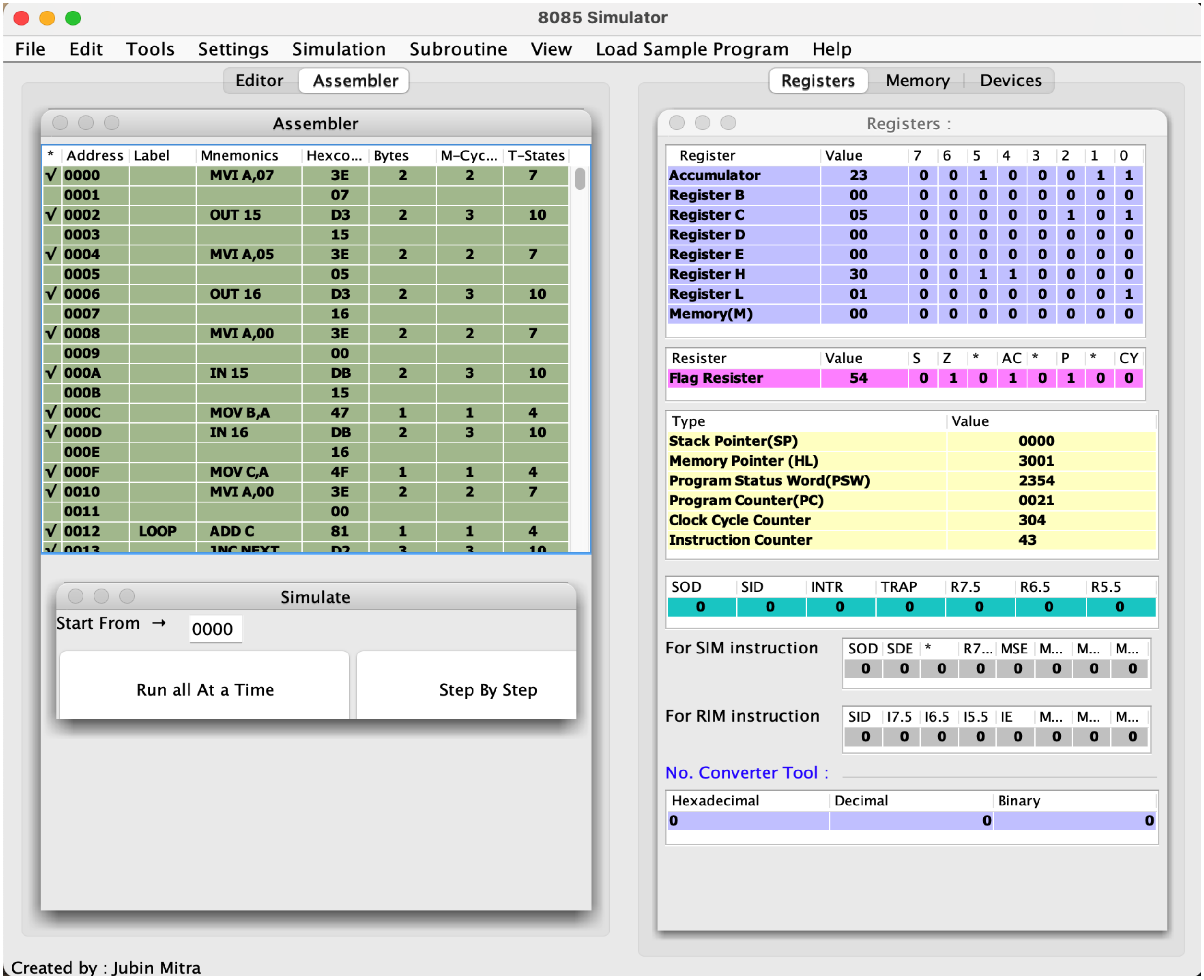
M&A

PRACTICAL -3

Aim: Learning Programs using Branch Instructions like JMP, JZ,JNZ, JC etc.

Exercise : (Solution must be handwritten in book)

1. Write a Program to multiply two 8-bit numbers given as input on input ports 15H and 16H. Save the lower byte of the result on memory location 3000H and higher byte of the result on3001H.



*	Address	Label	Mnemonics	Hexco...	Bytes	M-Cyc...	T-States
√	0000		MVI A,07	3E	2	2	7
	0001			07			
√	0002		OUT 15	D3	2	3	10
	0003			15			
√	0004		MVI A,05	3E	2	2	7
	0005			05			
√	0006		OUT 16	D3	2	3	10
	0007			16			
√	0008		MVI A,00	3E	2	2	7
	0009			00			
√	000A		IN 15	DB	2	3	10
	000B			15			
√	000C		MOV B,A	47	1	1	4
√	000D		IN 16	DB	2	3	10
	000E			16			
√	000F		MOV C,A	4F	1	1	4
√	0010		MVI A,00	3E	2	2	7
	0011			00			
√	0012	LOOP	ADD C	81	1	1	4
√	0013		JNC NEXT	D2	3	3	10

Start From → 0000

Register	Value	7	6	5	4	3	2	1	0
Accumulator	23	0	0	1	0	0	0	1	1
Register B	00	0	0	0	0	0	0	0	0
Register C	05	0	0	0	0	0	1	0	1
Register D	00	0	0	0	0	0	0	0	0
Register E	00	0	0	0	0	0	0	0	0
Register H	30	0	0	1	1	0	0	0	0
Register L	01	0	0	0	0	0	0	0	1
Memory(M)	00	0	0	0	0	0	0	0	0

