful and 01 if the search is successful in the register.
- a. Use Linear Search Technique



The screenshot shows a Visual Studio Code window with the title 'LINSRC.S - MPCA\_LAB - Visual Studio Code'. The editor displays an ARM assembly file named 'LINSRC.S' with the following code:

```
1 ;LINEAR SEARCH
2 .DATA
3     A: .WORD 100,200,300,400,500,600;SORTED ARRAY
4     B: .WORD 0
5
6 .TEXT
7     MOV R0,#500 ; KEY ELEMENT
8     MOV R3,#4
9     LDR R1,=A
10    LDR R5,=B
11    MOV R7,#00
12    MOV R8,#01
13
14    LOOP:
15        CMP R4,#6 ;SERACHING WITHIN THE INDEX
16        BEQ NOTFOUND
17        LDR R2,[R1],R3
18        ADD R4,R4,#1
19        CMP R0,R2
20        BEQ FOUND
21        BNE LOOP
22
23    NOTFOUND: STR R7,[R5]
24               SWI 0X011
25
26    FOUND: STR R8,[R5]
27            SWI 0X011
28
```

