Sub: Microprocessors and Microcontrollers Lab

Lab Task 1

TASK1 EXPT1

WAP to perform basic arithmetic operation using Keil micro vision

PROGRAM 1-ADDITION

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

PROGRAM 2-SUBTRACTION

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

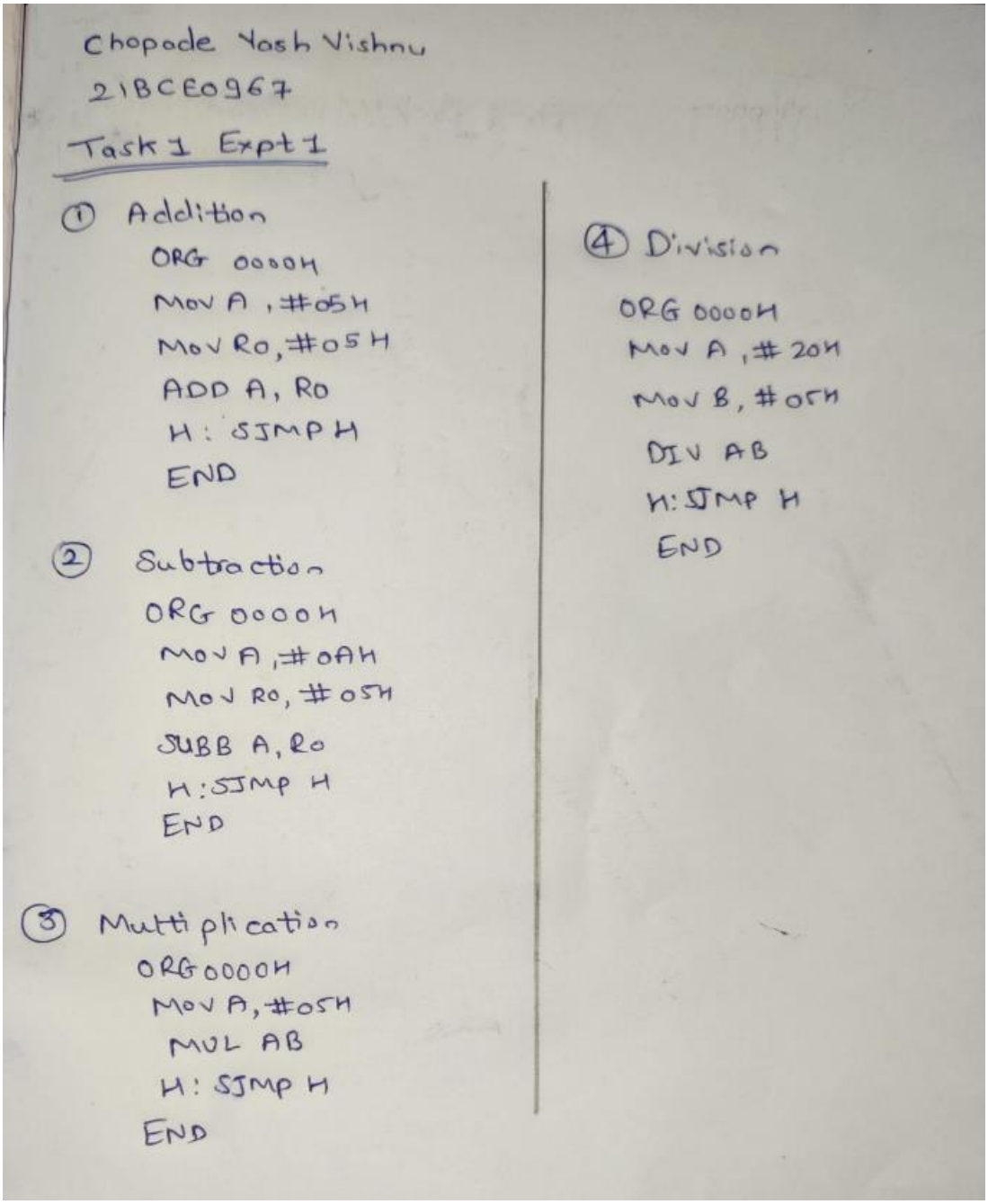
PROGRAM 3-MULTIPLICATION

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

PROGRAM 4-DIVISON

```
ORG 0000H
MOV A, #20H
```

MOV B, #05H
DIV AB
H: SJMP H
END



Chopade Yash Vishnu
21BCE0967
Task 1 Expt 1

① Addition

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

② Subtraction

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

③ Multiplication

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

④ Division

```
ORG 0000H
MOV A, #20H
MOV B, #05H
DIV AB
H: SJMP H
END
```

Addition:

Results: On addition of 05H we get 0a in accumulation

Subtraction:

Results: On subtraction of 05H from 0a we get 05 as results in accumulation

Multiplication:

Results: On multiplication of 05H and 05h we get 19 as results in accumulation

Division:

Results: On division 0*20 by 0*05 we get 0*06 as the result in the accumulation.