Resister	Value	S	Z	*	AC	*	P	*	CY
Flag Resister	54	0	1	0	1	0	1	0	0

Type	Value
Stack Pointer(SP)	0000
Memory Pointer (HL)	3001
Program Status Word(PSW)	2354
Program Counter(PC)	0021
Clock Cycle Counter	304
Instruction Counter	43

SOD	SID	INTR	TRAP	R7.5	R6.5	R5.5
0	0	0	0	0	0	0

For SIM instruction

SOD	SDE	*	R7...	MSE	M...	M...	M...
0	0	0	0	0	0	0	0

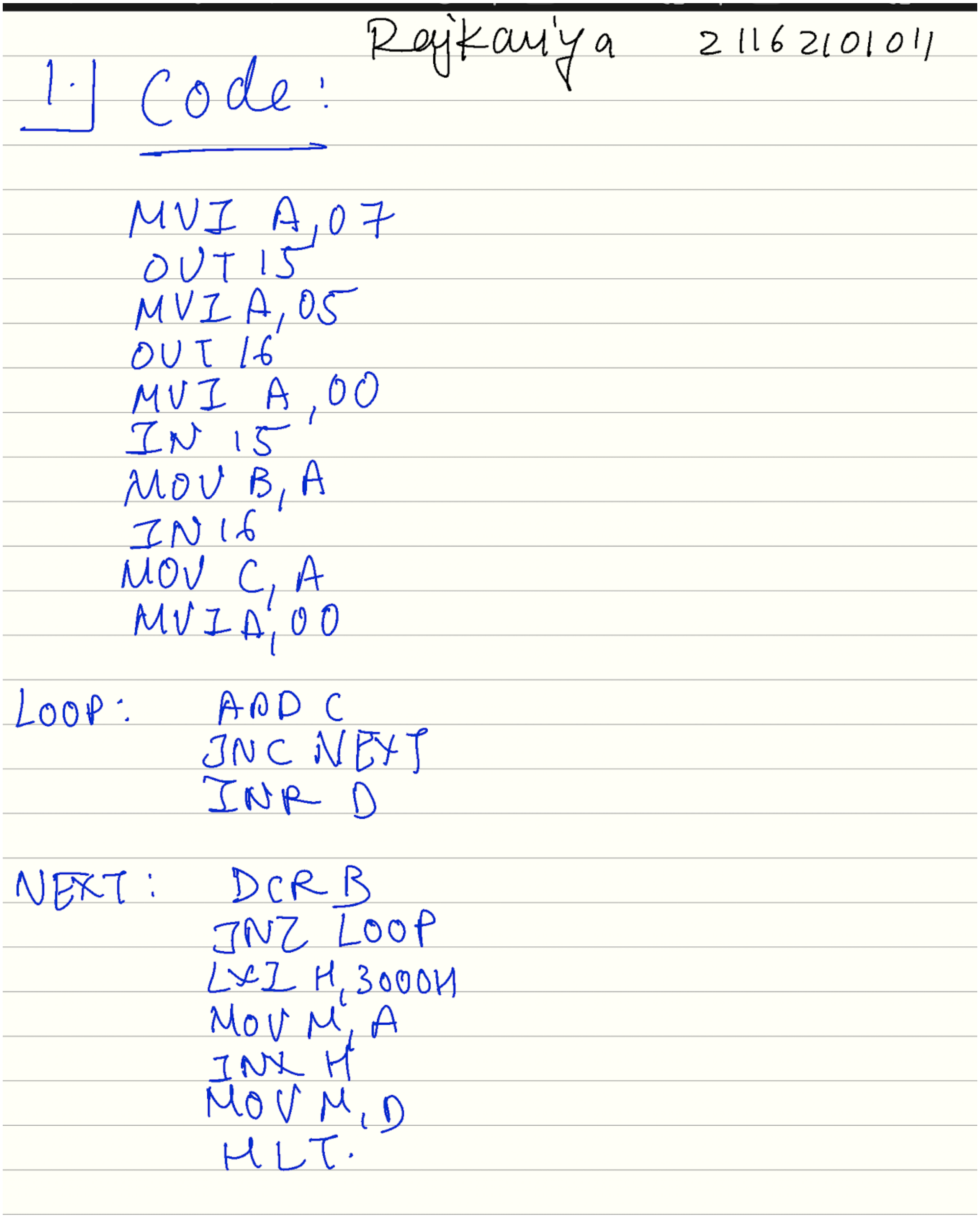
For RIM instruction

SID	I7.5	I6.5	I5.5	IE	M...	M...	M...
0	0	0	0	0	0	0	0

No. Converter Tool :

