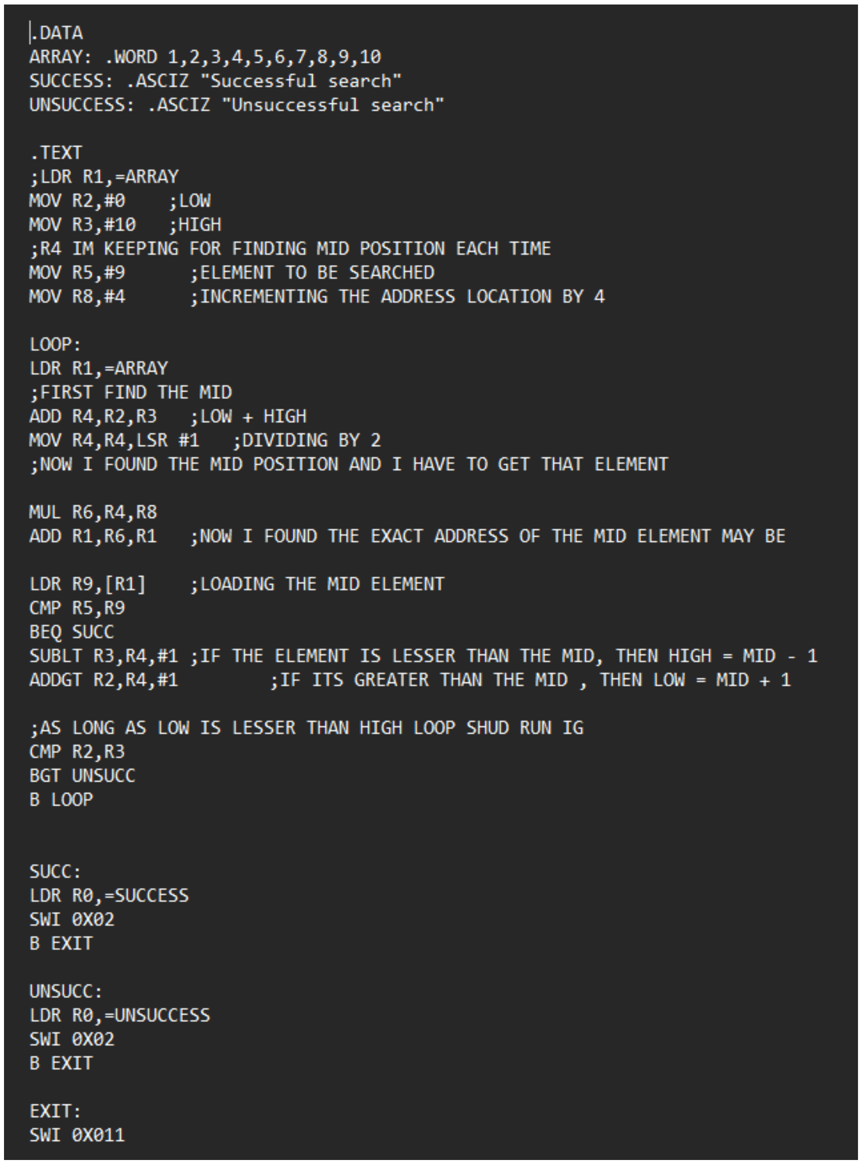
MPCA Theory Assignment

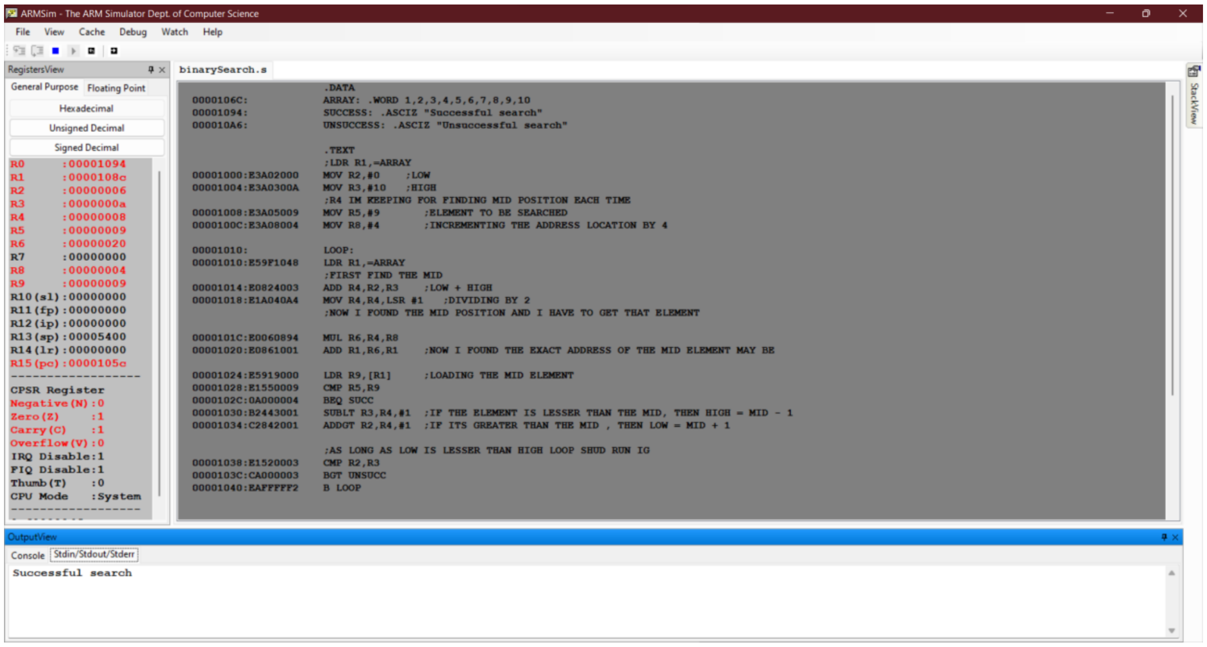
Name: Adithya M   SRN: PES1UG20CS621   Section: K

1) Binary Search



```
.DATA
ARRAY: .WORD 1,2,3,4,5,6,7,8,9,10
SUCCESS: .ASCIZ "Successful search"
UNSUCCESS: .ASCIZ "Unsuccessful search"

.TEXT
;LDR R1,=ARRAY
MOV R2,#0    ;LOW
MOV R3,#10   ;HIGH
;R4 IM KEEPING FOR FINDING MID POSITION EACH TIME
MOV R5,#9      ;ELEMENT TO BE SEARCHED
MOV R8,#4      ;INCREMENTING THE ADDRESS LOCATION BY 4

LOOP:
LDR R1,=ARRAY
;FIRST FIND THE MID
ADD R4,R2,R3   ;LOW + HIGH
MOV R4,R4,LSR #1   ;DIVIDING BY 2
;NOW I FOUND THE MID POSITION AND I HAVE TO GET THAT ELEMENT

MUL R6,R4,R8
ADD R1,R6,R1   ;NOW I FOUND THE EXACT ADDRESS OF THE MID ELEMENT MAY BE

LDR R9,[R1]    ;LOADING THE MID ELEMENT
CMP R5,R9
BEQ SUCC
SUBLT R3,R4,#1 ;IF THE ELEMENT IS LESSER THAN THE MID, THEN HIGH = MID - 1
ADDGT R2,R4,#1        ;IF ITS GREATER THAN THE MID , THEN LOW = MID + 1

;AS LONG AS LOW IS LESSER THAN HIGH LOOP SHUD RUN IG
CMP R2,R3
BGT UNSUCC
B LOOP


SUCC:
LDR R0,=SUCCESS
SWI 0X02
B EXIT

UNSUCC:
LDR R0,=UNSUCCESS
SWI 0X02
B EXIT

EXIT:
SWI 0X011
```

Output:

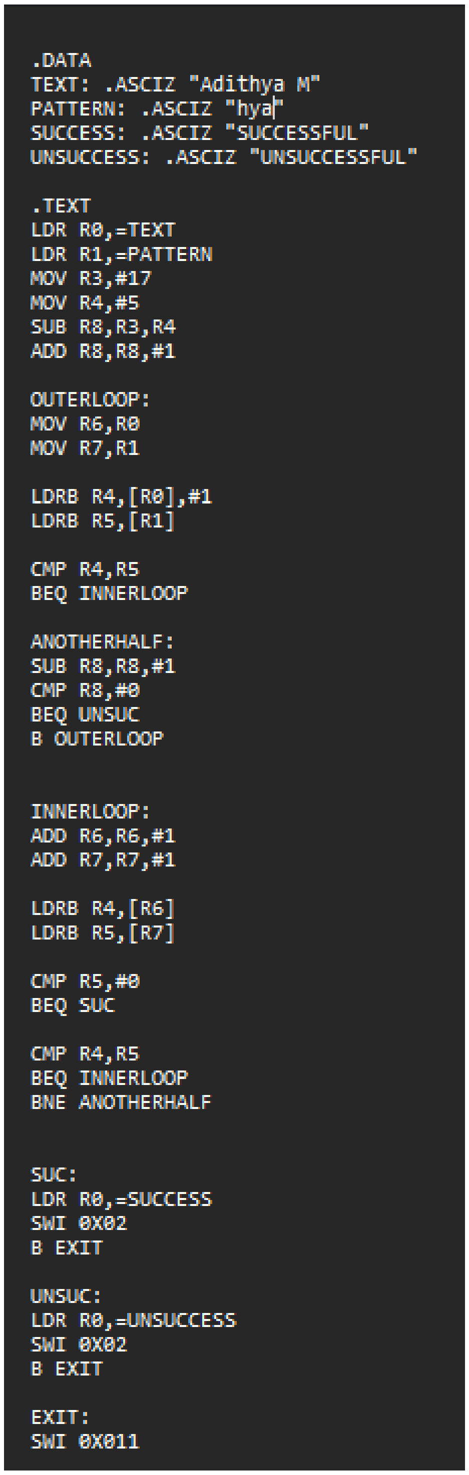

ARMSim - The ARM Simulator Dept. of Computer Science
File View Cache Debug Watch Help
RegistersView
General Purpose Floating Point
Hexadecimal
Unsigned Decimal
Signed Decimal
binarySearch.s
OutputView
Console Stdin/Stdout/Stderr
StackView

```
R0        :00001094
R1        :0000108c
R2        :00000006
R3        :0000000a
R4        :00000008
R5        :00000009
R6        :00000020
R7        :00000000
R8        :00000004
R9        :00000009
R10(sl)   :00000000
R11(fp)   :00000000
R12(ip)   :00000000
R13(sp)   :00005400
R14(lr)   :00000000
R15(pc)   :0000105c
------------------
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   :System
------------------
```

```
                         .DATA
0000106C:                ARRAY: .WORD 1,2,3,4,5,6,7,8,9,10
00001094:                SUCCESS: .ASCIZ "Successful search"
000010A6:                UNSUCCESS: .ASCIZ "Unsuccessful search"

                         .TEXT
                         ;LDR R1,=ARRAY
00001000:E3A02000    MOV R2,#0    ;LOW
00001004:E3A0300A    MOV R3,#10   ;HIGH
                     ;R4 IM KEEPING FOR FINDING MID POSITION EACH TIME
00001008:E3A05009    MOV R5,#9        ;ELEMENT TO BE SEARCHED
0000100C:E3A08004    MOV R8,#4        ;INCREMENTING THE ADDRESS LOCATION BY 4

00001010:            LOOP:
00001010:E59F1048    LDR R1,=ARRAY
                     ;FIRST FIND THE MID
00001014:E0824003    ADD R4,R2,R3     ;LOW + HIGH
00001018:E1A040A4    MOV R4,R4,LSR #1    ;DIVIDING BY 2
                     ;NOW I FOUND THE MID POSITION AND I HAVE TO GET THAT ELEMENT

0000101C:E0060894    MUL R6,R4,R8
00001020:E0861001    ADD R1,R6,R1     ;NOW I FOUND THE EXACT ADDRESS OF THE MID ELEMENT MAY BE

00001024:E5919000    LDR R9,[R1]      ;LOADING THE MID ELEMENT
00001028:E1550009    CMP R5,R9
0000102C:0A000004    BEQ SUCC
00001030:B2443001    SUBLT R3,R4,#1  ;IF THE ELEMENT IS LESSER THAN THE MID, THEN HIGH = MID - 1
00001034:C2842001    ADDGT R2,R4,#1  ;IF ITS GREATER THAN THE MID , THEN LOW = MID + 1

                     ;AS LONG AS LOW IS LESSER THAN HIGH LOOP SHUD RUN IG
00001038:E1520003    CMP R2,R3
0000103C:CA000003    BGT UNSUCC
00001040:EAFFFFF2    B LOOP
```