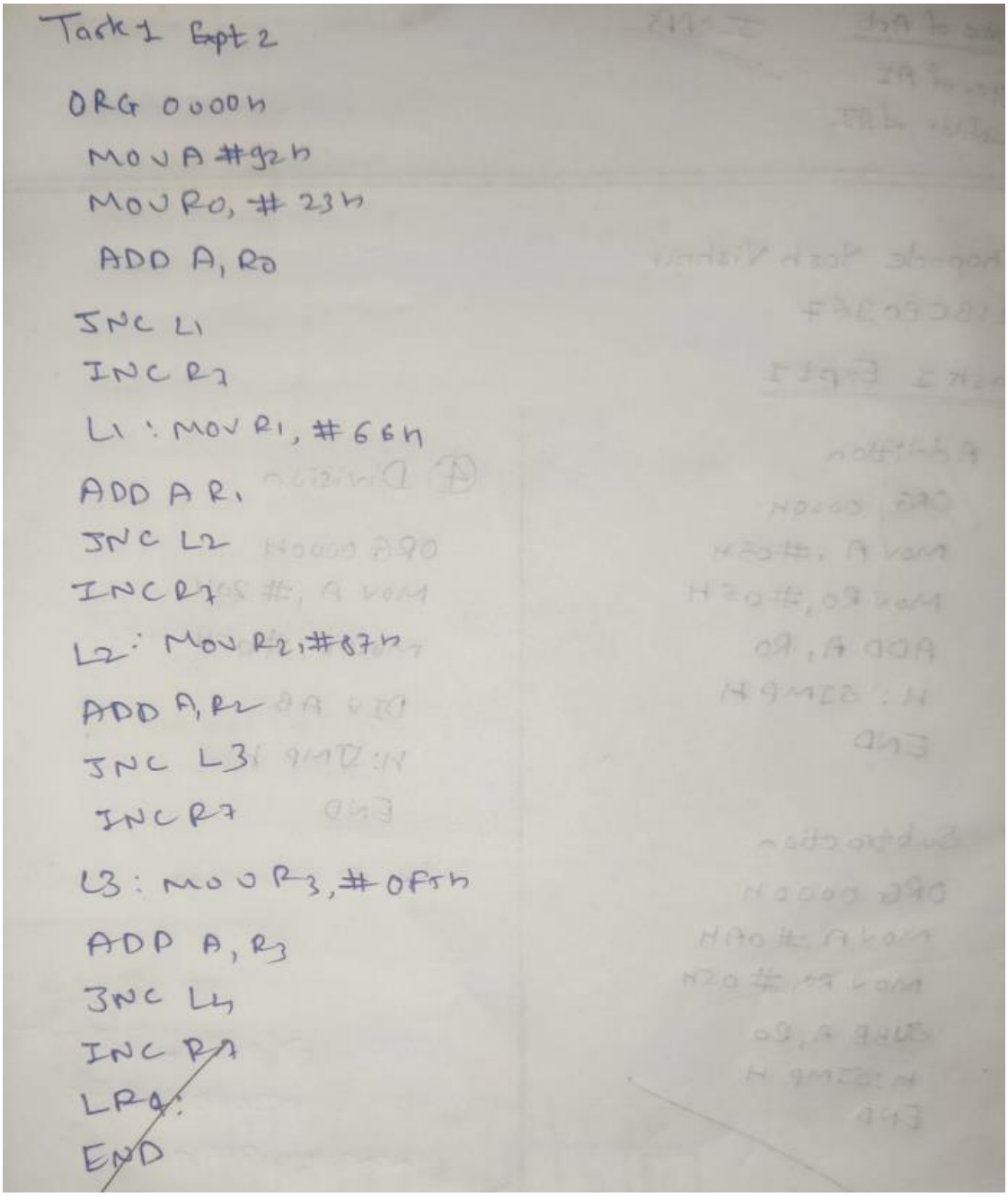
TASK1 EXPT2

Program - 1

Write and assemble a program to add the following data and then use the simulator to examine the CY flag.

92H, 23H, 66H, 87H, F5H

```
ORG 0000H
MOV A, #92H
MOV R0, #23H
ADD A, R0
JNC L1
INC R7
L1: MOV R1, # 66H
ADD A, R1
JNC L2
INC R7
L2: MOV R2, #87H
ADD A, R2
JNC L3
INC R7
L3: MOV R3, #0F5H
ADD A, R3
JNC L4
INC R7
L4:
END
```



Task 1 Expt 2

```
ORG 0000h
MOV A #92h
MOV R0, #23h
ADD A, R0
JNC L1
INC R7
L1: MOV R1, #66h
ADD A R1
JNC L2
INCR7
L2: MOV R2, #87h
ADD A, R2
JNC L3
INCR7
L3: MOV R3, #0F5h
ADD A, R3
JNC L4
INC R7
LR4:
END
```

