VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Computer Science and Engineering
Fall Semester 2024-2025

Student Name: DHRUV AGRAWAL Reg No: 22BCT0059 Email: dhruv.agrawal2022a@vitstudent.ac.in	
Faculty Name: **DHRUV AGRAWAL**	
Subject Code with Name: **BCSE204P – Micro-Controller and Micro-Processor**	
Assessment No.:	01
Date of Submission:	09/02/2025

1.1 ↳ In lab Mold

(1) Create new folder.

(2) μ vision

(3) project -> new μVision Project
select folder -> file name

Legacy - Device
P89V51RD2

New file.

] creation of Project

(4) Program

```
ORG 0000H   — memory from where it will start
MOV A,  #05H              # means 05 data
         ↑                     else address
MOV R0, #05H              # is used only in micro
ADD A,  R0                            controller
END
```

(5) Save

- .asm extension

(6) Source Group
add existing file
build - to check errors.
debug.

(7) In micro controller - by default stack Pointer (SP) value is 07.

PSW - to check status of result
(Program status window) (even Parity / odd Parity)

(8) can execute stepwise or complete

continue 1.1a

Aim : To write, perform, execute assembly language program for 8051 microcontroller for addition of 2 8-bit numbers and store result in accumulator (A)

Software used : KIEL µVISION 6.0

Program :

```
ORG 0000H          ; set the Program starting address
                     to 0000H (hexadecimal)

MOV A, #05H        ; Move the immediate value 05H
                     into the accumulator

MOV R0, #05H       ; Move the immediate value 05H
                     into the register R0

ADD A, R0          ; Add the value in R0 to
                     accumulator

H: SJMP H          ; Create an infinite loop to halt
                     the program

END                ; END of the program
```

Expected Output : 05H
(Theoretical)

in binary

$$\begin{array}{r} 0000\ 0101 \\ +\ 0000\ 0101 \\ \hline 0000\ 1010 \\ \hline \end{array}$$

→ in hexadecimal 0A H

Actual / Practical Output

=> 0AH

Carry bit = 0

AC = 0

OV = CY ⊕ CY'

= 0 ⊕ 0 = 0

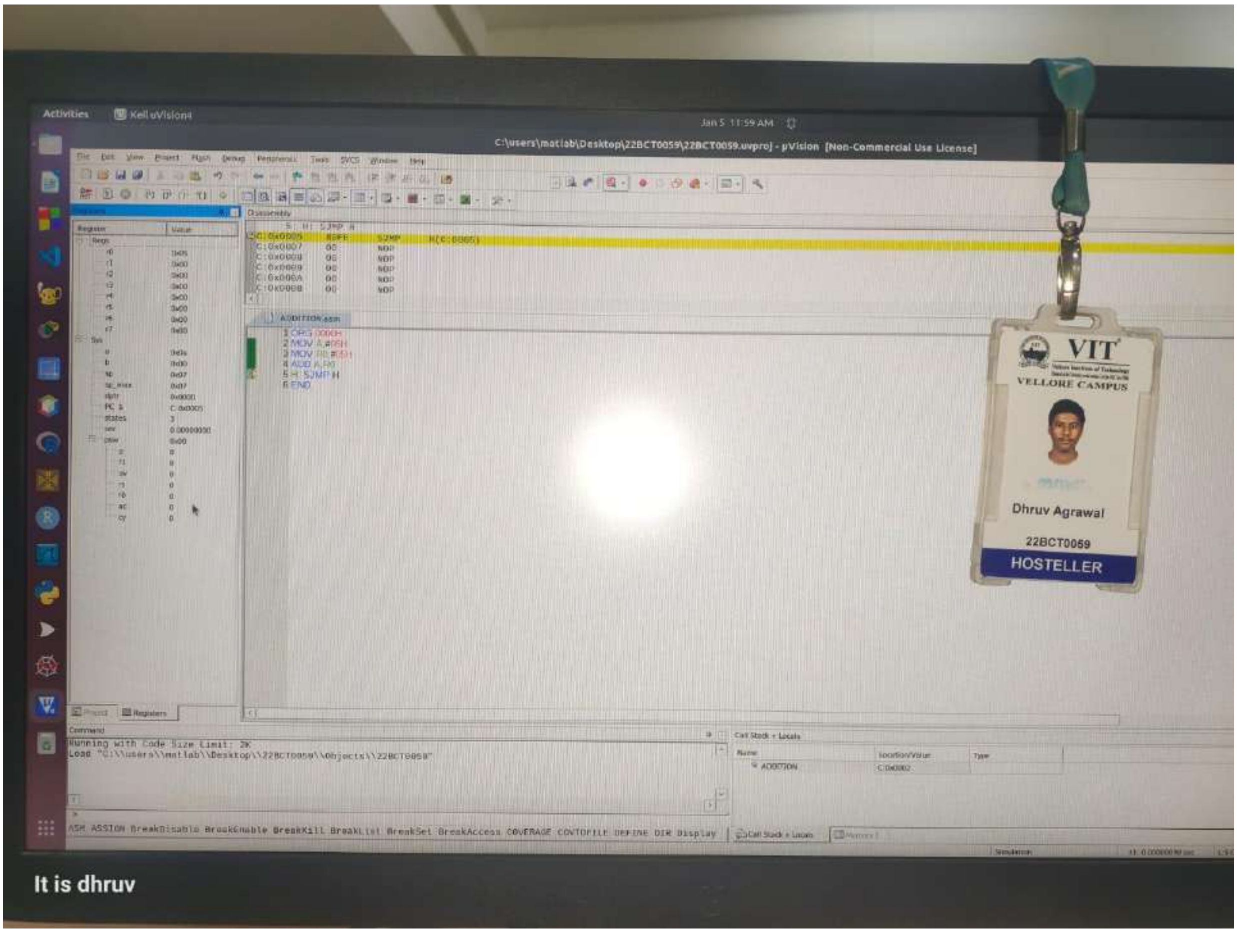
P (Parity Flag) = 0

$\longrightarrow$ ∴ PSW content (8-bit)

00000000

22BCT0059
DHRUV AGRAWAL

Lab Screenshot (with PSW):



```
Register    Value
Regs
  r0        0x05
  r1        0x00
  r2        0x00
  r3        0x00
  r4        0x00
  r5        0x00
  r6        0x00
  r7        0x00
Sys
  a         0x0a
  b         0x00
  sp        0x07
  sp_max    0x07
  dptr      0x0000
  PC $      C:0x0005
  states    3
  sec       0.00000300
  psw       0x00
    p       0
    f1      0
    ov      0
    rs      0
    f0      0
    ac      0
    cy      0
```

```
     5: H: SJMP H
C:0x0005    80FE     SJMP     H(C:0005)
C:0x0007    00       NOP
C:0x0008    00       NOP
C:0x0009    00       NOP
C:0x000A    00       NOP
C:0x000B    00       NOP
```

```
ORG 0000H
MOV A,#05H
MOV R0,#05H
ADD A,R0
H: SJMP H
END
```

```
Running with Code Size Limit: 2K
Load "C:\\users\\matlab\\Desktop\\22BCT0059\\Objects\\22BCT0059"
```

```
ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE COVTOFILE DEFINE DIR Display
```

Name	Location/Value	Type
ADDITION	C:0x0002	

It is dhruv

1.1 b

Aim: To write, perform, execute assembly language program for 8051 microcontroller for sub of 2 8-bit numbers and store result in accumulator

Software used : KIEL uVISION 6.0

Program:

```
ORG  0000H        ;set the Program starting
                   address to 0000H
MOV  A, #0AH      ;Move the immediate value
                   0AH into the accumulator
MOV  R0, #05H     ;Move the immediate value
                   05H in-to the register R0
SUBB A, R0        ;Subtract the value in
                   R0 from accumulator
                   with borrow
H:SJMP H          ;create an infinite loop to
                   halt the program
END               ;END the program
```

Expected : Output (Theoretical)

0AH -> 0000 1010 - binary
05H -> 0000 0101 - binary

0000 0101

Converting to hexadecimal -> 05H //

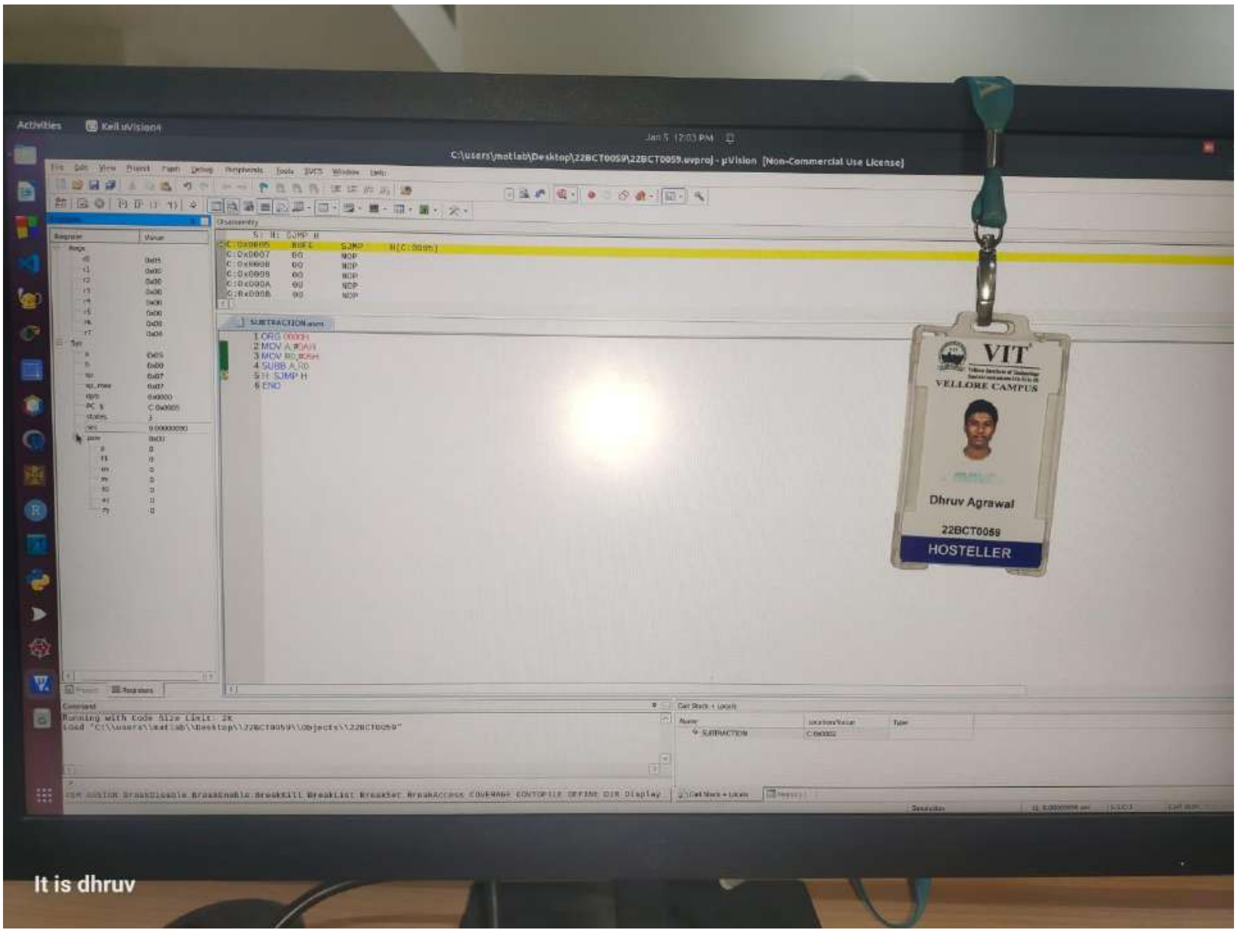
PSW content (8-bit) ⟶ CY = 0
AC = 0
OV = 0
P(Parity Flag) = 0