Hexadecimal	Decimal	Binary
0	0	0

Created by : Jubin Mitra



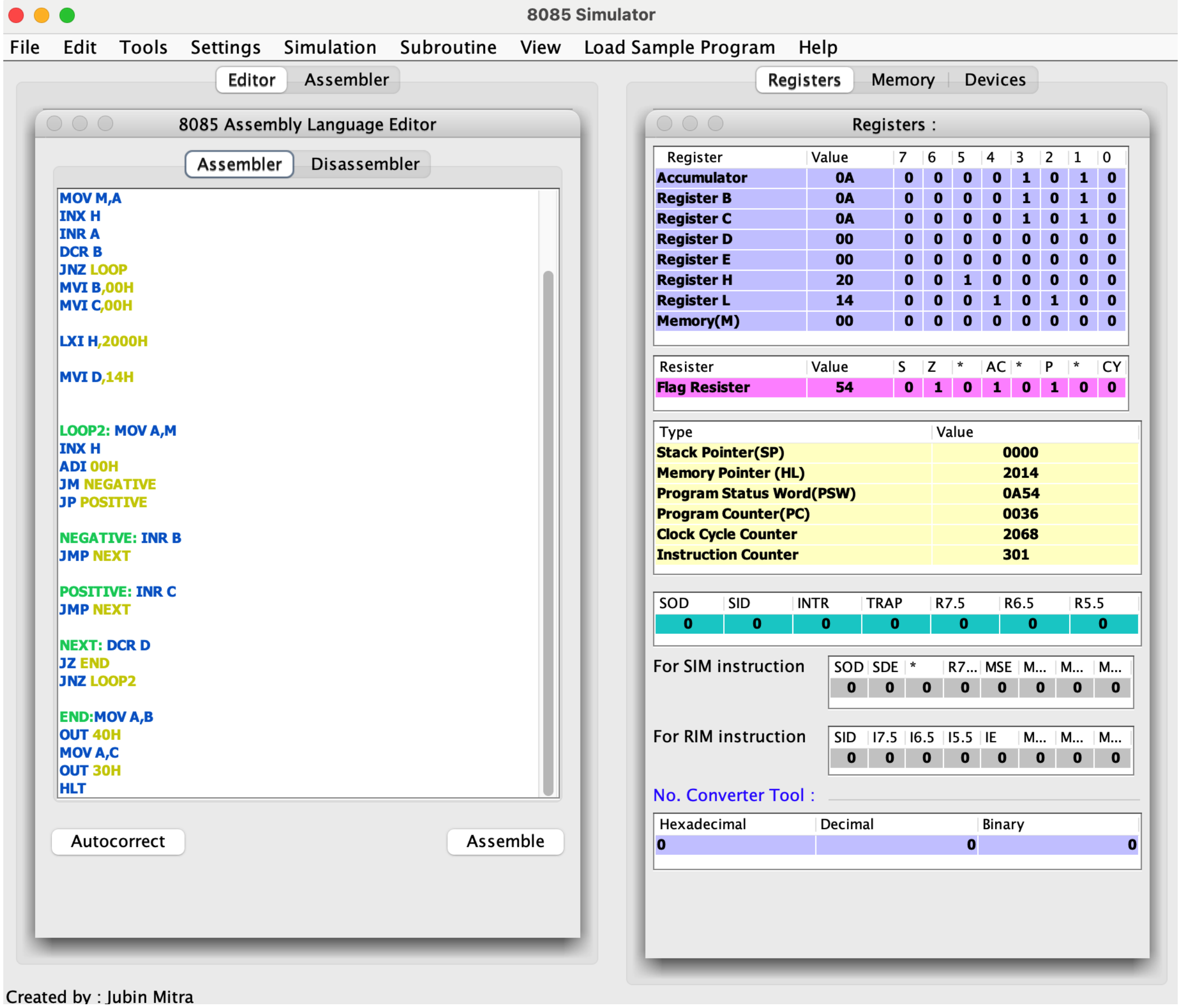
Rajkariya 21162101011

1.] Code:

```
MVI A,07
OUT 15
MVI A,05
OUT 16
MVI A,00
IN 15
MOV B,A
IN 16
MOV C,A
MVI A,00

LOOP:   AND C
        JNC NEXT
        INR D

NEXT:   DCR B
        JNZ LOOP
        LXI H,3000H
        MOV M,A
        INX H
        MOV M,D
        HLT.
```

2. Write a Program to count positive and negative numbers out of 20 numbers stored in memory. (Assume appropriate memory location

in your program and load 20 different bytes in memory using assembler directives). Display the count of positive nos on output port 30 H and negative numbers on 40 H.



```
MOV M,A
INX H
INR A
DCR B
JNZ LOOP
MVI B,00H
MVI C,00H

LXI H,2000H

MVI D,14H


LOOP2: MOV A,M
INX H
ADI 00H
JM NEGATIVE
JP POSITIVE

NEGATIVE: INR B
JMP NEXT

POSITIVE: INR C
JMP NEXT

NEXT: DCR D
JZ END
JNZ LOOP2

END:MOV A,B
OUT 40H
MOV A,C
OUT 30H
HLT
```

Register	Value	7	6	5	4	3	2	1	0
Accumulator	0A	0	0	0	0	1	0	1	0
Register B	0A	0	0	0	0	1	0	1	0
Register C	0A	0	0	0	0	1	0	1	0
Register D	00	0	0	0	0	0	0	0	0
Register E	00	0	0	0	0	0	0	0	0
Register H	20	0	0	1	0	0	0	0	0
Register L	14	0	0	0	1	0	1	0	0
Memory(M)	00	0	0	0	0	0	0	0	0