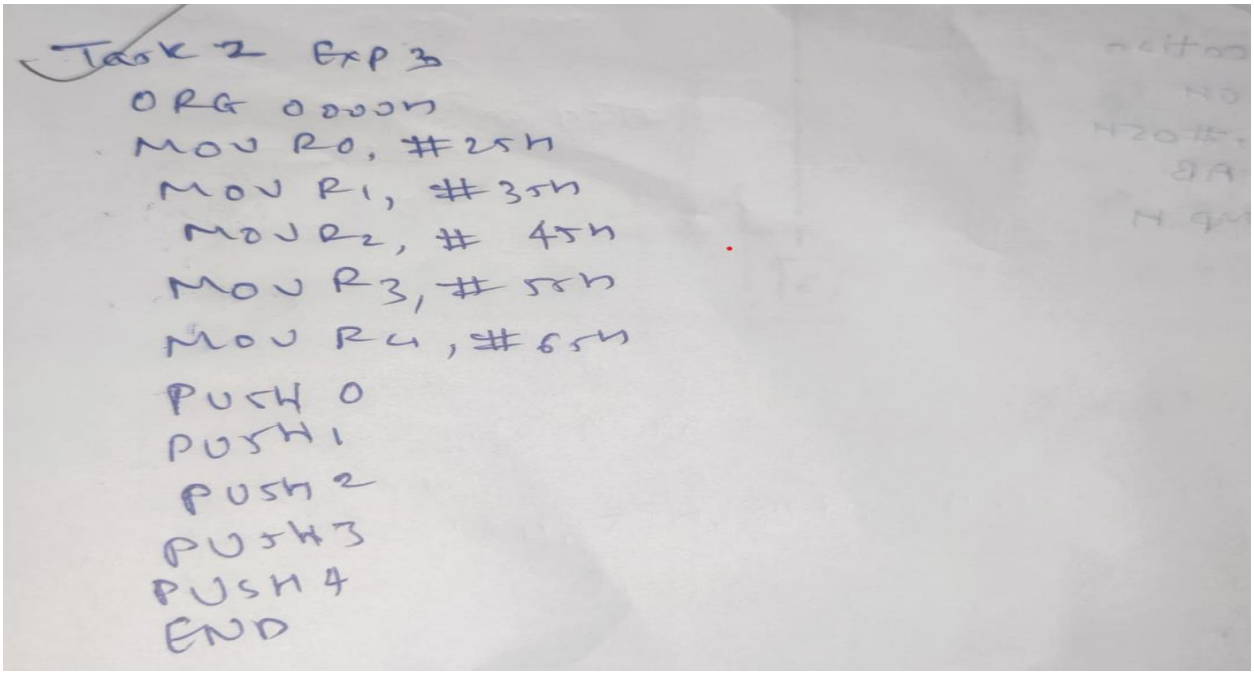
Results: On addition of 92H, 23H, 66H, B7H, F5H we get 97H as results in accumulation

TASK1 EXPT3

Program - 1

Write and assemble a program to load values into each of registers R0 - R4 and then push each of these registers onto the stack. Single-step the program, and examine the stack and the SP register after the execution of each instruction.

```
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
```



Results: with the help of PUSH operation, we get 25H, 35H, 45H, 55h, 65H ,45H, 55H, 65H in the r0, r1, r2, r3, r4 registry respectively.