00000000

Actual / Practical output = 05H

22BCT0059
DHRUV AGRAWAL

Lab Screenshot (with PSW):



C:\users\matlab\Desktop\22BCT0059\22BCT0059.uvproj - µVision [Non-Commercial Use License]

```
    5: H: SJMP H
C:0x0005    80FE    SJMP    H(C:0005)
C:0x0007    00      NOP
C:0x0008    00      NOP
C:0x0009    00      NOP
C:0x000A    00      NOP
C:0x000B    00      NOP
```

SUBTRACTION.asm

```
ORG 0000H
MOV A,#0AH
MOV R0,#05H
SUBB A,R0
H: SJMP H
END
```

Register	Value
Regs	
r0	0x05
r1	0x00
r2	0x00
r3	0x00
r4	0x00
r5	0x00
r6	0x00
r7	0x00
Sys	
a	0x05
b	0x00
sp	0x07
sp_max	0x07
dptr	0x0000
PC $	C:0x0005
states	3
sec	0.00000300
psw	0x00
p	0
f1	0
ov	0
rs	0
f0	0
ac	0
cy	0

```
Running with Code Size Limit: 2K
Load "C:\\users\\matlab\\Desktop\\22BCT0059\\Objects\\22BCT0059"
```

1.1 C

Aim: To write, perform, execute the assembly language program for 8051 micro controller For multiplication of 2-8bit numbers and store result in Accumulator

Software used: KIEL uVISION 6.0

Program:

```
ORG 0000H        ; Set the program starting
                   address to 0000H (hexadecimal)
MOV A, #05H      ; Move the immediate value
                   05H into the accumulator
MOV B, #05H      ; move the immediate value
                   05H into the register B
MUL AB           ; multiply the value in A by
                   B.
H: SJMP H        ; create an infinite loop to halt
                   the program.
END              ; END of the program
```

Expected Output: (Theoretical)

```
05H →    0000 0101          5×5 = 25
05H →  × 0000 0101
        ----------          0001 1001
         0000 0101          16 + 8 + 1
        00000 000           25
       0 0000101
      0 0000000
      -----------
        0001 1001
```

Converting to hex → 19 H

PSW content (8-bit) → CY = 0
0000 0001

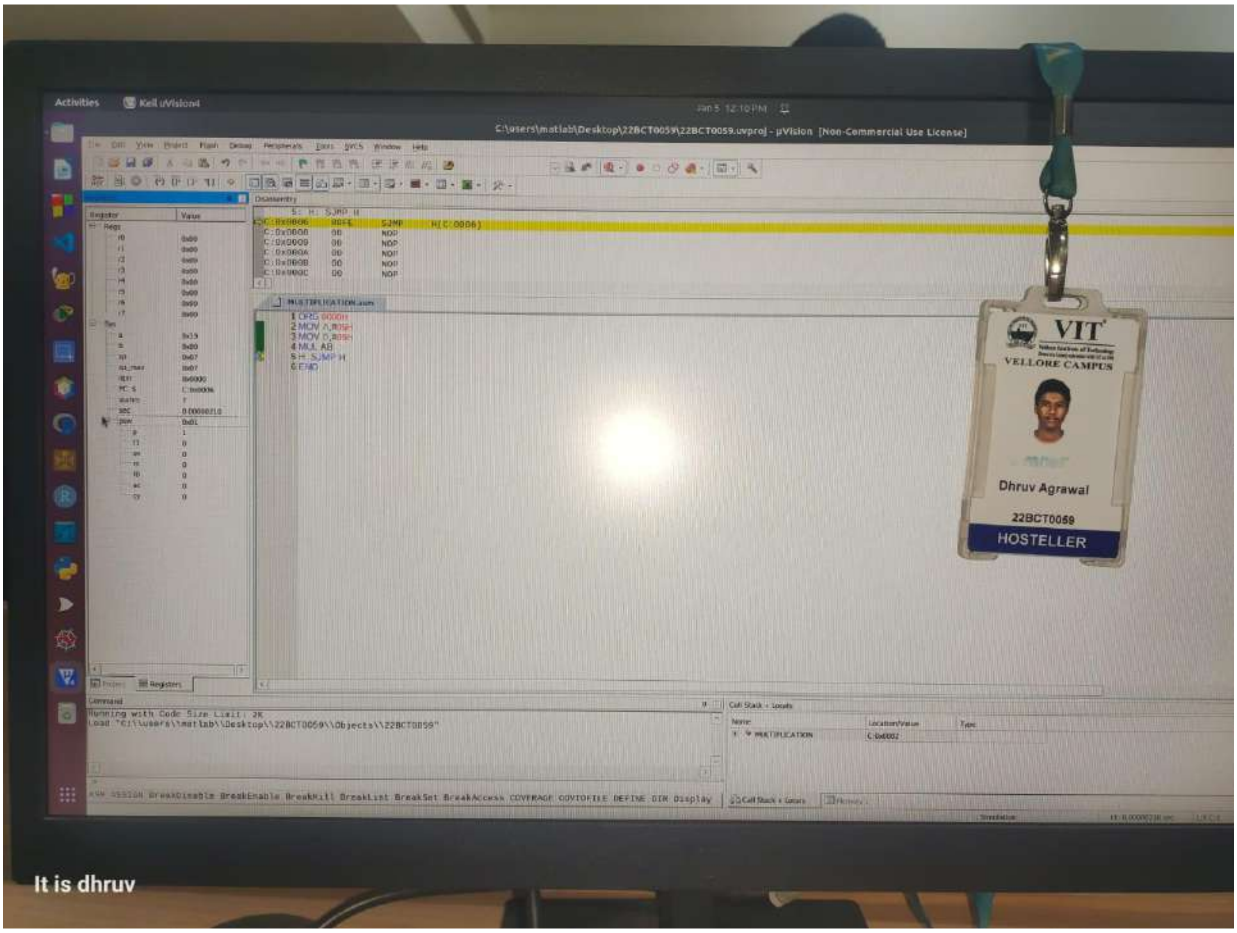
AC = 0

OV = 0 ⊕ 0 = 0

Parity (P) flag = 1

Actual/Practical = 19H

Lab Screenshot (with PSW):


Activities
Keil uVision4
C:\users\matlab\Desktop\22BCT0059\22BCT0059.uvproj - µVision [Non-Commercial Use License]
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
MULTIPLICATION.asm
Command
Call Stack + Locals
VIT
VELLORE CAMPUS
Dhruv Agrawal
22BCT0059
HOSTELLER
It is dhruv

```
ORG 0000H
MOV A,#05H
MOV B,#05H
MUL AB
H: SJMP H
END
```

```
    5: H: SJMP H
C:0x0006    80FE     SJMP     H(C:0006)
C:0x0008    00       NOP
C:0x0009    00       NOP
C:0x000A    00       NOP
C:0x000B    00       NOP
C:0x000C    00       NOP
```

```
Register        Value
Regs
  r0            0x00
  r1            0x00
  r2            0x00
  r3            0x00
  r4            0x00
  r5            0x00
  r6            0x00
  r7            0x00
Sys
  a             0x19
  b             0x00
  sp            0x07
  sp_max        0x07
  dptr          0x0000
  PC $          C:0x0006
  states        7
  sec           0.00000210
  psw           0x01
    p           1
    f1          0
    ov          0
    rs          0
    f0          0
    ac          0
    cy          0
```

```
Running with Code Size Limit: 2K
Load "C:\\users\\matlab\\Desktop\\22BCT0059\\Objects\\22BCT0059"
```

Name	Location/Value	Type
MULTIPLICATION	C:0x0002	

1.1 D

Aim: To write, perform, execute the assembly language program for 8051 micro controller for division of 2 8-bit numbers and store result in accumulator

Software used : KIEL uVISION 6.0

Program:

```
ORG 0000H       ; set the program starting
                  address to 0000H (hexadecimal)
MOV A, #20H     ; Move the immediate value
                  20H into the accumulator.
MOV B, #05H     ; Move the immediate value
                  05H into the register B.
DIV AB          ; Divide the value in A from
                  B and store result in A
H:SJMP H        ; create an infinite loop
                  to halt the program
END             ; END of the program
```

Expected Output (Theoretical) :- 20H → 0010 0000 — binary.
05H → 00000101 — binary.

```
        0110 -> Quotient
0101 ) 100000
       - 0
       ----
        1000
       - 101
       -----
        01010
       - 101
       ------
         0010
       -    0
       ------
         0010 -> Rem
       ------
```

Converting to HEX.

0110 → Q = 06 H — Stored in A

0010 → R = 02H — Stored in B.