Resister	Value	S	Z	*	AC	*	P	*	CY
Flag Resister	54	0	1	0	1	0	1	0	0

Type	Value
Stack Pointer(SP)	0000
Memory Pointer (HL)	2014
Program Status Word(PSW)	0A54
Program Counter(PC)	0036
Clock Cycle Counter	2068
Instruction Counter	301

SOD	SID	INTR	TRAP	R7.5	R6.5	R5.5
0	0	0	0	0	0	0

For SIM instruction

SOD	SDE	*	R7...	MSE	M...	M...	M...
0	0	0	0	0	0	0	0

For RIM instruction

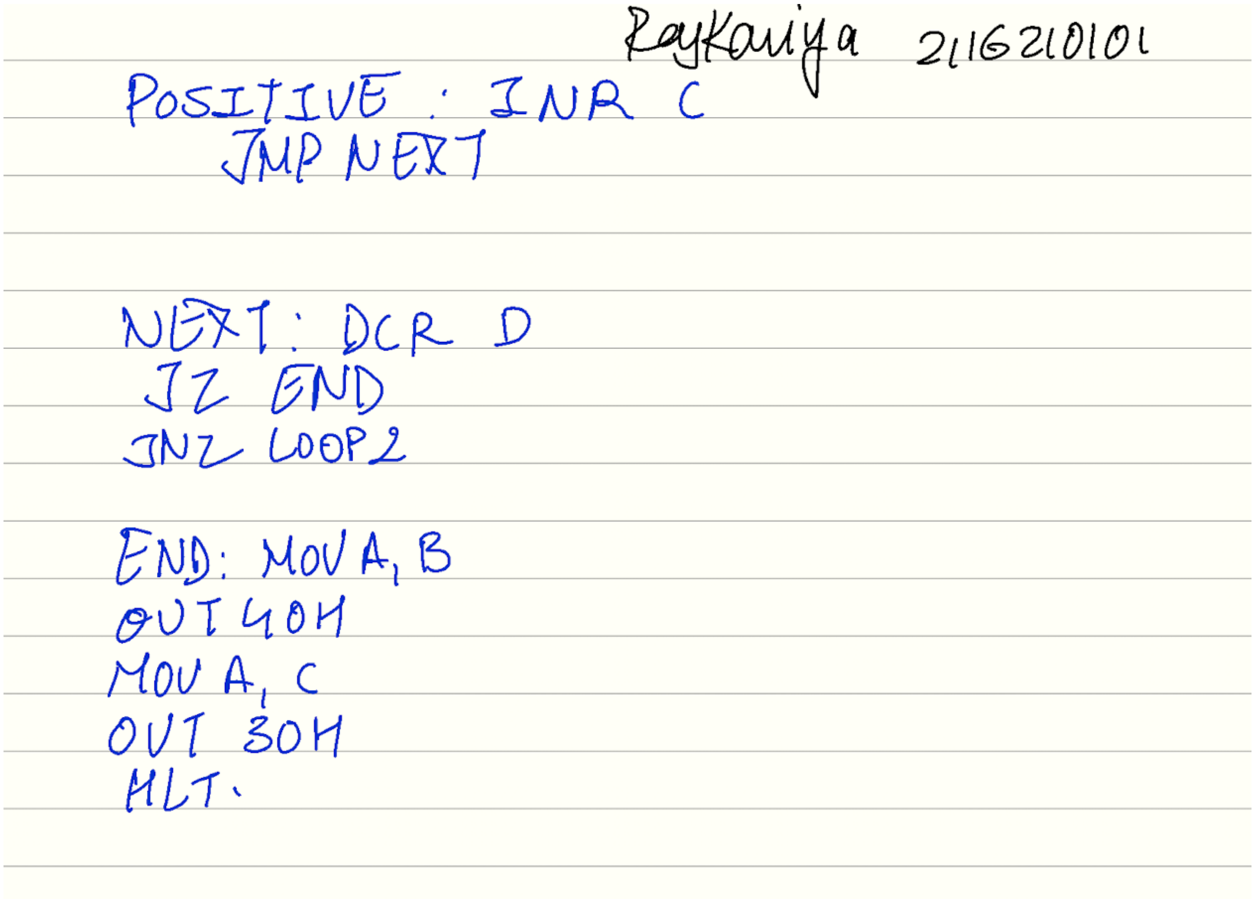
SID	I7.5	I6.5	I5.5	IE	M...	M...	M...
0	0	0	0	0	0	0	0

Hexadecimal	Decimal	Binary
0	0	0

Rajkariya 21162101011

2.)

```
MVI A, 76H
MVI B, 14H

LXI H, 2000H

LOOP:
MOV M, A
INX H
INR A
DCR B
JNZ LOOP
MVI B, 00H
MVI C, 00H

LXI H, 2000H
MVI D, 14H

LOOP2 : MOV A, M
INX H
ADI 00H
JM NEGATIVE
JP POSITIVE

NEGATIVE : INR B
JMP NEXT
```