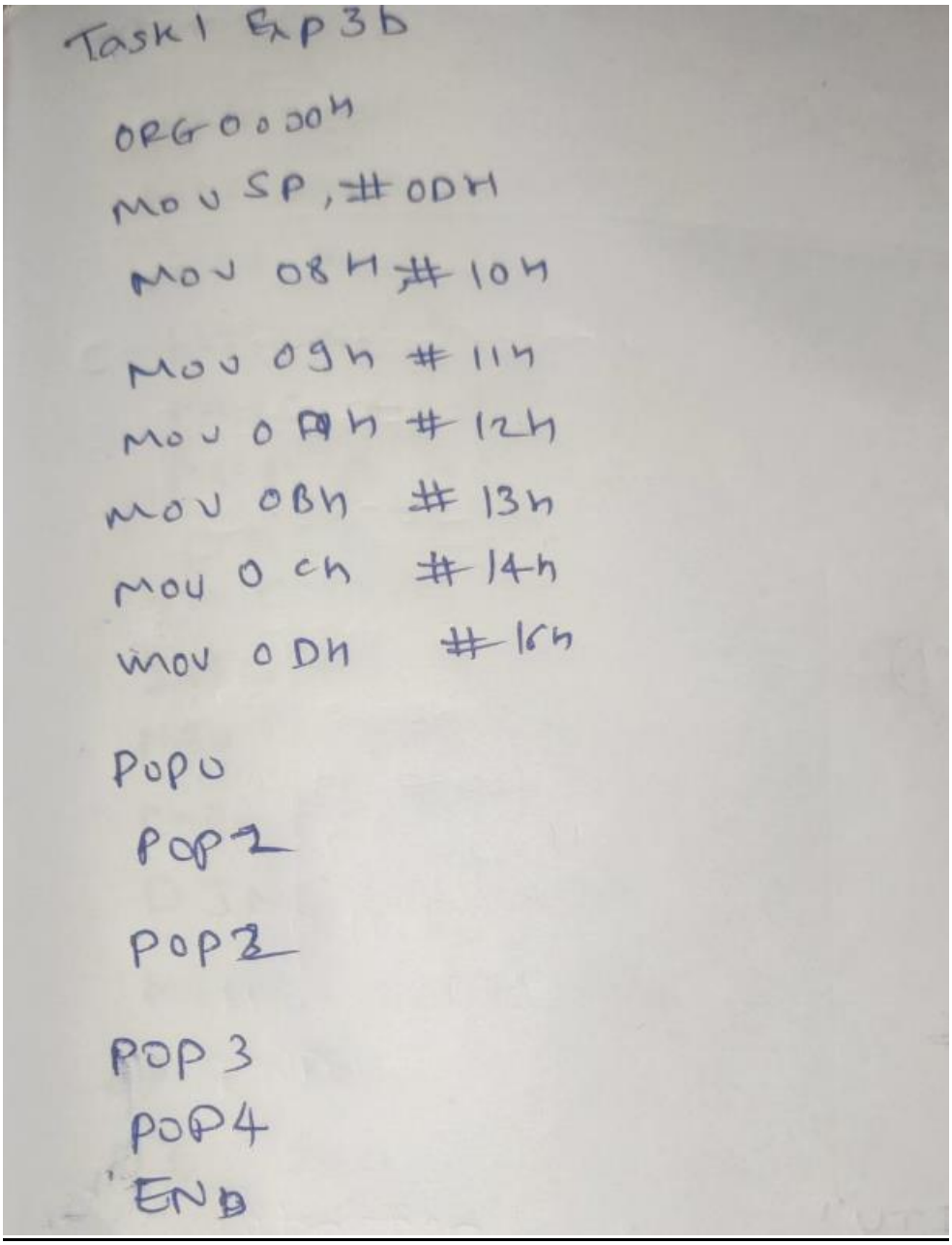
Program - 2

Write an 8051 assemble language program to:

(a) Set SP = 0D,
(b) Put a different value in each of RAM locations 0D, 0C, 0B, 0A, 09 and 08
(c) POP each stack location into registers R0 - R4.

Use the simulator to single-step and examine the registers, the stack, and the stack pointer.

```
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

END
```



Results: With the help of POP operation, we put the dots in RAM which we pop into registers.

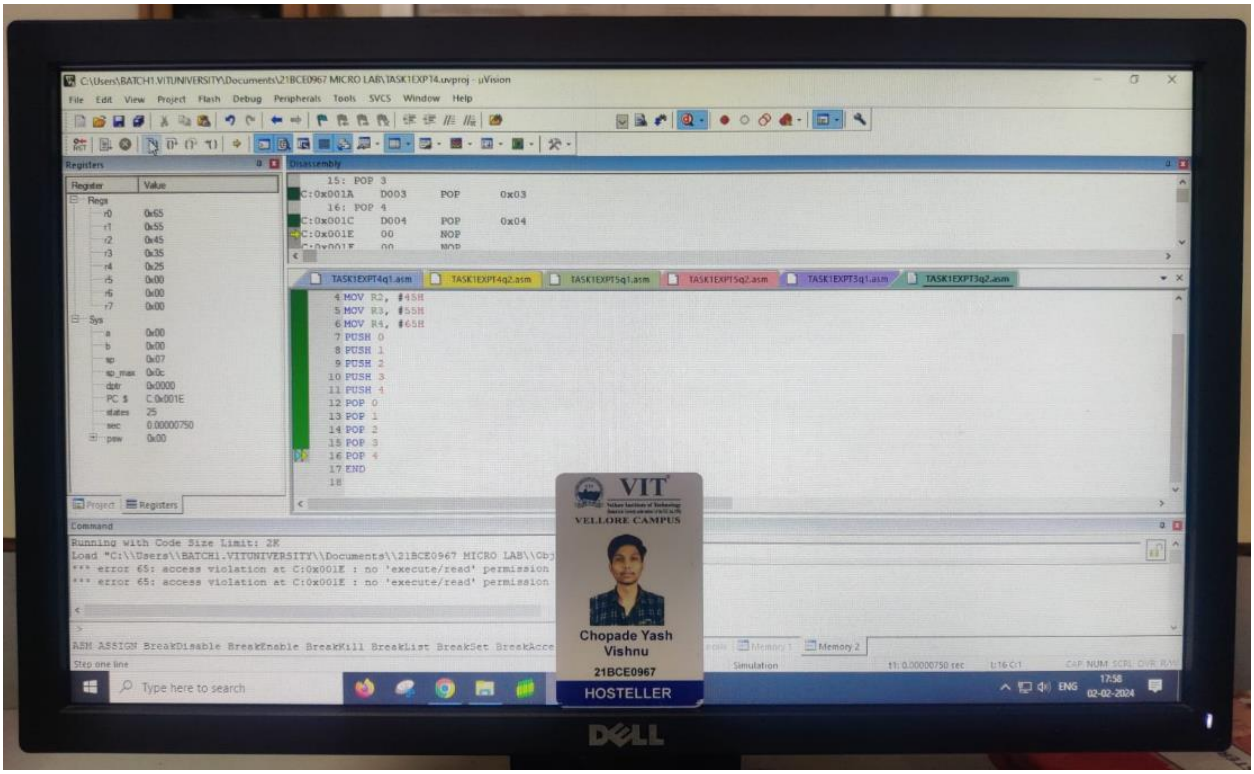
Program-3

Write and assemble a program to load values into each of registers R0 - R4 and then push each of these registers onto the stack and pop them back. Single-step the program, and examine the stack and the SP register after the execution of each instruction.

```
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 0

POP 1

POP 2

POP 3

POP 4

END
```

Task 1 Exp3

```
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 0
POP 1
POP 2
POP 3
POP 3
END
```



Results: Using PUSH operation, we first load each value into registers R0-R4 then push it in registers, and them POP them back and show them in registers.