//

PSW content (8-bit) → 00000000

CY = 0

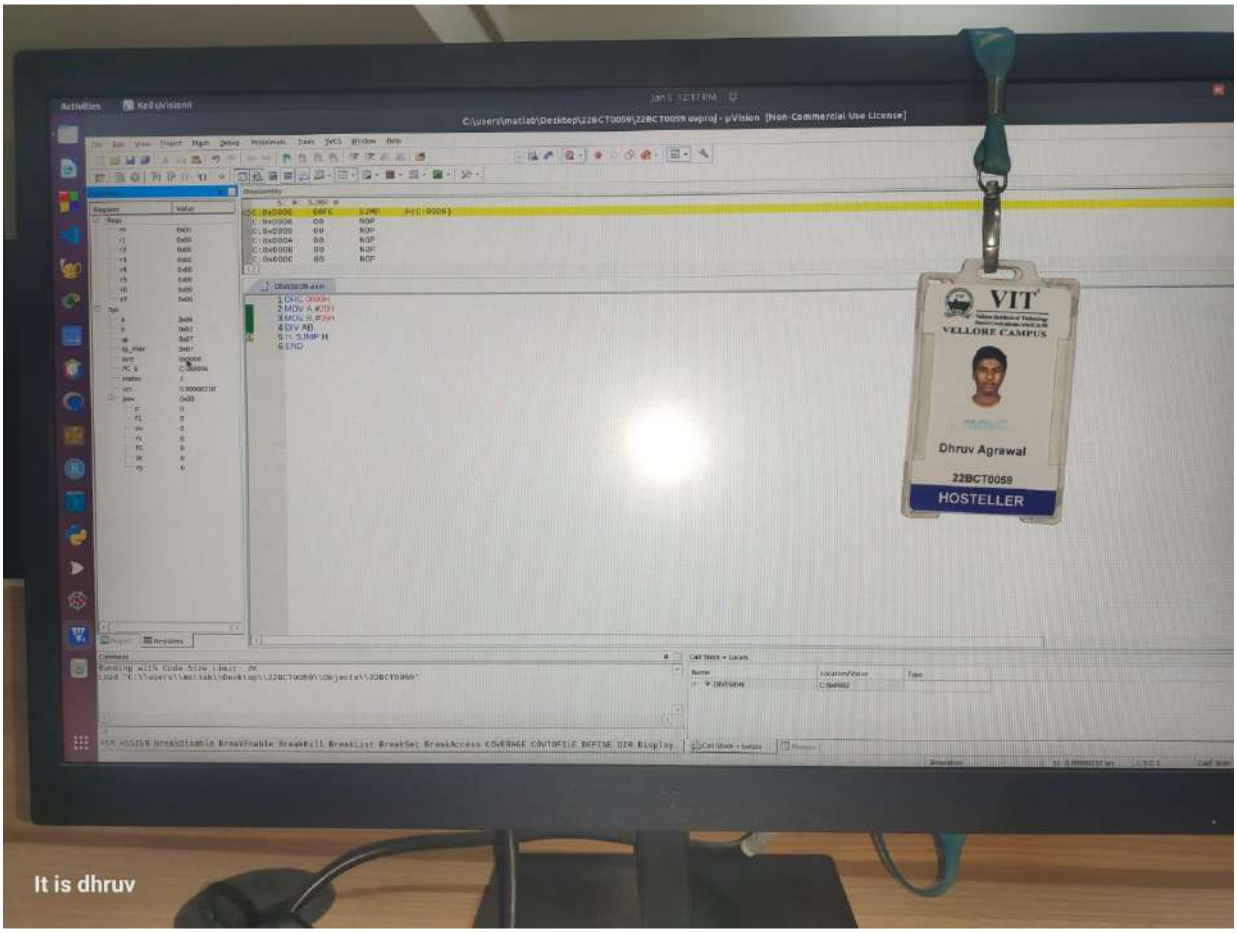
AC = 0

$OV = 0 \oplus 0 = 0$

Parity (P) = 0
Flag

Actual/Practical = 06H

Lab Screenshot (with PSW):



C:\users\matlab\Desktop\22BCT0059\22BCT0059.uvproj - µVision [Non-Commercial Use License]

```
C:0x0006    80FE    SJMP    H(C:0006)
C:0x0008    00      NOP
C:0x0009    00      NOP
C:0x000A    00      NOP
C:0x000B    00      NOP
C:0x000C    00      NOP
```

DIVISION.asm

```
ORG 0000H
MOV A,#[illegible]
MOV B,#[illegible]
DIV AB
H: SJMP H
END
```

1.2

Aim: Write and assemble a program to add the following data and then use the simulator to examine the cy flag

92 H, 23 H, 66 H, 87 H, F5 H

Software used : KEIL uVISION 6.0

Program :

```
      ORG 000H          ; set the program starting
                          address to 0000H (hexadecimal)
      MOV A, #92H       ; Move immediate value 92H
                          into the accumulator.
      MOV R0, #23H      ; Move immediate value 23H
                          into the register R0.
      ADD A, R0         ; Add the value of R0 into A
      JNC L1            ; Jump to L1 if no carry genrated
      INC R7            ; Increment R7
  L1: MOV R1, #66H      ; Move the immediate value 66H
                          into the register R1
      ADD A, R1         ; Add the value of R1 into A
      JNC L2            ; Jump to L2 if no carry genrated
      INC R7            ; Increment R7
  L2: MOV R2, #87H      ; Move the immediate value of 87H
                          into the register R2
      ADD A, R2         ; Add the value of R2 into A
      JNC L3            ; Jump to L3, if no carry genrated
      INC R7            ; Increment R7
  L3: MOV R3, #DF5H     ; Move the immediate value of F5H
                          into the register R3
      ADD A, R3         ; Add the value of R3 into A
      JNC L4            ; Jump to L4 if no carry genrated
      INC R7            ; Increment R7
  L4: END               ; End of the Program
```

Expected output :-
(Theoretical)

92H -> 10010010]- binary
23H -> + 00100011

10110101

no carry genrated ∴ R7 will remain 00H

10110101
+ 01100110 ⟶ 66H

1 00011011

carry.
genrated.
∴ Increment R7.
∴ R7 = 0+1 = 01H

00011011
+ 10000111 — 87H

10100010

no carry genrated ∴ R7 will remain 01H

10100010
+ 11110101 — F5H

1 10010111 -> Converting to hexadecimal

Ans 97H

carry genrated.
∴ Increment R7
R7 = 01H + 1 = 02H

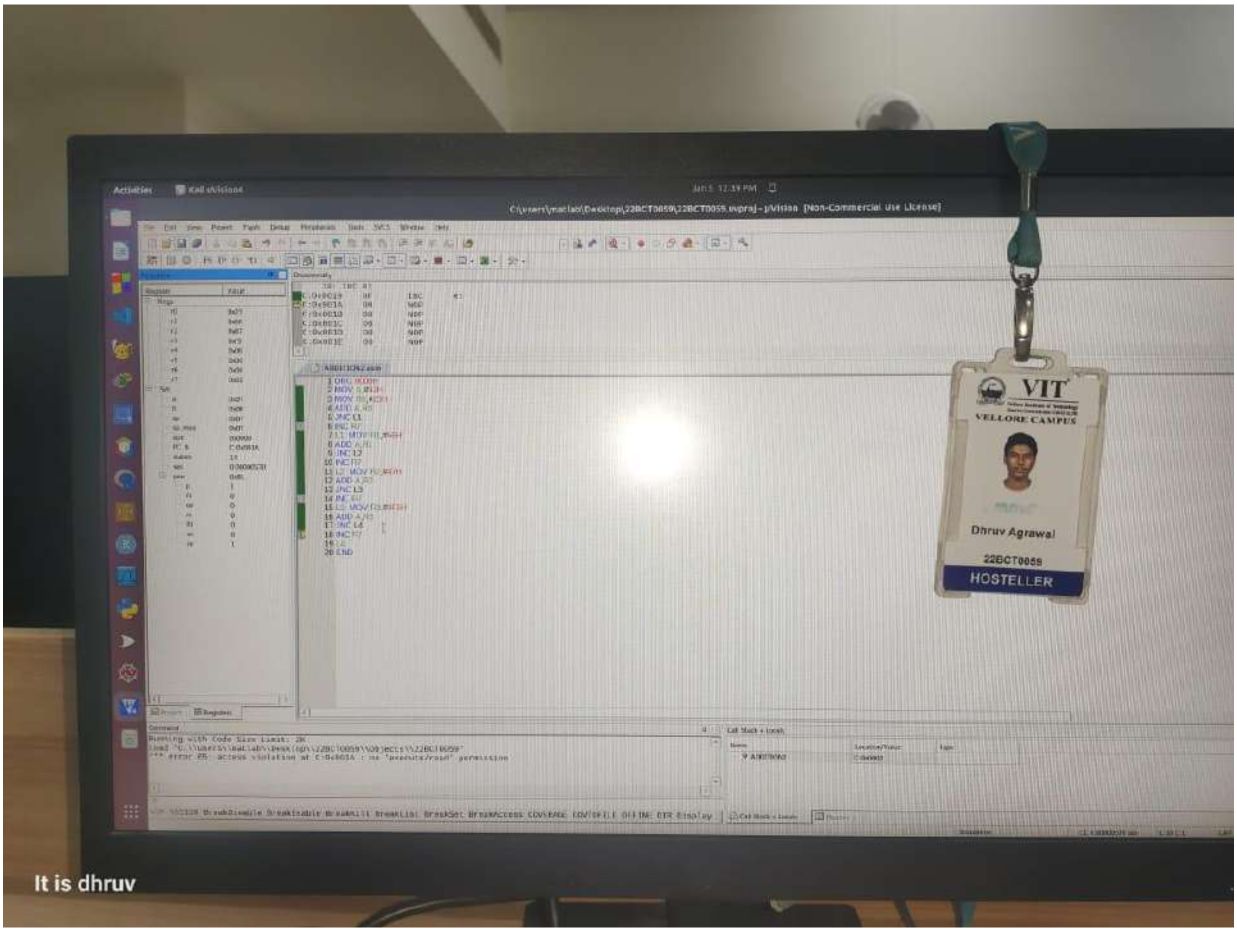
Practical
Actual output = 02H

PSW content
(8-bit)
10000001

-> CY = 1
AC = 0
OV = 1 $\oplus$ 1 = 0
Parity (P) = 1
Flag

Conclusion : The operations addition, subtraction, multiplication division, addn with carry are important arithmetic operations. to perform complex mathematical operation. verifying, Executing these functions is neccesary for developing efficient and accurate embedded/micro controller based systems.

Lab Screenshot (with PSW):



```
ORG 0000H
MOV A,#[illegible]H
MOV R0,#[illegible]H
ADD A,R0
JNC L1
INC R7
L1: MOV R1,#[illegible]H
ADD A,R1
JNC L2
INC R7
L2: MOV R2,#[illegible]H
ADD A,R2
JNC L3
INC R7
L3: MOV R3,#[illegible]H
ADD A,R3
JNC L4
INC R7
L4:
END
```

```
Running with Code Size Limit: 2K
*** error 65: access violation at C:0x001A : no 'execute/read' permission
```

1.3 A

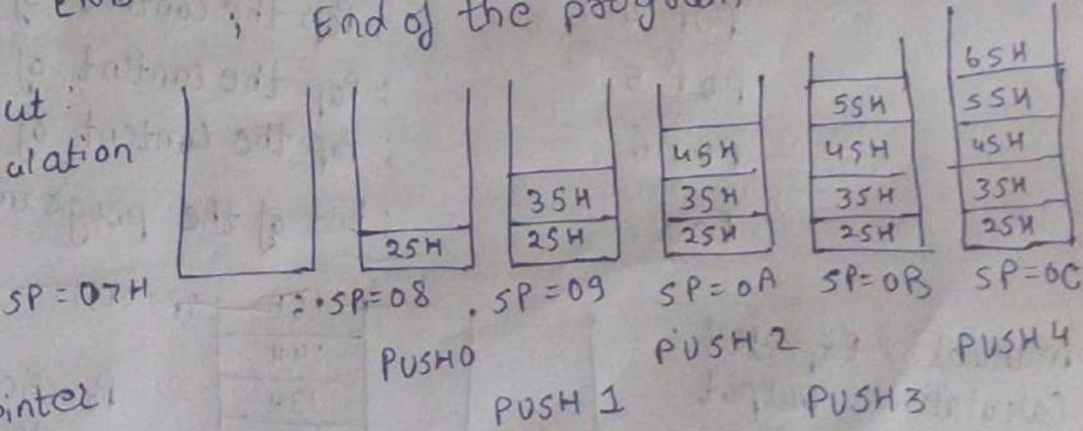
Aim: To write, perform and execute the assembly language program to load value into register R0 to R4 and then push it into stack. Examine the stack register values after execution of Program

Software used : Keil µVision 6.0

Program:

```
ORG 0000H ;  Program start address (origin)
mov R0;#25H ; Load value 25H in R0
mov R1, #35H ; Load value 35H in R1
mov R2, #45H ; Load value 45H in R2
mov R3, #55H ; Load value 55H in R3
mov R4, #65H ; Load value 65H in R4
PUSH 0 ;  Push value of R0 in stack
PUSH 1 ;  Push value of R1 in stack
PUSH 2 ;  Push value of R2 in stack
PUSH 3 ;  Push value of R3 in stack
PUSH 4 ;  Push value of R4 in stack
END    ;  End of the program
```