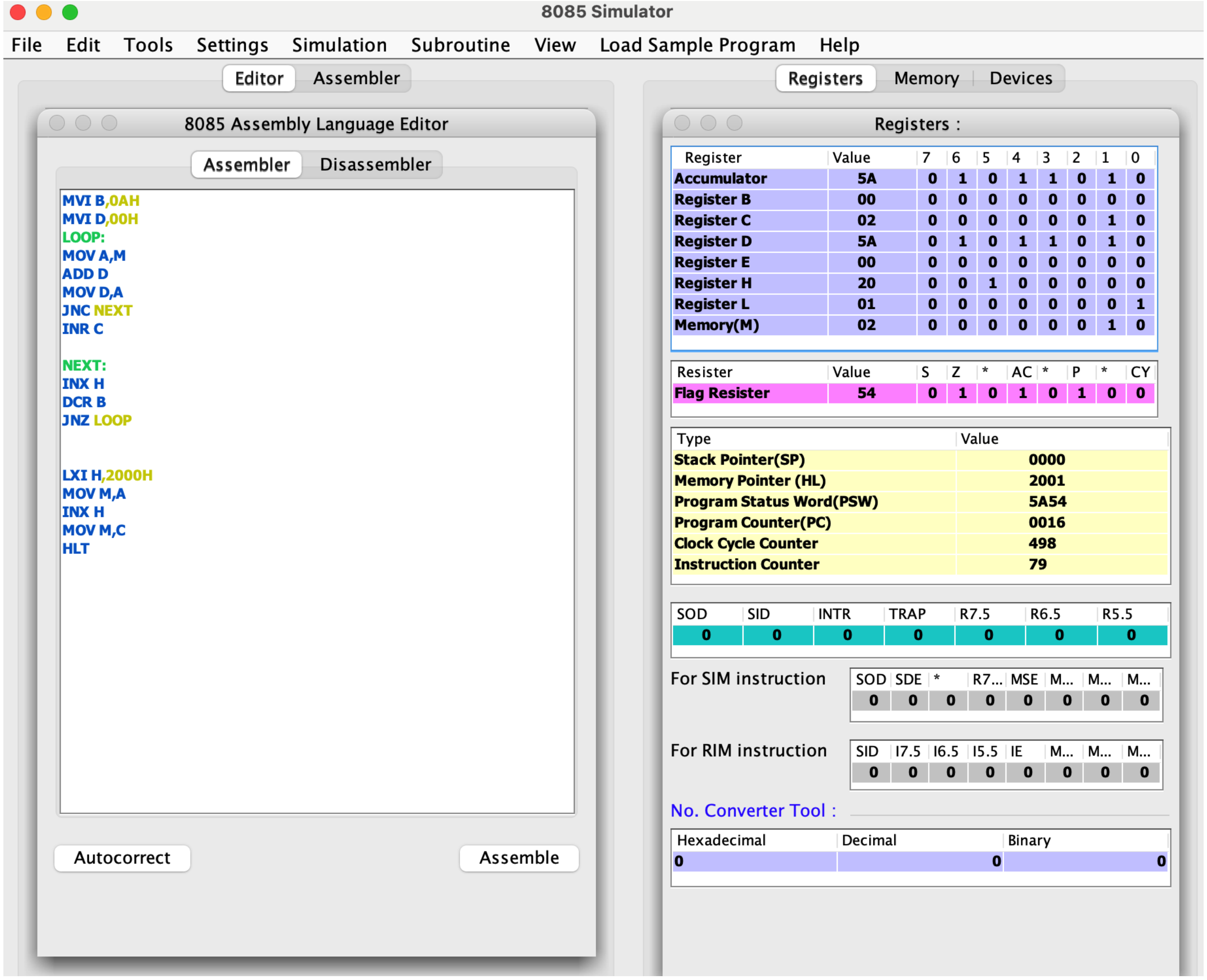


RajKariya 2116210101

```
POSITIVE : INR C
    JMP NEXT


NEXT: DCR D
 JZ END
JNZ LOOP2

END: MOV A, B
OUT 40H
MOV A, C
OUT 30H
HLT.
```

3. Write a program to add 10 bytes stored in a string starting from 1000H .Store result at 2000H (LSB), 2001H (MSB).

KARIYA RAJ 21162101011



```
MVI B,0AH
MVI D,00H
LOOP:
MOV A,M
ADD D
MOV D,A
JNC NEXT
INR C

NEXT:
INX H
DCR B
JNZ LOOP


LXI H,2000H
MOV M,A
INX H
MOV M,C
HLT
```

Register	Value	7	6	5	4	3	2	1	0
Accumulator	5A	0	1	0	1	1	0	1	0
Register B	00	0	0	0	0	0	0	0	0
Register C	02	0	0	0	0	0	0	1	0
Register D	5A	0	1	0	1	1	0	1	0
Register E	00	0	0	0	0	0	0	0	0
Register H	20	0	0	1	0	0	0	0	0
Register L	01	0	0	0	0	0	0	0	1
Memory(M)	02	0	0	0	0	0	0	1	0

Resister	Value	S	Z	*	AC	*	P	*	CY
Flag Resister	54	0	1	0	1	0	1	0	0

Type	Value
Stack Pointer(SP)	0000
Memory Pointer (HL)	2001
Program Status Word(PSW)	5A54
Program Counter(PC)	0016
Clock Cycle Counter	498
Instruction Counter	79