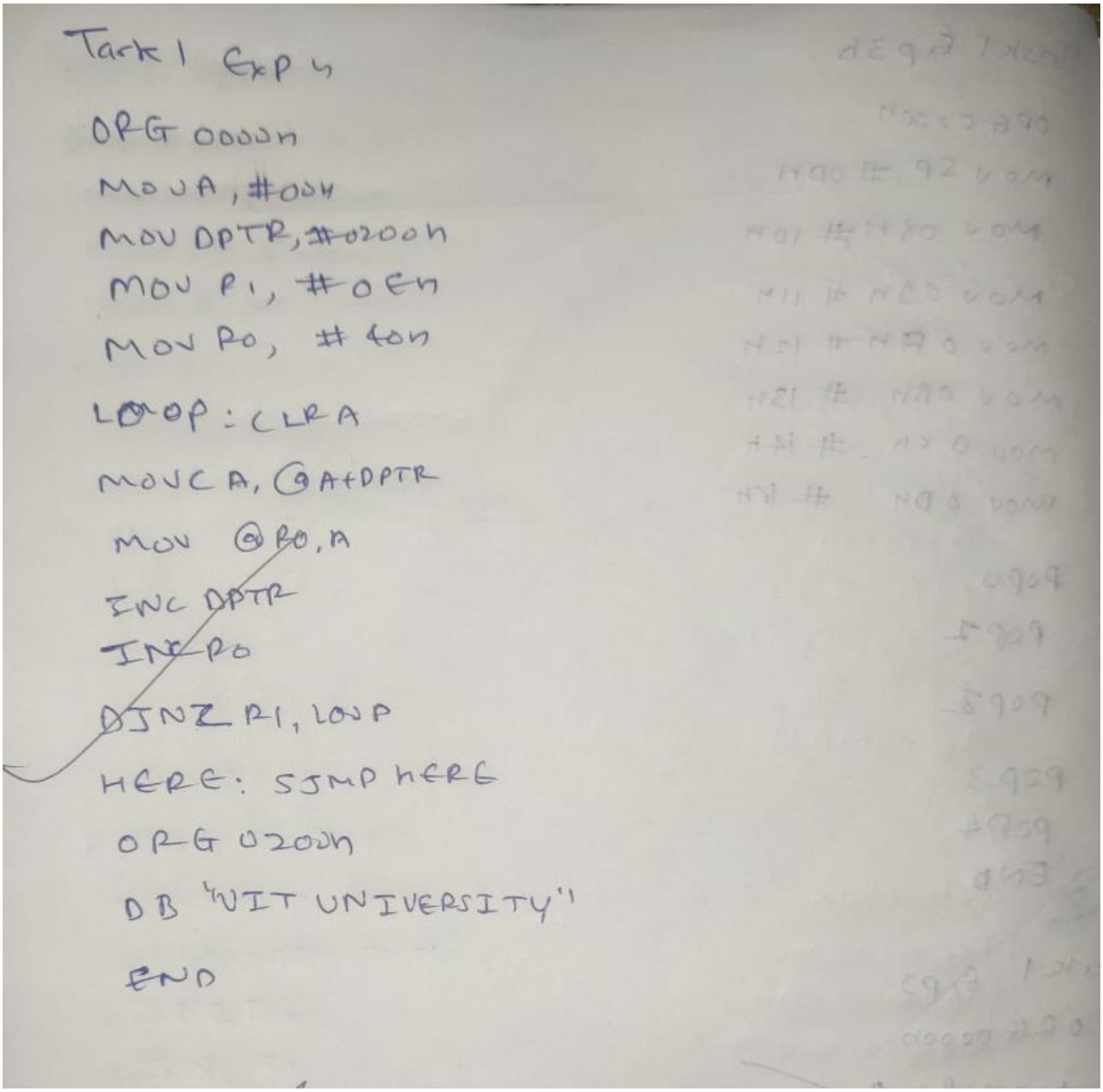
TASK1 EXPT4

Program-1

Write a program to transfer a string of data from code space starting at address 200H to RAM locations starting at 40H. The data is as shown below:

0200H: DB "VIT UNIVERSITY" Using the simulator, single-step through the program and examine the data transfer and registers.

```
ORG 0000H

MOV A, #00H

MOV DPTR, #0200H

MOV R1, #0EH

MOV RO, #40H

LOOP: CLR A

MOVC A,@A+DPTR

MOV @RO, A

INC DPTR

INC RO

DJNZ R1, LOOP

HERE: SJMP HERE

ORG 0200H

DB "VIT UNIVERSITY"

END
```