Expected output:
Manual Calculation

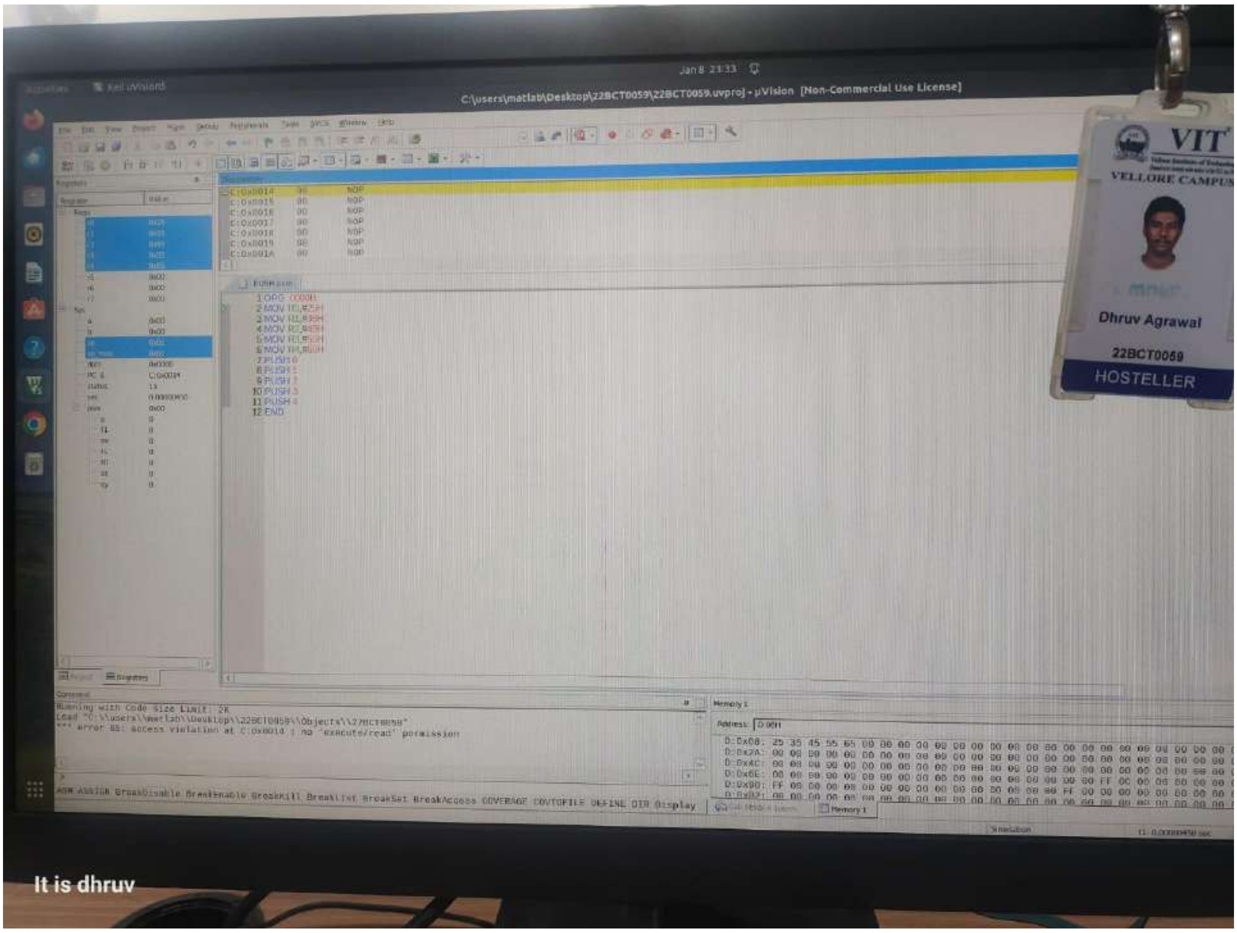


Final stack Pointer value is 0CH

Practical/ Actual Output : 0CH

22BCT0059
DHRUV AGRAWAL

Lab Screenshot (with PSW):



C:\users\matlab\Desktop\22BCT0059\22BCT0059.uvproj - µVision [Non-Commercial Use License]

```
C:0x0014    00    NOP
C:0x0015    00    NOP
C:0x0016    00    NOP
C:0x0017    00    NOP
C:0x0018    00    NOP
C:0x0019    00    NOP
C:0x001A    00    NOP
```

```
ORG 0000H
MOV R0,#25H
MOV R1,#35H
MOV R2,#45H
MOV R3,#55H
MOV R4,#65H
PUSH 0
PUSH 1
PUSH 2
PUSH 3
PUSH 4
END
```

```
Running with Code Size Limit: 2K
Load "C:\\users\\matlab\\Desktop\\22BCT0059\\Objects\\22BCT0059"
*** error 65: access violation at C:0x0014 : no 'execute/read' permission
```

```
Address: D:08H
D:0x08: 25 35 45 55 65 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
D:0x2A: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
D:0x4C: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
D:0x6E: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
D:0x90: FF 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF 00 00 00 00 00 00
D:0xB2: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF 00 00 00 00 00 00 00 00
```

ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE COVTOFILE DEFINE DIR Display
Memory 1
Simulation
t1: 0.00000450 sec
VIT
VELLORE CAMPUS
Dhruv Agrawal
22BCT0059
HOSTELLER
It is dhruv

1.3B

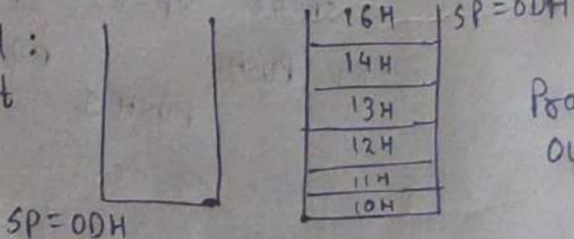
Aim: Write, perform and execute the assembly lang program to set SP = 0D. Then put diffrent value in each register from 08 to 0D, pop SP into R0 - R4

Software used: Keil µVision 6.0

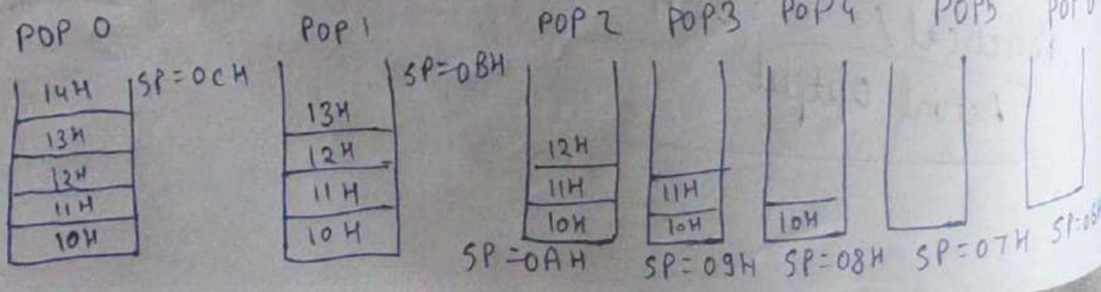
Program:

```
ORG 0000H        ; Program start address (origin)
mov SP, #0DH     ; move SP value to 0DH
mov 08H, #10H    ; move 10H in memory register location 08
mov 09H, #11H    ; move 11H in register 09H
mov 0AH, #12H    ; move 12H in register 0AH
mov 0BH, #13H    ; move 13H in register 0BH
mov 0CH, #14H    ; move 14H in register 0CH
mov 0DH, #16H    ; move 16H in register 0DH
POP 0            ; Pop the content of stack in R0
POP 1            ; Pop the content of stack in R1
POP 2            ; Pop the content of stack in R2
POP 3            ; Pop the content of stack in R3
POP 4            ; Pop the content of stack in R4
POP 5            ; Pop the content of stack in R5
POP 6            ; Pop the content of stack in R6
End.             ; End of the program
```

manual / Expected : Calculation output

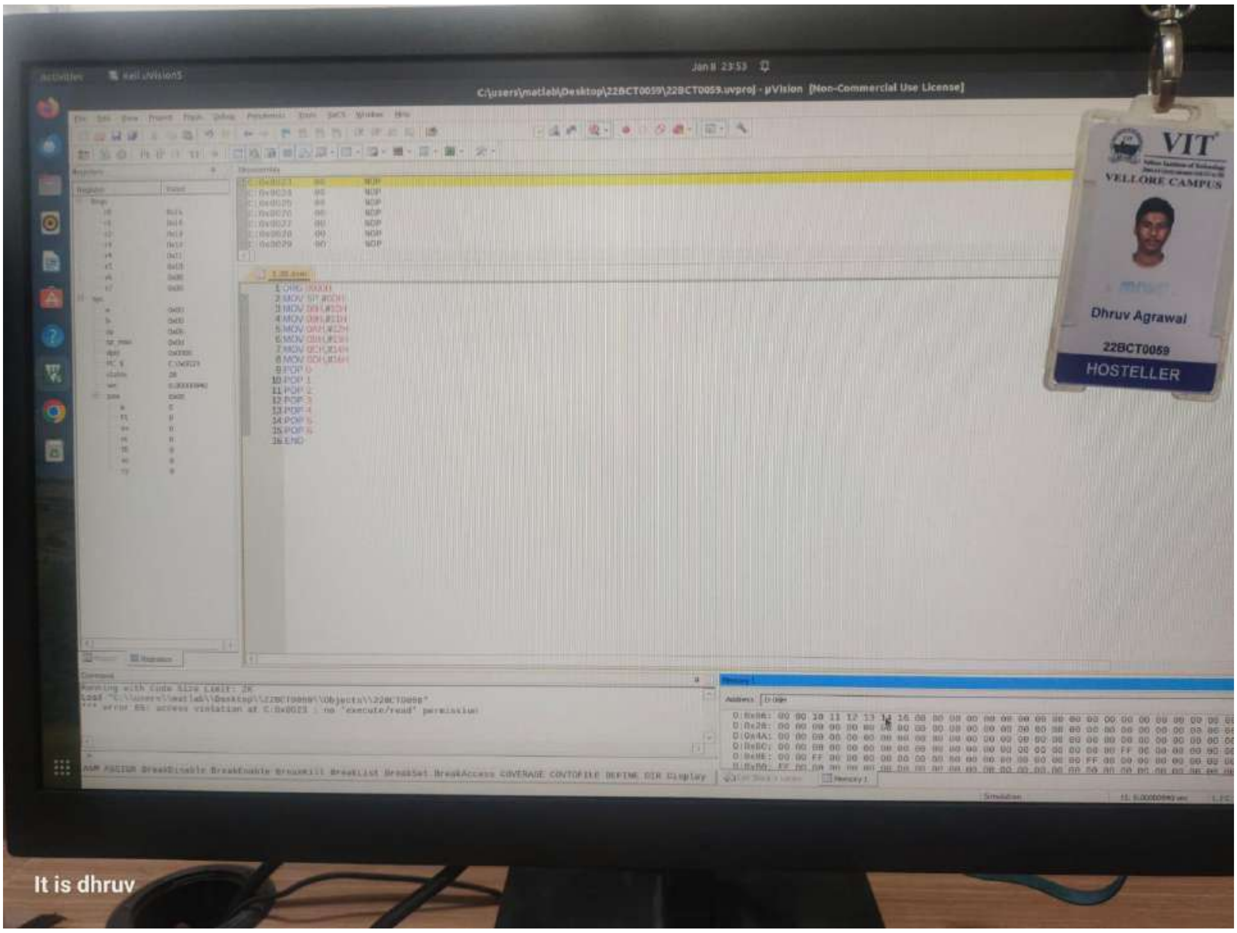


Practical output = 06H



22BCT0059
DHRUV AGRAWAL

Lab Screenshot (with PSW):



```
ORG 0000H
MOV SP,#0DH
MOV 08H,#10H
MOV 09H,#11H
MOV 0AH,#12H
MOV 0BH,#13H
MOV 0CH,#14H
MOV 0DH,#16H
POP 0
POP 1
POP 2
POP 3
POP 4
POP 5
POP 6
END
```

```
Running with Code Size Limit: 2K
Load "C:\\users\\matlab\\Desktop\\22BCT0059\\Objects\\22BCT0059"
*** error 65: access violation at C:0x0023 : no 'execute/read' permission
```