SOD	SID	INTR	TRAP	R7.5	R6.5	R5.5
0	0	0	0	0	0	0

For SIM instruction

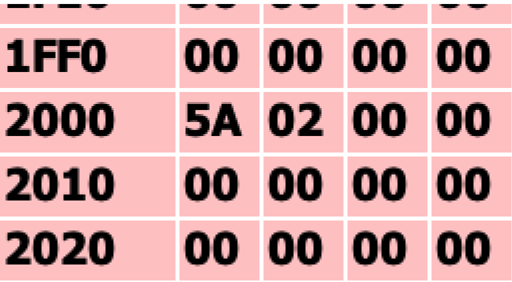
SOD	SDE	*	R7...	MSE	M...	M...	M...
0	0	0	0	0	0	0	0

For RIM instruction

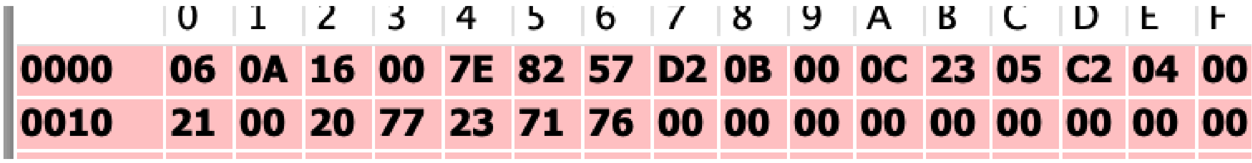
SID	I7.5	I6.5	I5.5	IE	M...	M...	M...
0	0	0	0	0	0	0	0

No. Converter Tool :

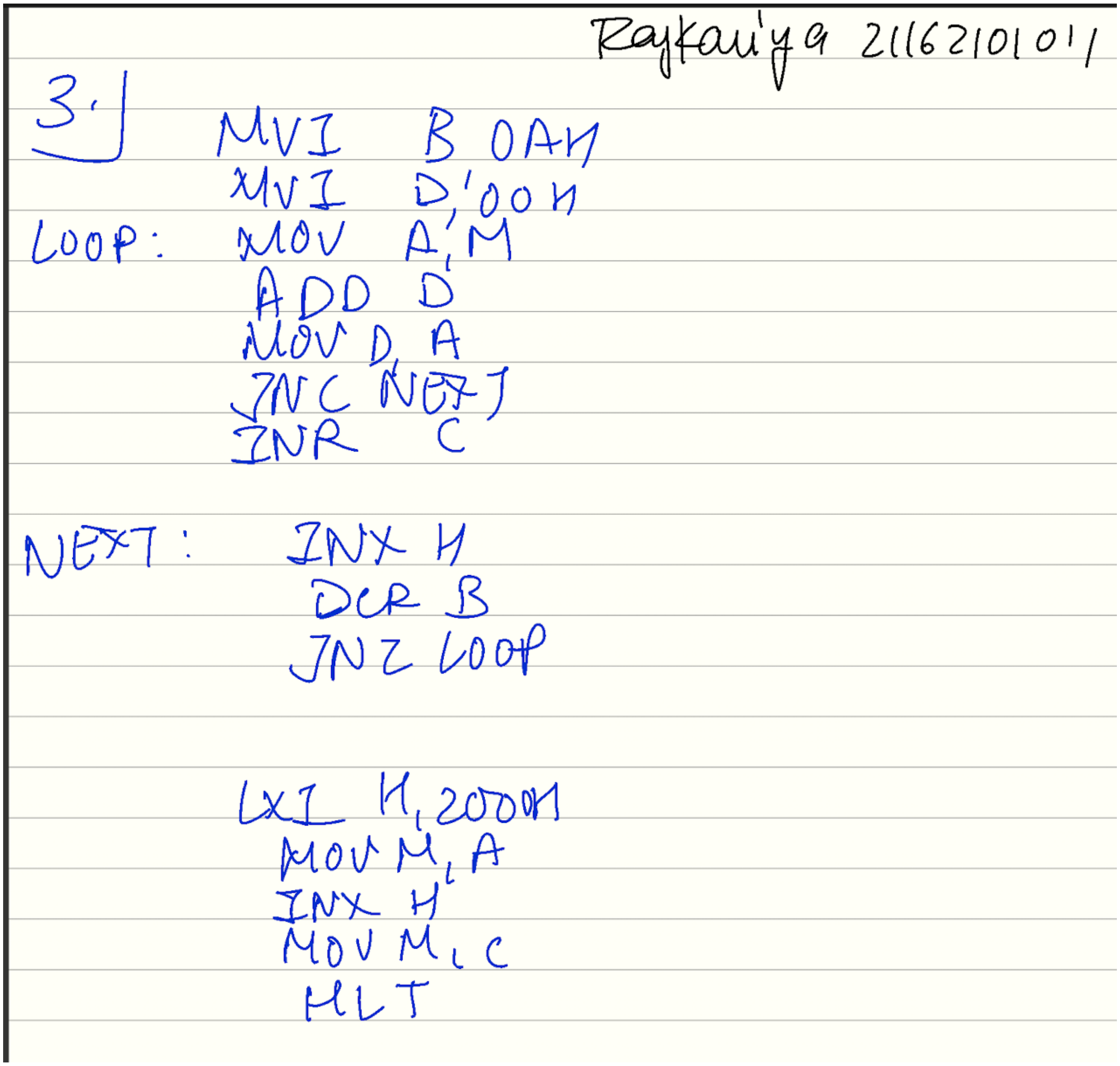
Hexadecimal	Decimal	Binary
0	0	0



1FF0	00	00	00	00
2000	5A	02	00	00
2010	00	00	00	00
2020	00	00	00	00



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	06	0A	16	00	7E	82	57	D2	0B	00	0C	23	05	C2	04	00
0010	21	00	20	77	23	71	76	00	00	00	00	00	00	00	00	00