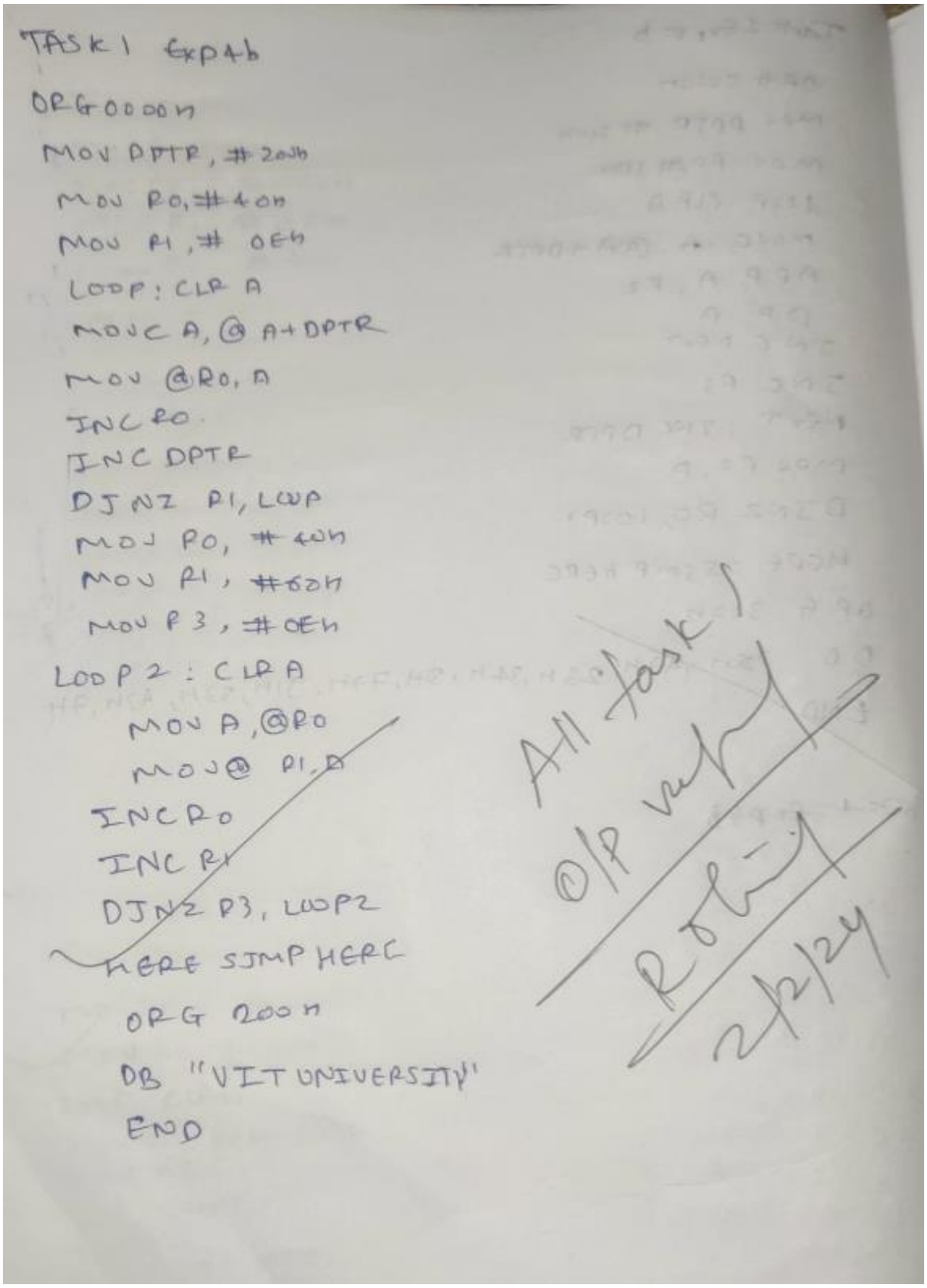


Results: Program transfers string "VIT UNIVERSITY" from code space (0200H) to RAM (starting at 40H) using 8051 microcontroller instructions MOV, CLR, MOVC, INC, DJNZ, and SJMP, with DPTR and R0 registers. Loop iterates through string characters, copying each to RAM. Program ends in an infinite loop**.**

Program 2

Add the following subroutine to the program 1, single-step through the subroutine and examine the RAM locations. After data has been transferred from ROM space into RAM, the subroutine should copy the data from RAM locations starting at 40H to RAM locations starting at 60H

```
ORG 000H

MOV DPTR, #200H

MOV RO, #40H

MOV R1, #0EH

LOOP: CLR A

MOVC A, @A+DPTR

MOV @RO, A

INC RO

INC DPTR

DJNZ R1, LOOP

MOV RO, #40H

MOV R1, #60H

MOV R3, #0EH

LOOP2: CLR A

MOV A,@RO

MOV @R1, A

INC RO

INC R1

DJNZ R3, LOOP2

HERE: SJMP HERE

ORG 200H

DB "VIT UNIVERSITY"

END
```



```
TASK 1 Expt 6
ORG 0000H
MOV DPTR, #2000H
MOV R0, #40H
MOV R1, # 0EH
LOOP: CLR A
MOVC A, @ A+DPTR
MOV @R0, A
INC R0
INC DPTR
DJNZ R1, LOOP
MOV R0, #40H
MOV R1, #60H
MOV P3, #0EH
LOOP 2 : CLR A
MOV A, @R0
MOV @ R1, A
INC R0
INC R1
DJNZ P3, LOOP2
HERE SJMP HERE
ORG 2000H
DB "VIT UNIVERSITY"
END
```

All task I/P verified
Result
24/2/24

Results: Subroutine added to copy data from RAM (starting at 40H) to RAM (starting at 60H) after initial data transfer from ROM to RAM. It uses MOV, CLR, MOVC, INC, DJNZ, and SJMP instructions. Two loops iterate through data, first copying from ROM to RAM, then from RAM to RAM. Program ends in an infinite loop.