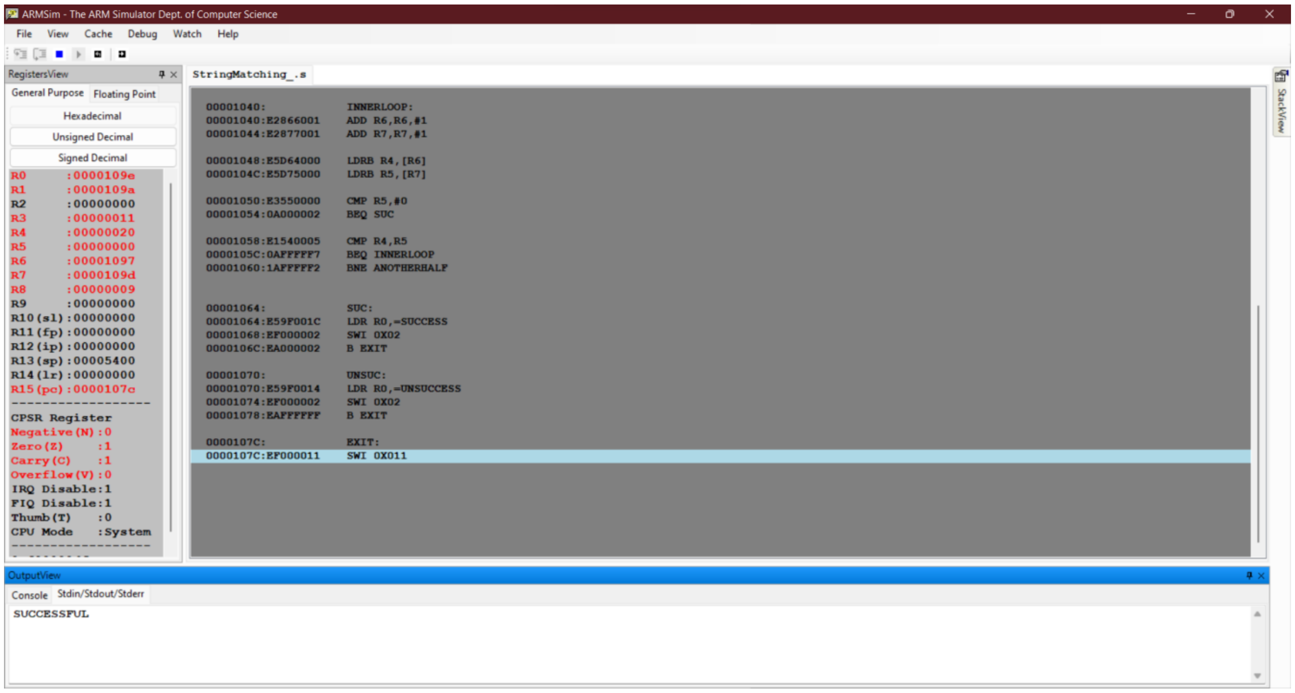
```
Successful search
```

2) String Matching



```
.DATA
TEXT: .ASCIZ "Adithya M"
PATTERN: .ASCIZ "hya"
SUCCESS: .ASCIZ "SUCCESSFUL"
UNSUCCESS: .ASCIZ "UNSUCCESSFUL"

.TEXT
LDR R0,=TEXT
LDR R1,=PATTERN
MOV R3,#17
MOV R4,#5
SUB R8,R3,R4
ADD R8,R8,#1

OUTERLOOP:
MOV R6,R0
MOV R7,R1

LDRB R4,[R0],#1
LDRB R5,[R1]

CMP R4,R5
BEQ INNERLOOP

ANOTHERHALF:
SUB R8,R8,#1
CMP R8,#0
BEQ UNSUC
B OUTERLOOP


INNERLOOP:
ADD R6,R6,#1
ADD R7,R7,#1

LDRB R4,[R6]
LDRB R5,[R7]

CMP R5,#0
BEQ SUC

CMP R4,R5
BEQ INNERLOOP
BNE ANOTHERHALF


SUC:
LDR R0,=SUCCESS
SWI 0X02
B EXIT

UNSUC:
LDR R0,=UNSUCCESS
SWI 0X02
B EXIT

EXIT:
SWI 0X011
```

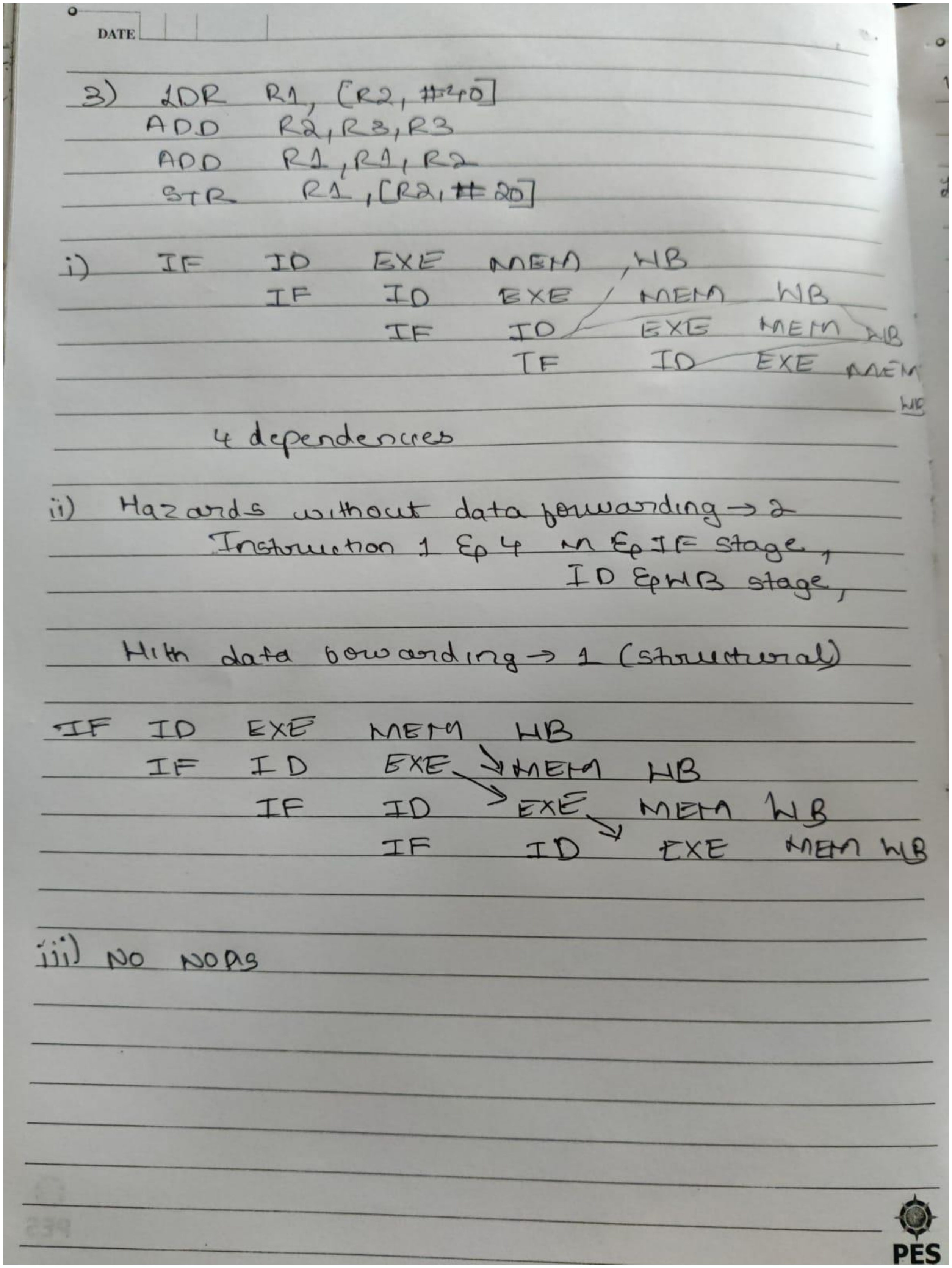
Output



```
R0       :0000109e
R1       :0000109a
R2       :00000000
R3       :00000011
R4       :00000020
R5       :00000000
R6       :00001097
R7       :0000109d
R8       :00000009
R9       :00000000
R10(sl)  :00000000
R11(fp)  :00000000
R12(ip)  :00000000
R13(sp)  :00005400
R14(lr)  :00000000
R15(pc)  :0000107c
-------------------
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   :System
-------------------
```

```
00001040:              INNERLOOP:
00001040:E2866001      ADD R6,R6,#1
00001044:E2877001      ADD R7,R7,#1

00001048:E5D64000      LDRB R4,[R6]
0000104C:E5D75000      LDRB R5,[R7]

00001050:E3550000      CMP R5,#0
00001054:0A000002      BEQ SUC

00001058:E1540005      CMP R4,R5
0000105C:0AFFFFF7      BEQ INNERLOOP
00001060:1AFFFFF2      BNE ANOTHERHALF


00001064:              SUC:
00001064:E59F001C      LDR R0,=SUCCESS
00001068:EF000002      SWI 0X02
0000106C:EA000002      B EXIT

00001070:              UNSUC:
00001070:E59F0014      LDR R0,=UNSUCCESS
00001074:EF000002      SWI 0X02
00001078:EAFFFFFF      B EXIT

0000107C:              EXIT:
0000107C:EF000011      SWI 0X011
```

```
SUCCESSFUL
```