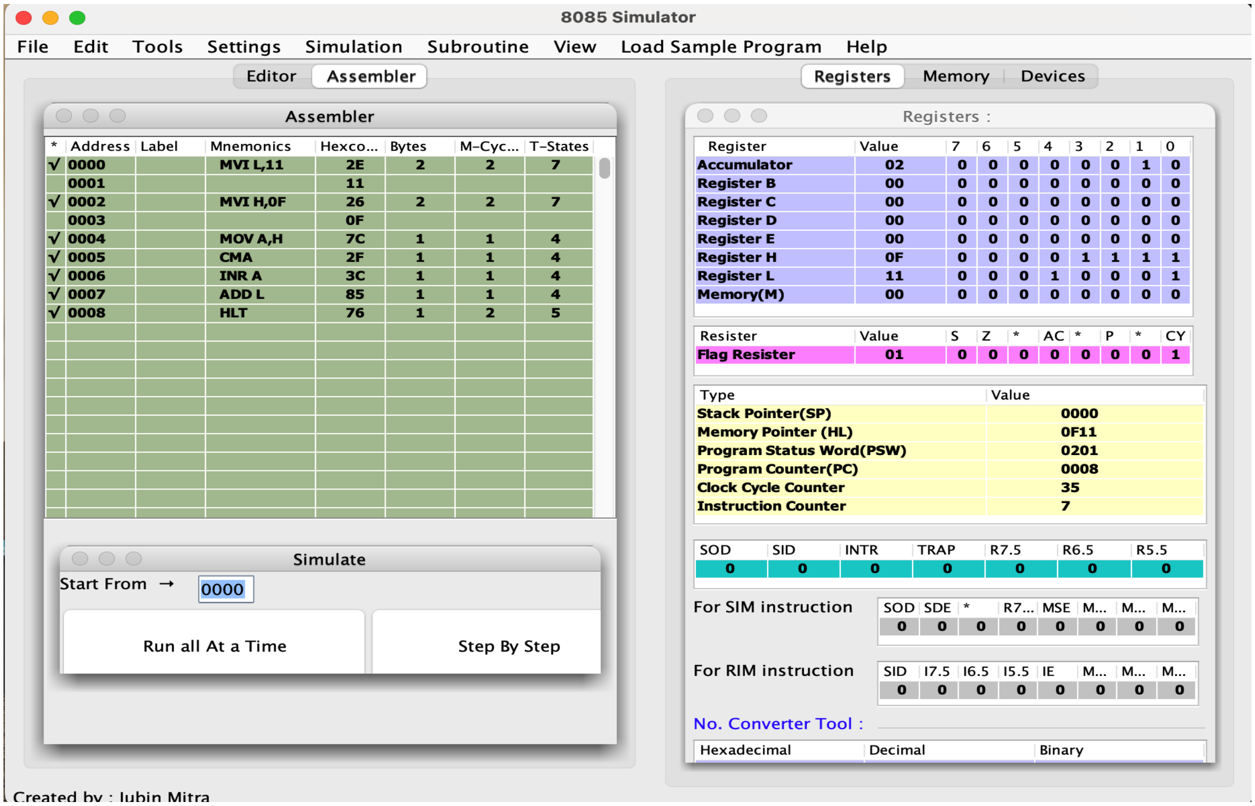


Rajkariya 21162101011

3.

```
        MVI  B 0AH
        MVI  D,00H
LOOP:   MOV  A,M
        ADD  D
        MOV  D, A
        JNC  NEXT
        INR  C

NEXT:   INX H
        DCR B
        JNZ LOOP

        LXI H,2000H
        MOV M,A
        INX H
        MOV M,C
        HLT
```

4. Write a program to subtract the contents of register H from register L without using any of the subtract instructions.
(a) Apply 2's Complement method.



*	Address	Label	Mnemonics	Hexco...	Bytes	M-Cyc...	T-States
√	0000		MVI L,11	2E	2	2	7
	0001			11			
√	0002		MVI H,0F	26	2	2	7
	0003			0F			
√	0004		MOV A,H	7C	1	1	4
√	0005		CMA	2F	1	1	4
√	0006		INR A	3C	1	1	4
√	0007		ADD L	85	1	1	4
√	0008		HLT	76	1	2	5

Start From → 0000

Register	Value	7	6	5	4	3	2	1	0
Accumulator	02	0	0	0	0	0	0	1	0
Register B	00	0	0	0	0	0	0	0	0
Register C	00	0	0	0	0	0	0	0	0
Register D	00	0	0	0	0	0	0	0	0
Register E	00	0	0	0	0	0	0	0	0
Register H	0F	0	0	0	0	1	1	1	1
Register L	11	0	0	0	1	0	0	0	1
Memory(M)	00	0	0	0	0	0	0	0	0