TASK1 EXPT5

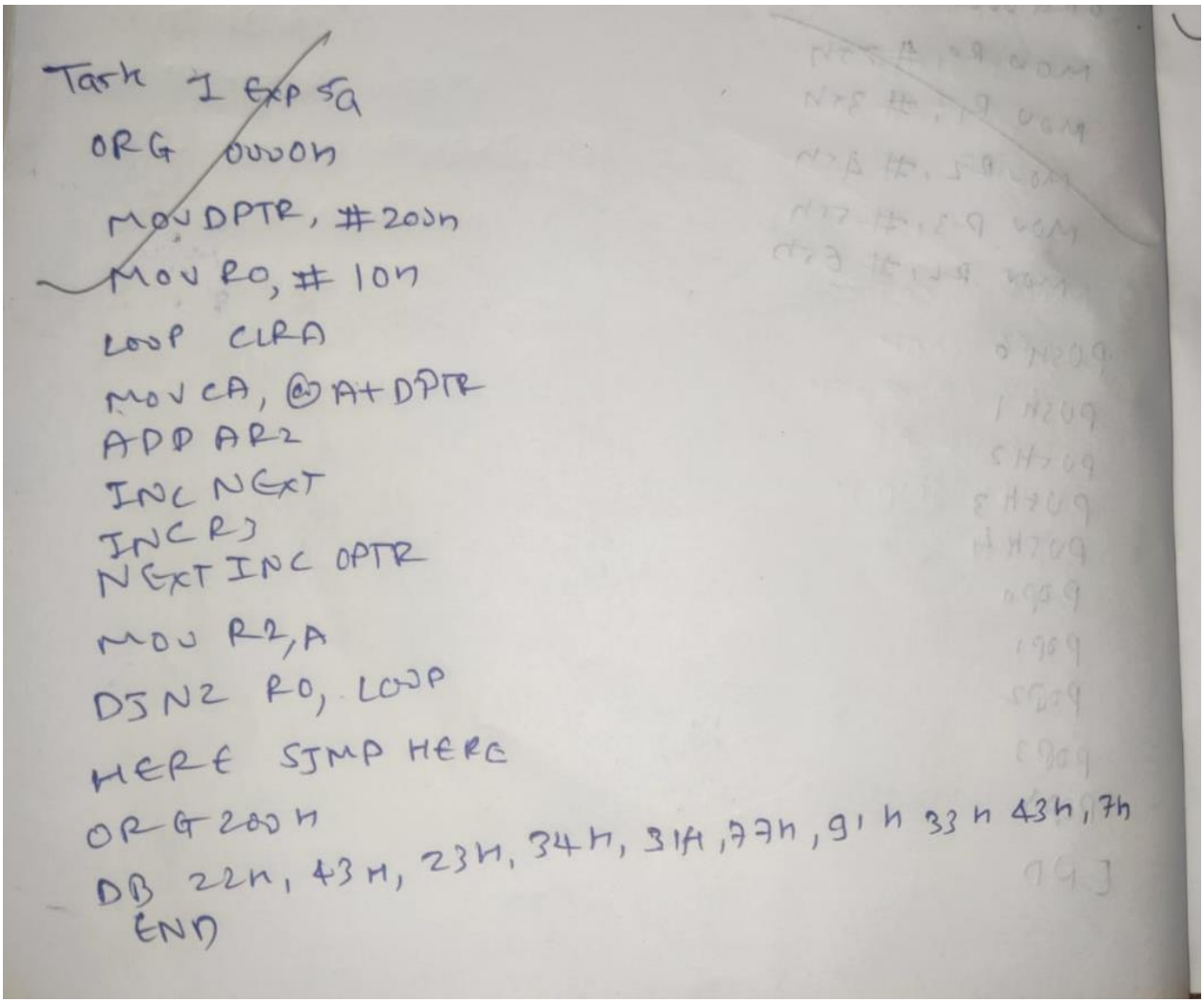
Program 1

Write a program to add 10 bytes of data and store the result in registers R2 and R3. The bytes are stored in the ROM space starting at 200H. The data would look as follows:

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

Pick your own data. Notice that you must first bring the data from ROM space into the CPU's RAM, and then add them together. Use a simulator to single-step the program and examine the data.

```
ORG 000H

MOV DPTR, #200H

MOV RO, #10H

LOOP: CLR A

MOVCA, @A+DPTR

ADD A R2

INC NEXT

INC R3

NEXT INC OPTR

MOV R2, A

DJNZ RO, LOOP

HERE SIMP HERE

ORG 200H

DB 22H,43H,23H,34H,314,774,914,33H,434,7H

END
```



Results: Program adds 10 bytes from ROM (starting at 200H) by bringing data into RAM and adding to registers R2 and R3. It uses MOV, CLR, MOVC, ADD, INC, DJNZ, and SJMP instructions in a loop. Results are stored in R2 and R3. Program ends in an infinite loop.