C:/users/matlab/Desktop/22BCT0059/22BCT0059.uvproj - µVision [Non-Commercial Use License]
VIT
VELLORE CAMPUS
Dhruv Agrawal
22BCT0059
HOSTELLER
It is dhruv

1.3 C.

Aim: Write, perform and execute assembly language program to load values into registers R0 to R4, then push it into stack, pop them back

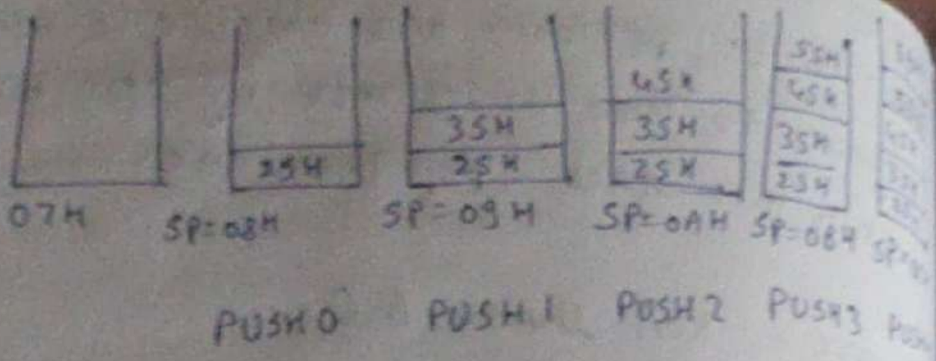
Software used : Keil µVision 6.0

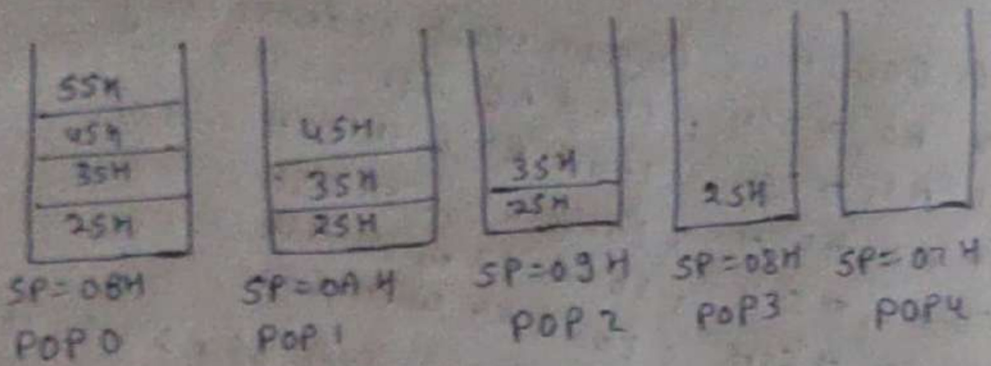
Program:

```
ORG 0000H      ; Program start address (origin)
mov R0, #25H   ; Load R0 with 25H
mov R1, #35H   ; Load R1 with 35H
mov R2, #45H   ; Load R2 with 45H
mov R3, #55H   ; Load R3 with 55H
mov R4, #65H   ; Load R4 with 65H
PUSH 0         ; Push the content of R0 into stack
PUSH 1         ; Push the content of R1 into stack
PUSH 2         ; Push the content of R2 into stack
PUSH 3         ; Push the content of R3 into stack
PUSH 4         ; Push the content of R4 into stack
POP 10H        ; Pop the content of stack in 10H
POP 11H        ; Pop the content of stack in 11H
POP 12H        ; Pop the content of stack in 12H
POP 13H        ; Pop the content of stack in 13H
POP 14H        ; Pop the content of stack in 14H
END            ; End of the program
```

Practical Output = 07H

manual calculations:
Expected Output

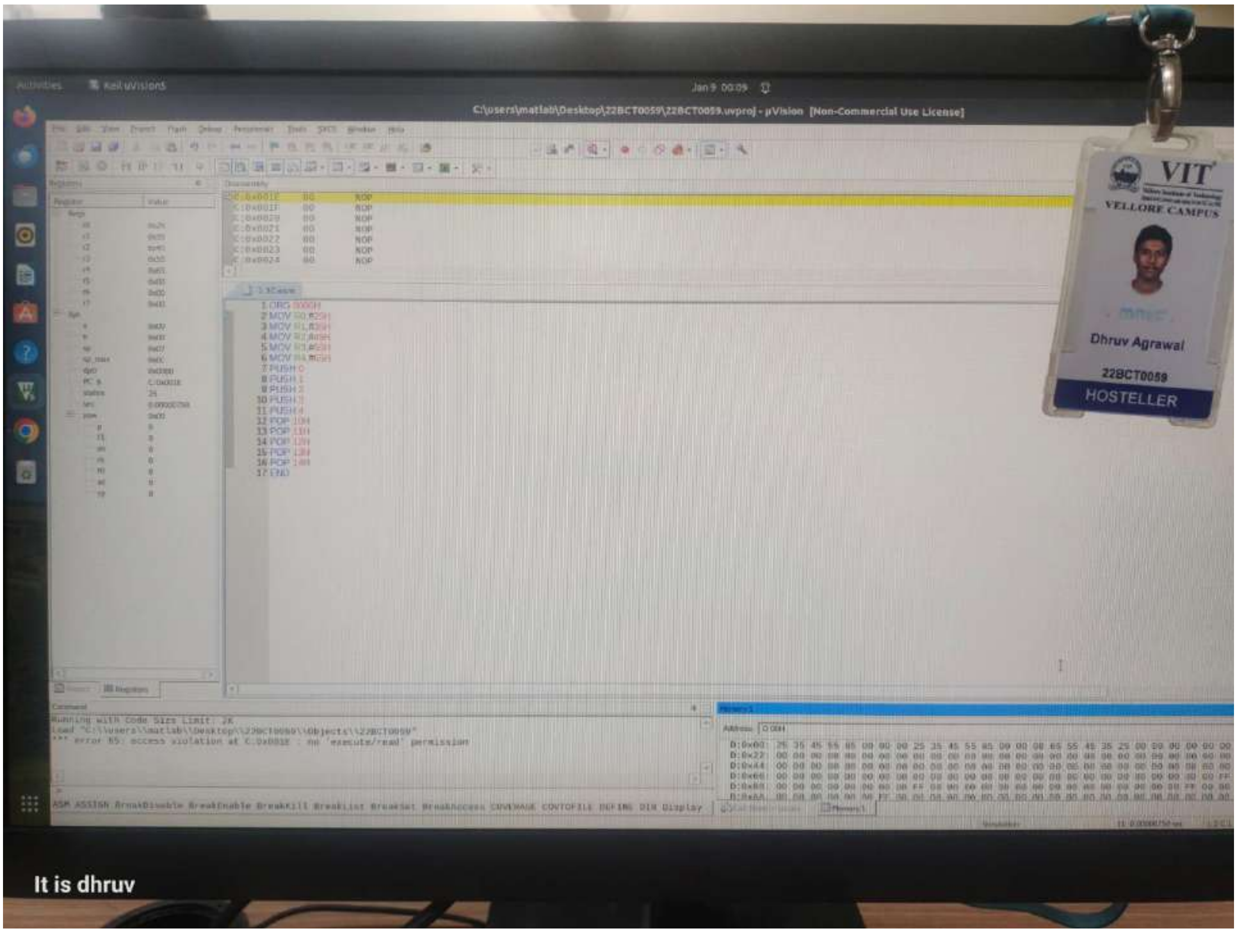




Conclusion/ Result: The Push and Pop operation are fundamental for stack management with Push adding data to stack from register and Pop, moving data from stack to register. Both are efficient in memory management and handling fn call, interrupt. The entire process is executed on Keil u Vision and answers/output is verified.

22BCT0059
DHRUV AGRAWAL

Lab Screenshot (with PSW):



C:\users\matlab\Desktop\22BCT0059\22BCT0059.uvproj - µVision [Non-Commercial Use License]

```
C:0x001E    00    NOP
C:0x001F    00    NOP
C:0x0020    00    NOP
C:0x0021    00    NOP
C:0x0022    00    NOP
C:0x0023    00    NOP
C:0x0024    00    NOP
```

```
ORG 0000H
MOV R0,#25H
MOV R1,#35H
MOV R2,#45H
MOV R3,#55H
MOV R4,#65H
PUSH 0
PUSH 1
PUSH 2
PUSH 3
PUSH 4
POP 10H
POP 11H
POP 12H
POP 13H
POP 14H
END
```

```
Running with Code Size Limit: 2K
Load "C:\\users\\matlab\\Desktop\\22BCT0059\\Objects\\22BCT0059"
*** error 65: access violation at C:0x001E : no 'execute/read' permission
```

ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE COVTOFILE DEFINE DIR Display
Memory 1
Address: D:00H

```
D:0x00: 25 35 45 55 65 00 00 00 25 35 45 55 65 00 00 00 65 55 45 35 25 00 00 00 00 00 00
```

1.4 A

Aim: Write, perform and execute assembly language program to transfer string of data from code space starting at address 200H to RAM locations starting at 40H. data is

0200H DB "VIT UNIVERSITY"

Software used : Keil uVision 6.0

Program:

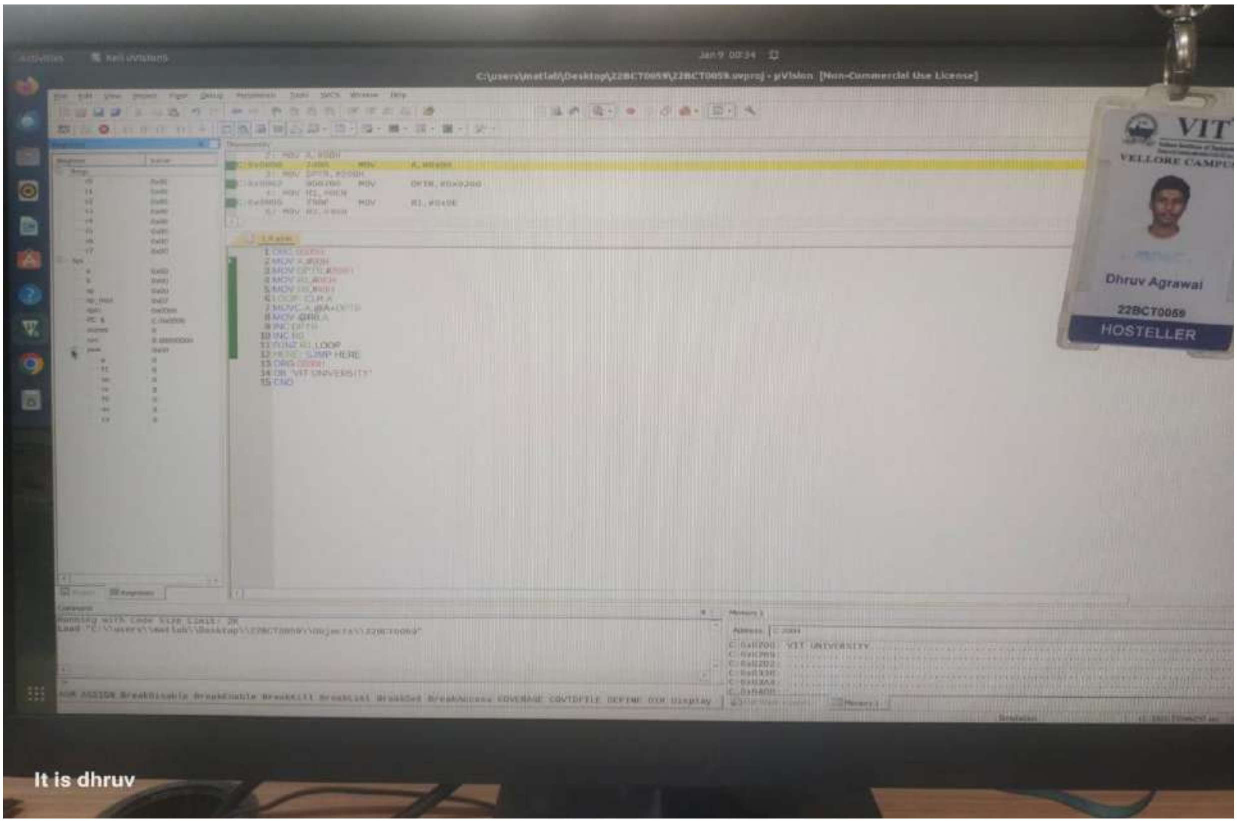
```
        ORG 0000H ; Program start address
        mov A,#00H  ; Set value 00 in A /clear Accumulator
        mov DPTR,#200H ; load DPTR with code memory add
        mov R1, #0EH   ; Load R1 with 0EH
        mov R0, #40H   ; move value 40H in R0
LOOP:   CLR A          ; clear the accumulator
        MOVC A,@A+DPTR ; Read byte from code memory into A
        mov @R0, A     ; Write byte from A into RAM
        INC DPTR   ; Increment DPTR to next code memory location
        INC R0     ; Increment R0 to next RAM location
        DJNZ R1, loop ; Decrement R1 and jump to loop until R1≠0
HERE:   SJMP HERE   ; Infinite loop.
        ORG0200H    ; code memory location where string is stored
        DB: "VIT UNIVERSITY" ; string to be copied
        END   ; End of Program
```

Expected Output:

D: 0x40 : VIT UNIVERSITY — RAM SPACE

Practical output = VIT UNIVERSITY

Lab Screenshot (with PSW):

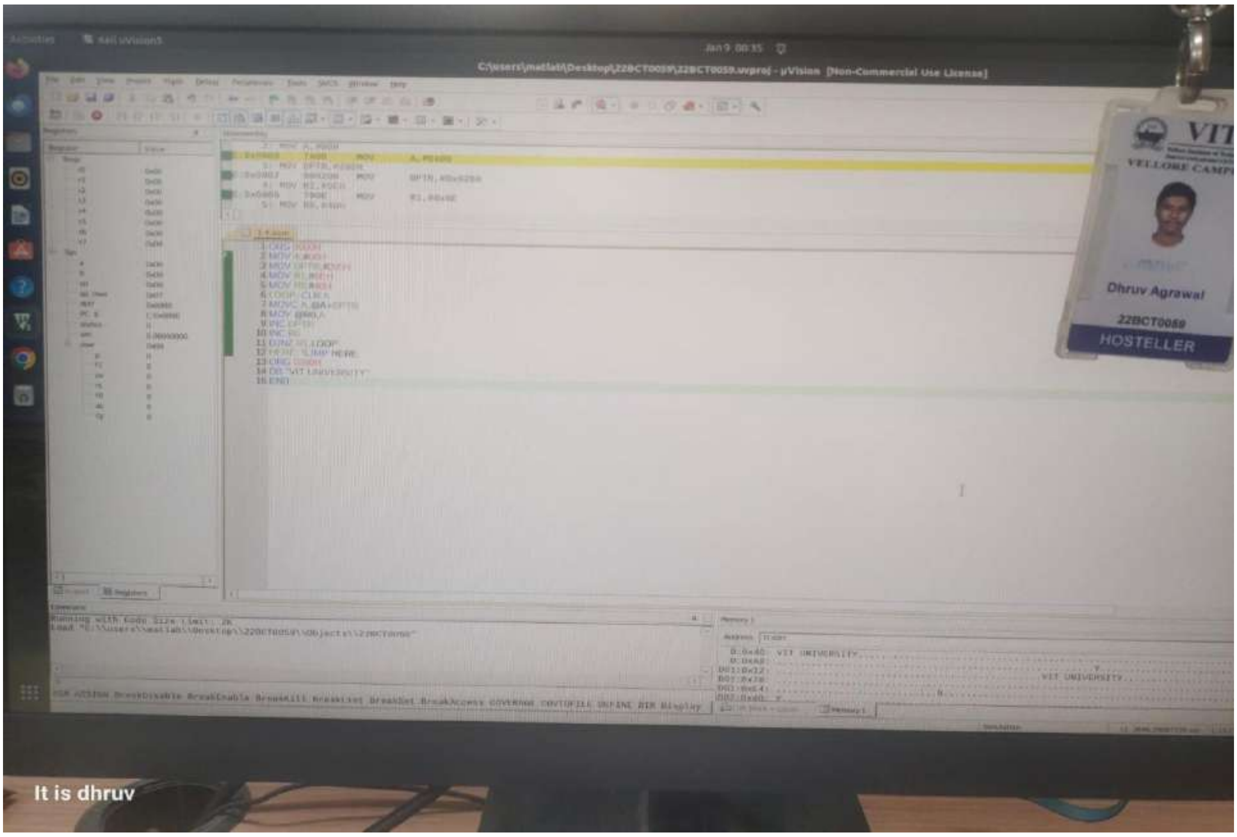


C:\users\matlab\Desktop\22BCT0059\22BCT0059.uvproj - µVision [Non-Commercial Use License]

```
ORG 0000H
MOV A,#00H
MOV DPTR,#200H
MOV R1,#0EH
MOV R0,#40H
LOOP: CLR A
MOVC A,@A+DPTR
MOV @R0,A
INC DPTR
INC R0
DJNZ R1,LOOP
HERE: SJMP HERE
ORG 200H
DB "VIT UNIVERSITY"
END
```

```
Running with Code Size Limit: 2K
Load "C:\\users\\matlab\\Desktop\\22BCT0059\\Objects\\22BCT0059"
```

```
C:0x0200: VIT UNIVERSITY
```



1.4B

Aim: To write, perform and execute assembly language program to transfer string of data from code space to RAM space in reverse order

0200H : MYDATA "VIT UNIVERSITY"

Software used : Keil uVision 6.0

Program:

```
        ORG 0000H           , Program Start address
        MOV DPTR, #020DH    ; Load DPTR with Code Memory add
        MOV R1, #14H        ; Load R1 with 14H
        MOV R0, #40H        ; Load R0 with 40H
LOOP:   CLR A               ; clear Accumulator
        MOVC A, @A+DPTR     ; Read byte from Code memory into A
        MOV @R0, A          ; write byte from A into RAM
        DEC DPL             ; Decrement DPTR by 1
        INC R0              ; Increment R0 to next RAM locatio
        DJNZ R1, LOOP       ; Decrement R1 and jump to loop if ≠0
HERE :  SJMP HERE           ; Infinite loop

        ORG 0200H           ; Code memory location where string is stored
        DB "VIT UNIVERSITY" ; String to be copied
        END                 ; End of the program
```

manual / Expected :
Calculation output

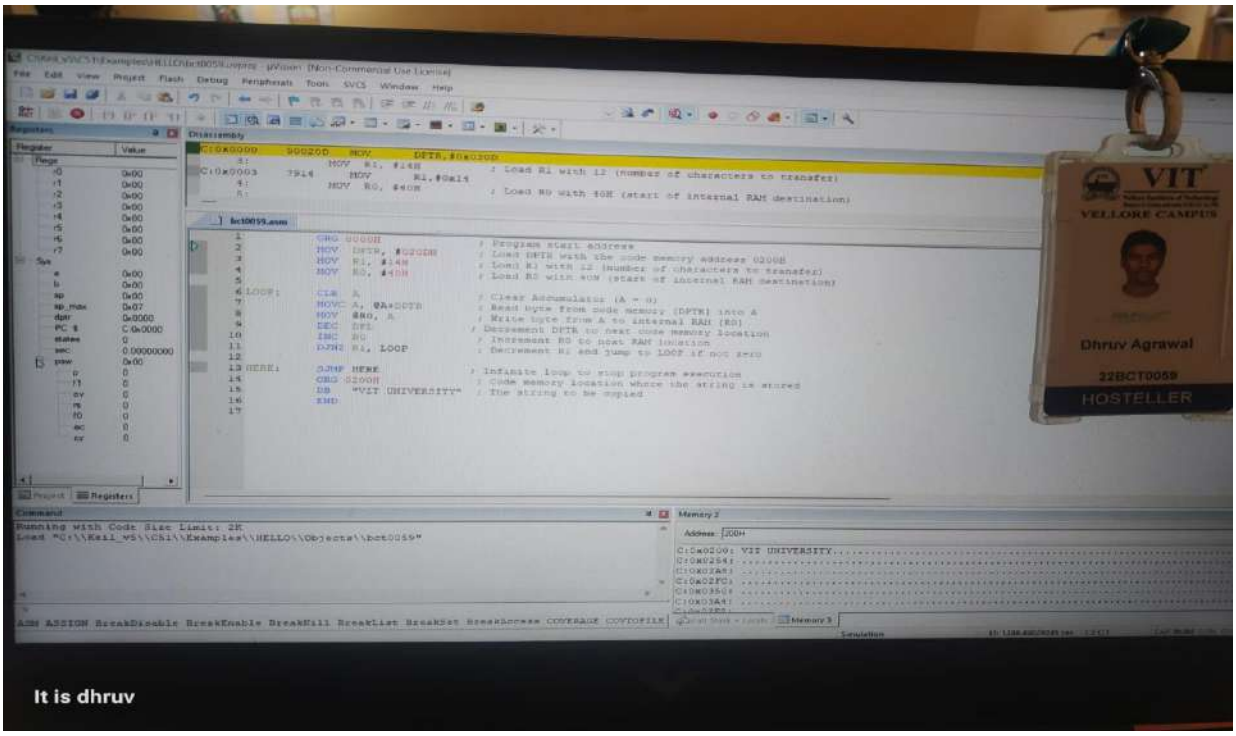
0200H : "VIT UNIVERSITY" — Code memory

D:0X40H : YTISREVINU TIV — RAM, space

PRACTICAL OUTPUT : YTISREVINU TIV

22BCT0059
DHRUV AGRAWAL

Lab Screenshot (with PSW):



```
Disassembly
C:0x0000    90020D   MOV      DPTR,#0x020D
     3:          MOV  R1, #14H              ; Load R1 with 12 (number of characters to transfer)
C:0x0003    7914     MOV      R1,#0x14
     4:          MOV  R0, #40H              ; Load R0 with 40H (start of internal RAM destination)
     5:
```