3) LDR R1, [R2, #40]
ADD R2, R3, R3
ADD R1, R1, R2
STR R1, [R2, #20]

i)
```
IF   ID   EXE  MEM  WB
     IF   ID   EXE  MEM  WB
          IF   ID   EXE  MEM  WB
               IF   ID   EXE  MEM
                                  WB
```

4 dependencies

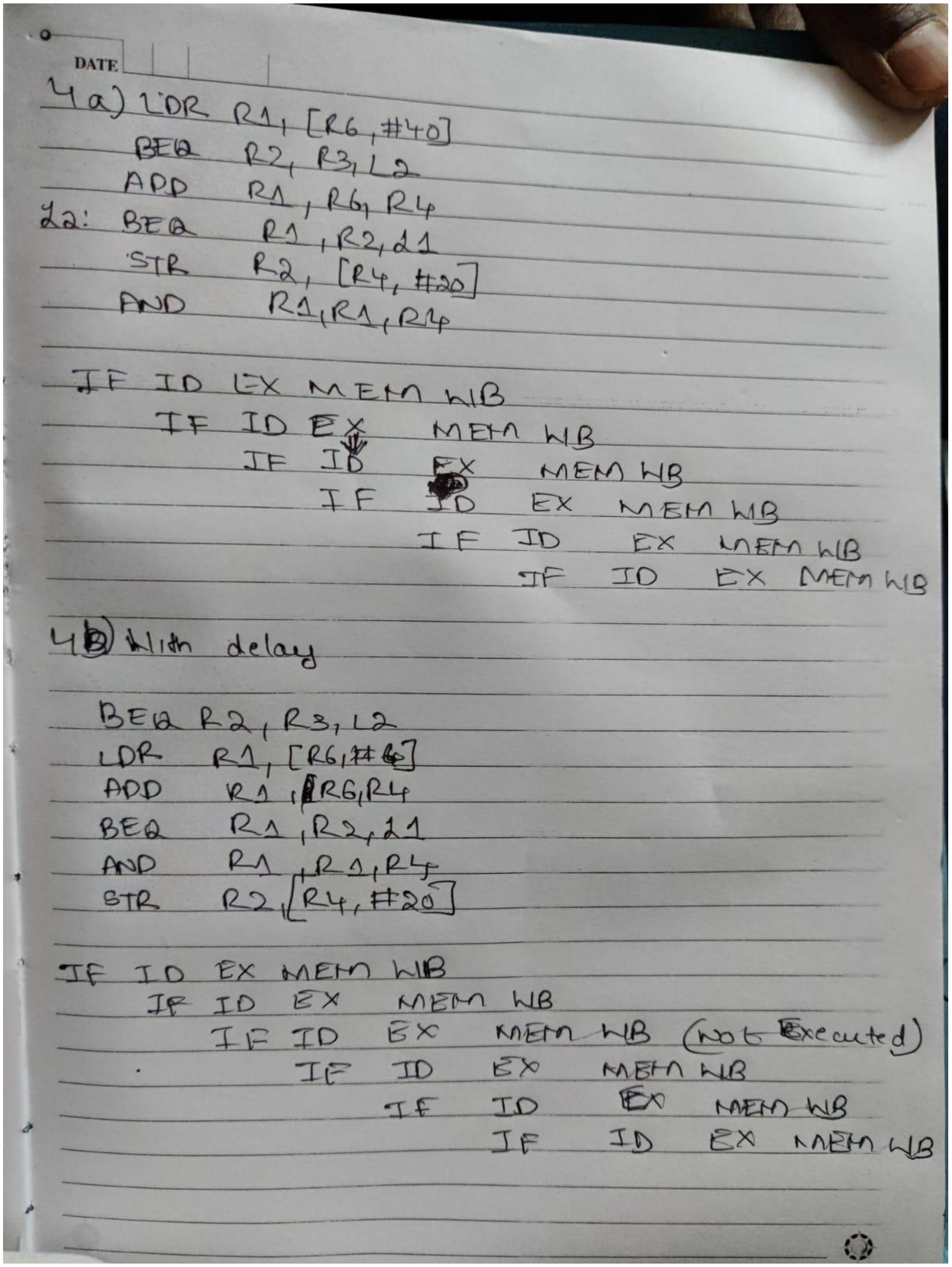
ii) Hazards without data forwarding → 2
Instruction 1 & 4 in & IF stage,
ID & WB stage,

With data forwarding → 1 (structural)

```
IF   ID   EXE  MEM  WB
     IF   ID   EXE  MEM  WB
          IF   ID   EXE  MEM  WB
               IF   ID   EXE  MEM  WB
```

iii) NO NOPS

3)



DATE

```
4a) LDR   R1, [R6, #40]
    BEQ   R2, R3, L2
    ADD   R1, R6, R4
L2: BEQ   R1, R2, L1
    STR   R2, [R4, #20]
    AND   R1, R1, R4

IF ID EX MEM WB
   IF ID EX    MEM WB
      IF ID    EX   MEM WB
         IF    ID   EX   MEM WB
               IF   ID   EX   MEM WB
                    IF   ID   EX   MEM WB
```

4B) With delay

```
BEQ R2, R3, L2
LDR  R1, [R6, #40]
ADD  R1, R6, R4
BEQ  R1, R2, L1
AND  R1, R1, R4
STR  R2, [R4, #20]

IF ID EX MEM WB
   IF ID EX  MEM WB
      IF ID  EX   MEM WB (not Executed)
         IF  ID   EX   MEM WB
             IF   ID   EX   MEM WB
                  IF   ID   EX   MEM WB
```

4)