Program 2

Write a program to add 10 bytes of BCD data and store the result in R2 and R3. The bytes are stored in. ROM space starting at 300H. The data would look as follows:

MYDATA: DB 92H, 34H, 84H, 29H... pick your own data.

Notice that you must first bring the data from ROM space into the CPU's RAM, and then add them together. Use a simulator to single-step the program and examine the data

ORG DOOH

MOV DPTR, #300H-

MOV RD, #10H

LOOP: CLR A

MOVCA, @A+DPTR

ADD A, R2

DA A

INC NEXT

INC R3

NEKT INC DPTR

MOV R2, A

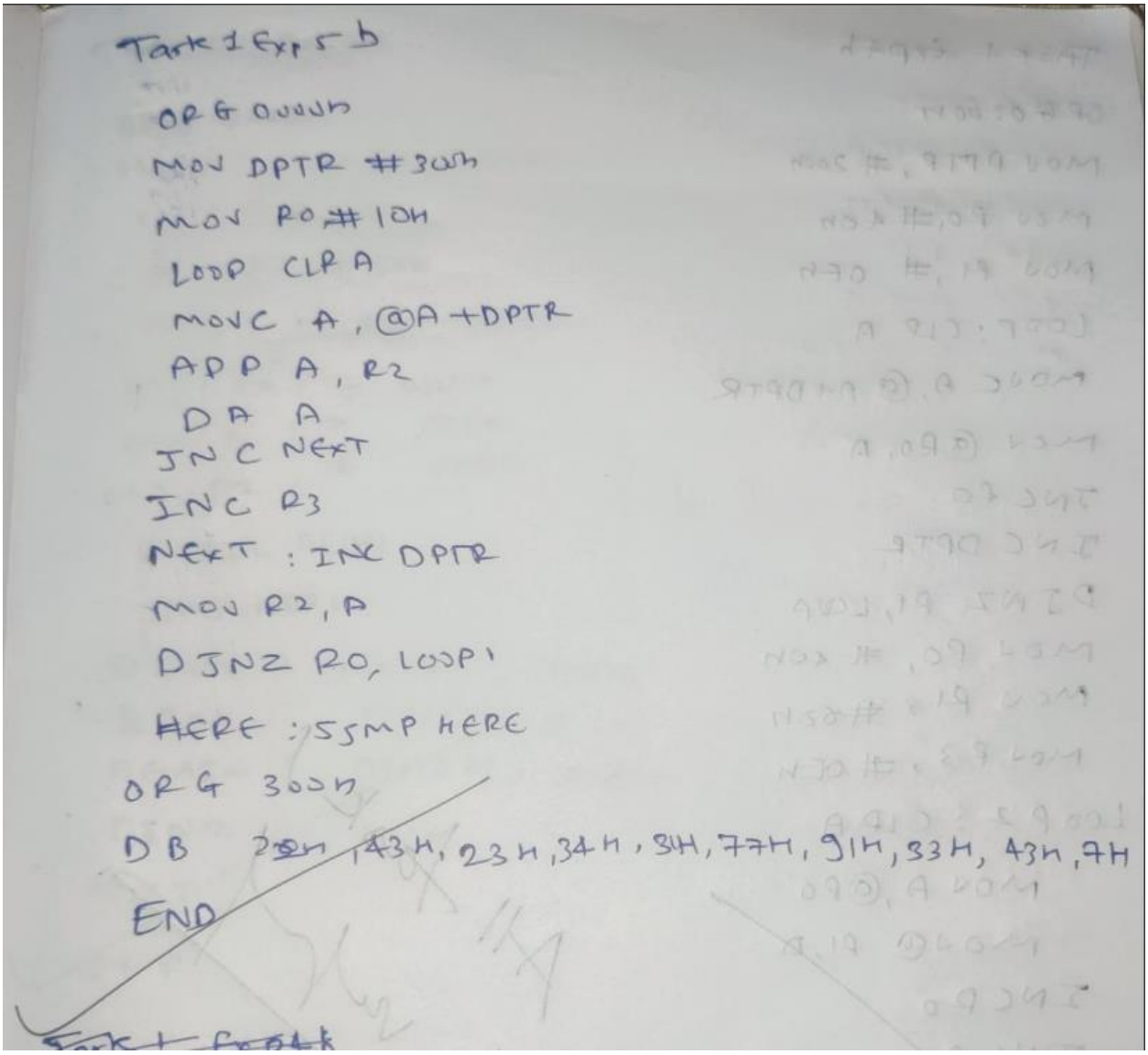
DJNZ RO, LOOP

HERE: SIMP HERE

ORG 300H

DB 22H 43H 23H,34H 31H,774,91H,338,43H 7H

END



Results: Program adds 10 bytes of BCD data from ROM (starting at 300H) by bringing data into RAM and adding to registers R2 and R3. It uses MOV, CLR, MOVC, ADD, DA, INC, DJNZ, and SJMP instructions in a loop. Results are stored in R2 and R3. Program ends in an infinite loop.