bct0059.asm

```
        ORG 0000H                ; Program start address
        MOV  DPTR, #0200H        ; Load DPTR with the code memory address 0200H
        MOV  R1, #14H            ; Load R1 with 12 (number of characters to transfer)
        MOV  R0, #40H            ; Load R0 with 40H (start of internal RAM destination)

LOOP:   CLR  A                   ; Clear Accumulator (A = 0)
        MOVC A, @A+DPTR          ; Read byte from code memory (DPTR) into A
        MOV  @R0, A              ; Write byte from A to internal RAM (R0)
        DEC  DPL                ; Decrement DPTR to next code memory location
        INC  R0                  ; Increment R0 to next RAM location
        DJNZ R1, LOOP            ; Decrement R1 and jump to LOOP if not zero

HERE:   SJMP HERE               ; Infinite loop to stop program execution
        ORG 0200H                ; Code memory location where the string is stored
        DB   "VIT UNIVERSITY"    ; The string to be copied
        END
```

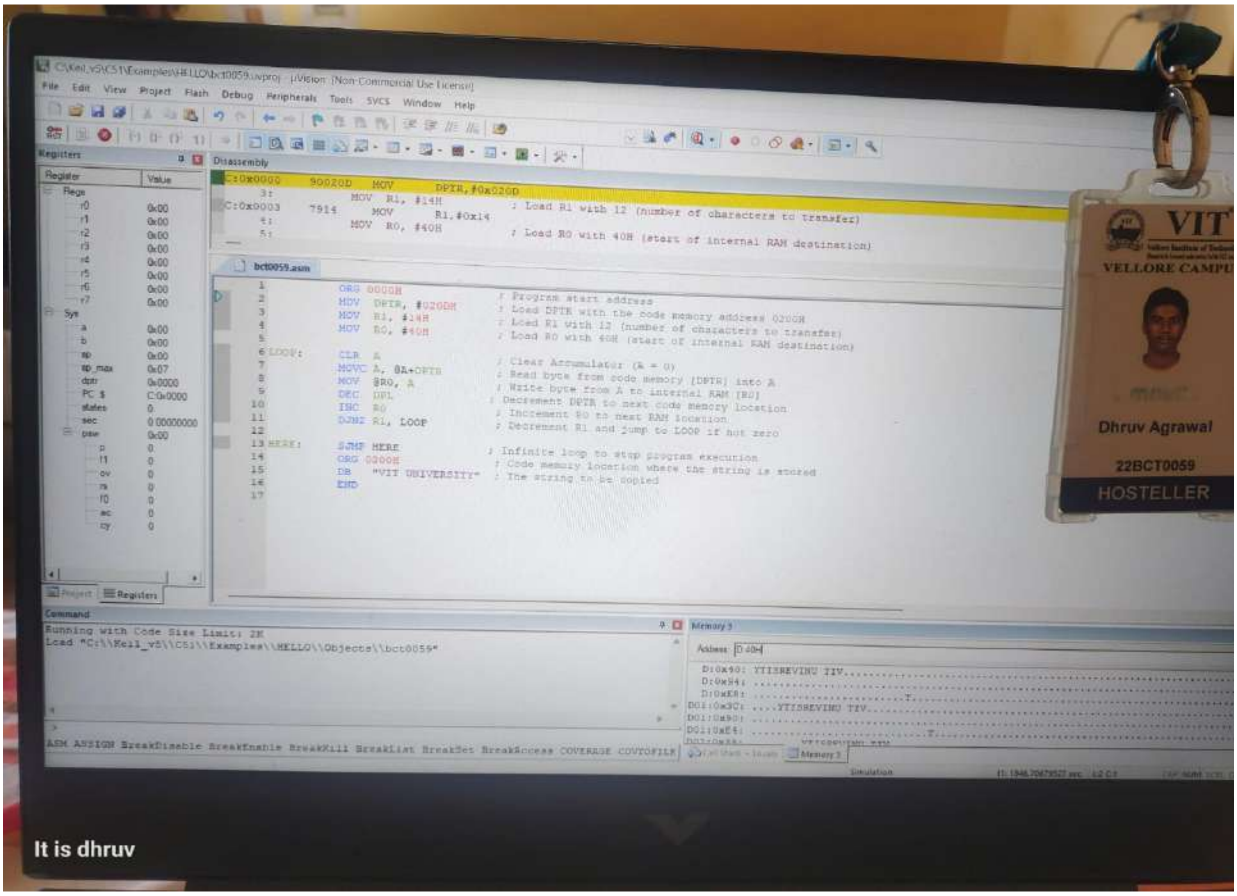
```
Register      Value
Regs
  r0          0x00
  r1          0x00
  r2          0x00
  r3          0x00
  r4          0x00
  r5          0x00
  r6          0x00
  r7          0x00
Sys
  a           0x00
  b           0x00
  sp          0x00
  sp_max      0x07
  dptr        0x0000
  PC $        C:0x0000
  states      0
  sec         0.00000000
  psw         0x00
    p         0
    f1        0
    ov        0
    rs        0
    f0        0
    ac        0
    cy        0
```

```
Command
Running with Code Size Limit: 2K
Load "C:\\Keil_v5\\C51\\Examples\\HELLO\\Objects\\bct0059"

ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE COVTOFILE
```

```
Memory 2
Address: 200H
C:0x0200: VIT UNIVERSITY.............................................
C:0x0254: ...........................................................
C:0x02A8: ...........................................................
C:0x02FC: ...........................................................
C:0x0350: ...........................................................
C:0x03A4: ...........................................................
```

It is dhruv



```
Disassembly
C:0x0000    90020D   MOV      DPTR,#0x020D
     3:          MOV  R1, #14H              ; Load R1 with 12 (number of characters to transfer)
C:0x0003    7914     MOV      R1,#0x14
     4:          MOV  R0, #40H              ; Load R0 with 40H (start of internal RAM destination)
     5:
```

bct0059.asm

```
        ORG 0000H                ; Program start address
        MOV  DPTR, #0200H        ; Load DPTR with the code memory address 0200H
        MOV  R1, #14H            ; Load R1 with 12 (number of characters to transfer)
        MOV  R0, #40H            ; Load R0 with 40H (start of internal RAM destination)

LOOP:   CLR  A                   ; Clear Accumulator (A = 0)
        MOVC A, @A+DPTR          ; Read byte from code memory (DPTR) into A
        MOV  @R0, A              ; Write byte from A to internal RAM (R0)
        DEC  DPL                ; Decrement DPTR to next code memory location
        INC  R0                  ; Increment R0 to next RAM location
        DJNZ R1, LOOP            ; Decrement R1 and jump to LOOP if not zero

HERE:   SJMP HERE               ; Infinite loop to stop program execution
        ORG 0200H                ; Code memory location where the string is stored
        DB   "VIT UNIVERSITY"    ; The string to be copied
        END
```

```
Register      Value
Regs
  r0          0x00
  r1          0x00
  r2          0x00
  r3          0x00
  r4          0x00
  r5          0x00
  r6          0x00
  r7          0x00
Sys
  a           0x00
  b           0x00
  sp          0x00
  sp_max      0x07
  dptr        0x0000
  PC $        C:0x0000
  states      0
  sec         0.00000000
  psw         0x00
    p         0
    f1        0
    ov        0
    rs        0
    f0        0
    ac        0
    cy        0
```

```
Command
Running with Code Size Limit: 2K
Load "C:\\Keil_v5\\C51\\Examples\\HELLO\\Objects\\bct0059"

ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE COVTOFILE
```

```
Memory 3
Address: D:40H
  D:0x40: YTISREVINU TIV.......................................
  D:0x94: .....................................................
  D:0xE8: .....................................................
DO1:0x3C: ....YTISREVINU TIV...................................
DO1:0x90: .....................................................
DO1:0xE4: .....................................................
```

It is dhruv

1.5A

Aim: To write, perform and execute assembly language program to add 10 bytes of data and store result in register R2 and R3. Bytes are stored in ROM space 0200H

0200H: MYDATA DB: 92, 34, 84, 129, . . .

Software used : Keil µVision 6.0

Program:

```
        ORG 0000H          ; Program start address.
        MOV DPTR, #200H    ; Load DPTR with code memory address
        MOV R0, #10H       ; Load R1 with 10H
LOOP:   CLR A              ; clear accumulator
        MOVC A,@A+DPTR     ; Read byte from code memory into A
        ADD A, R2          ; Add value of R2 into A
        JNC NEXT           ; Jump to next location if C≠1
        INC R3             ; Increment R3 by 1
NEXT:   INC DPTR           ; Increment DPTR to next code memory location
        MOV R2, A          ; Load value of A in R2
        DJNZ R0, LOOP      ; Decrement R0 and jump to loop if R0≠0
HERE:   SJMP HERE          ; Infinite loop
        ORG : 200H         ; code memory location where string is stored
        DB 22H, 43H, 23H, 34H, 31H, 77H, 91H, 33H, 43H, 7H  ↳ Data to be added.
        END                ; end of program
```

Practical = 272H

Expected output:

manual Calculation

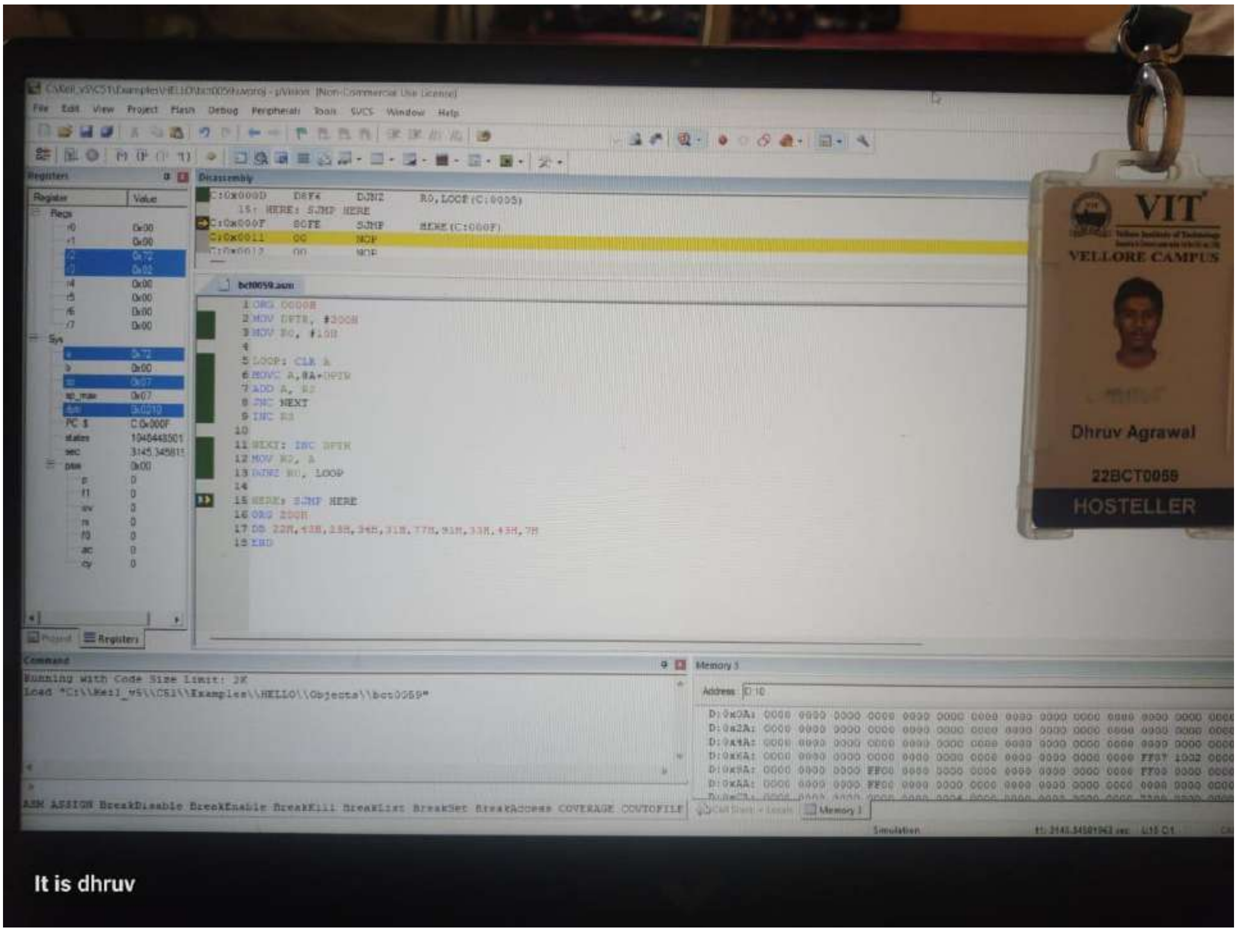
22H + 43H + 23H + 34H + 31H + 77H + 91H + 33H
+ 43H + 7H

= 272H ∴ $\frac{02}{R3}$ $\frac{72}{R2}$ / Ans.

= 1001110010 ∴ Parity bit = 1 ∴ PSW = 0x01

22BCT0059
DHRUV AGRAWAL

Lab Screenshot (with PSW):



```
C:0x000D    D8F4    DJNZ    R0,LOOP(C:0005)
    15: HERE: SJMP HERE
C:0x000F    80FE    SJMP    HERE(C:000F)
C:0x0011    00      NOP
C:0x0012    00      NOP
```

```
ORG 0000H
MOV DPTR, #200H
MOV R0, #10H

LOOP: CLR A
MOVC A,@A+DPTR
ADD A, R2
JNC NEXT
INC R3

NEXT: INC DPTR
MOV R2, A
DJNZ R0, LOOP

HERE: SJMP HERE
ORG 200H
DB 22H,43H,23H,34H,31H,77H,91H,33H,43H,7H
END
```

```
Regs
  r0      0x00
  r1      0x00
  r2      0x72
  r3      0x02
  r4      0x00
  r5      0x00
  r6      0x00
  r7      0x00
Sys
  a       0x72
  b       0x00
  sp      0x07
  sp_max  0x07
  dptr    0x0210
  PC $    C:0x000F
  states  1040448501
  sec     3145.345815
  psw     0x00
    p     0
    f1    0
    ov    0
    rs    0
    f0    0
    ac    0
    cy    0
```

```
Running with Code Size Limit: 2K
Load "C:\\Keil_v5\\C51\\Examples\\HELLO\\Objects\\bct0059"
```

It is dhruv

1.5B

Aim: Write a Program to add 10 Bytes of BCD data and store result in R2 and R3. Bytes are stored in ROM space at 300H

MYDATA: DB 92H, 34H, 84H, 29H

Software used: Keil µVision 6.0

Program:

```
        ORG 0000H      ; Program start Address (origin)
        mov DPTR, #300H ; Load DPTR with code memory add
        mov R0, #10H   ; Load R0 with 10H.
LOOP:   CLR A          ; clear the accumulator.
        movc A, @A+DPTR ; Read byte from code memory into A
        ADD A, R2      ; Add value of R2 into A
        DA A           ; Decimal adjust content of accumulator
        JNC NEXT       ; Jump to NEXT if carry ≠ 1
        INC R3         ; Increment R3 to next RAM location
NEXT:   INC DPTR       ; Increment DPTR by 1
        mov R2, A      ; Load value of Accumulator in R2
        DJNZ R0, LOOP  ; Decrement R0 and jump to LOOP
                         until R0 ≠ 0
HERE:   SJMP HERE      ; Infinite Loop
        ORG 300H       ; Code memory location
        DB 22H, 43H, 23H, 34H, 31H, 77H, 91H, 33H, 43H, 7H
                       ↳ Data in code memory
        END            ; End of Program
```

Manual Calculation / Expected Output: ⊛ Note in BCD only 0 to 9:

Practical = 404H

22H + 43H + 23H + 34H + 31H + 77H + 91H + 33H + 43 + 7H

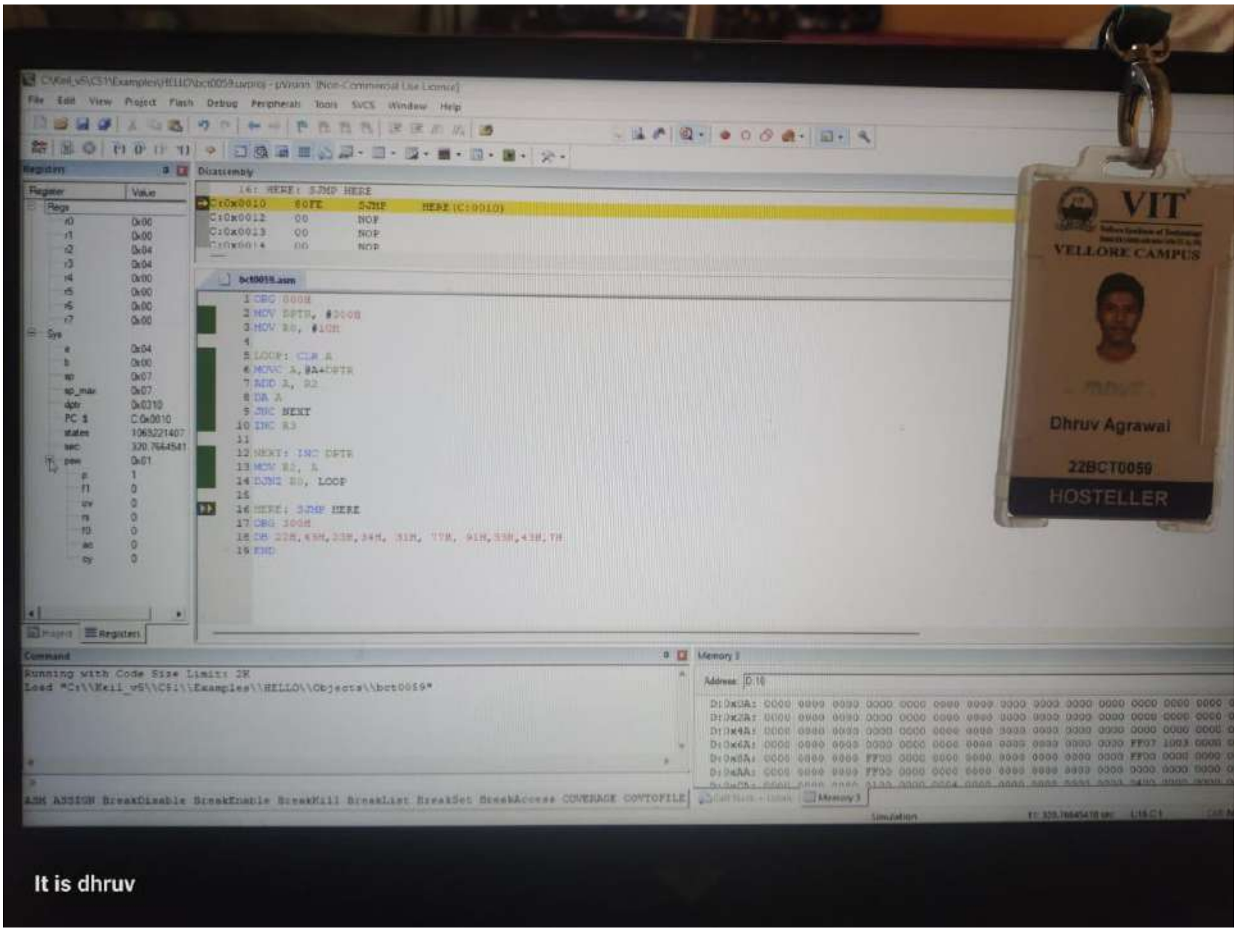
= 404H

= (1000000 0100)2

∴ R2 = 04H R3 = 04H

Ans

Lab Screenshot (with PSW):



C:\Keil_v5\C51\Examples\HELLO\bct0059.uvproj - µVision [Non-Commercial Use License]
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
bct0059.asm
Command
Memory 1
Project
Registers
Memory 1
Simulation
VIT
VELLORE CAMPUS
Dhruv Agrawal
22BCT0059
HOSTELLER

Register	Value
Regs	
r0	0x00
r1	0x00
r2	0x04
r3	0x04
r4	0x00
r5	0x00
r6	0x00
r7	0x00
Sys	
a	0x04
b	0x00
sp	0x07
sp_max	0x07
dptr	0x0310
PC $	C:0x0010
states	1063221407
sec	320.7664541
psw	0x01
p	1
f1	0
ov	0
rs	0
f0	0
ac	0
cy	0

```
    16: HERE: SJMP HERE
C:0x0010    80FE     SJMP     HERE(C:0010)
C:0x0012    00       NOP
C:0x0013    00       NOP
C:0x0014    00       NOP
```

```
ORG 000H
MOV DPTR, #300H
MOV R0, #10H

LOOP: CLR A
MOVC A,@A+DPTR
ADD A, R2
DA A
JNC NEXT
INC R3

NEXT: INC DPTR
MOV R2, A
DJNZ R0, LOOP

HERE: SJMP HERE
ORG 300H
DB 22H,43H,23H,34H, 31H, 77H, 91H,33H,43H,7H
END
```

```
Running with Code Size Limit: 2K
Load "C:\\Keil_v5\\C51\\Examples\\HELLO\\Objects\\bct0059"
```

```
ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE COVTOFILE
```

Address: D:10

It is dhruv

Conclusion: Transfer of data from code space to RAM space in reverse and same order is essential for data manipulation. It is typically done byte by byte. The data in code is added in accumulator to store the result in R2, R3. For BCD no., we need to use decimal adjust instruction.