Resister	Value	S	Z	*	AC	*	P	*	CY
Flag Resister	01	0	0	0	0	0	0	0	1

Type	Value
Stack Pointer(SP)	0000
Memory Pointer (HL)	0F11
Program Status Word(PSW)	0201
Program Counter(PC)	0008
Clock Cycle Counter	35
Instruction Counter	7

SOD	SID	INTR	TRAP	R7.5	R6.5	R5.5
0	0	0	0	0	0	0

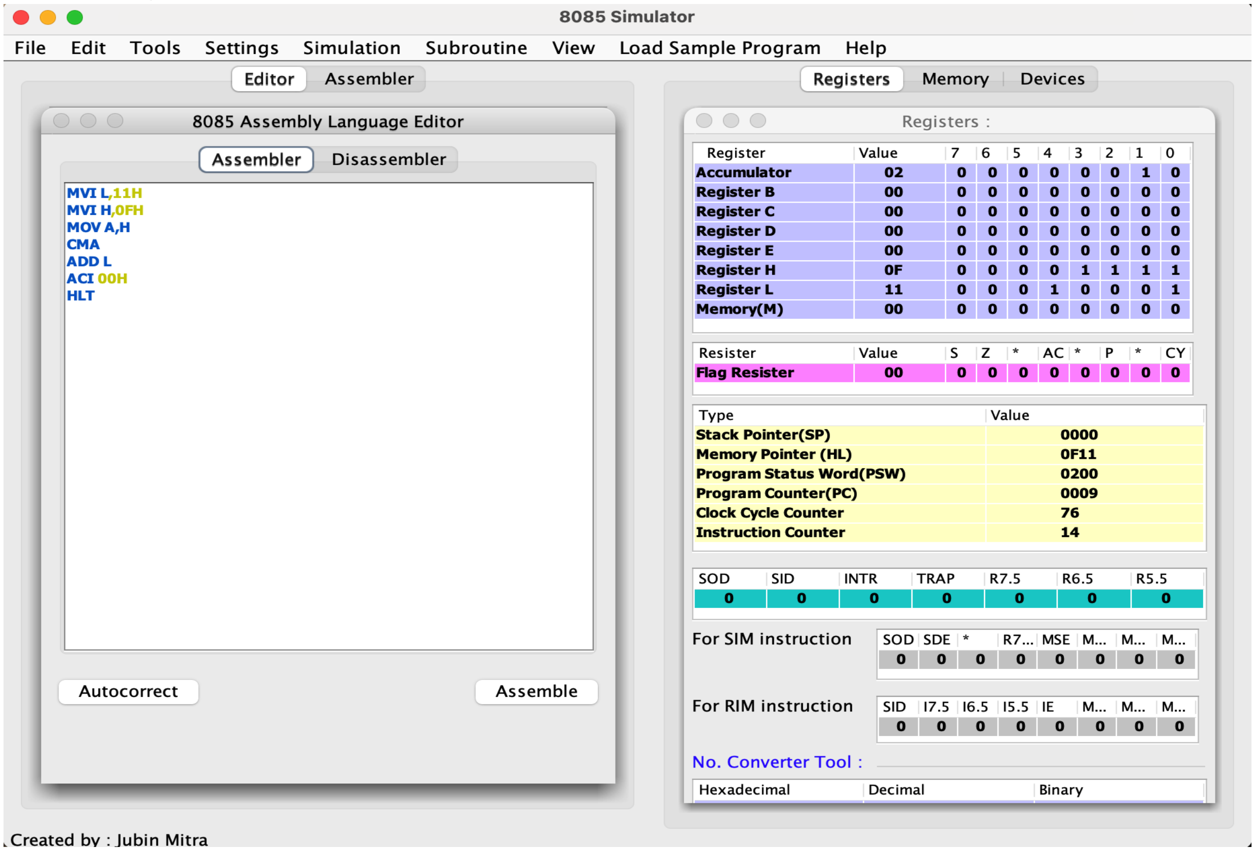
For SIM instruction

SOD	SDE	*	R7...	MSE	M...	M...	M...
0	0	0	0	0	0	0	0

For RIM instruction

SID	I7.5	I6.5	I5.5	IE	M...	M...	M...
0	0	0	0	0	0	0	0

(b) Apply 1's Complement method.



```
MVI L,11H
MVI H,0FH
MOV A,H
CMA
ADD L
ACI 00H
HLT
```

Register	Value	7	6	5	4	3	2	1	0
Accumulator	02	0	0	0	0	0	0	1	0
Register B	00	0	0	0	0	0	0	0	0
Register C	00	0	0	0	0	0	0	0	0
Register D	00	0	0	0	0	0	0	0	0
Register E	00	0	0	0	0	0	0	0	0
Register H	0F	0	0	0	0	1	1	1	1
Register L	11	0	0	0	1	0	0	0	1
Memory(M)	00	0	0	0	0	0	0	0	0

Resister	Value	S	Z	*	AC	*	P	*	CY
Flag Resister	00	0	0	0	0	0	0	0	0

Type	Value
Stack Pointer(SP)	0000
Memory Pointer (HL)	0F11
Program Status Word(PSW)	0200
Program Counter(PC)	0009
Clock Cycle Counter	76
Instruction Counter	14

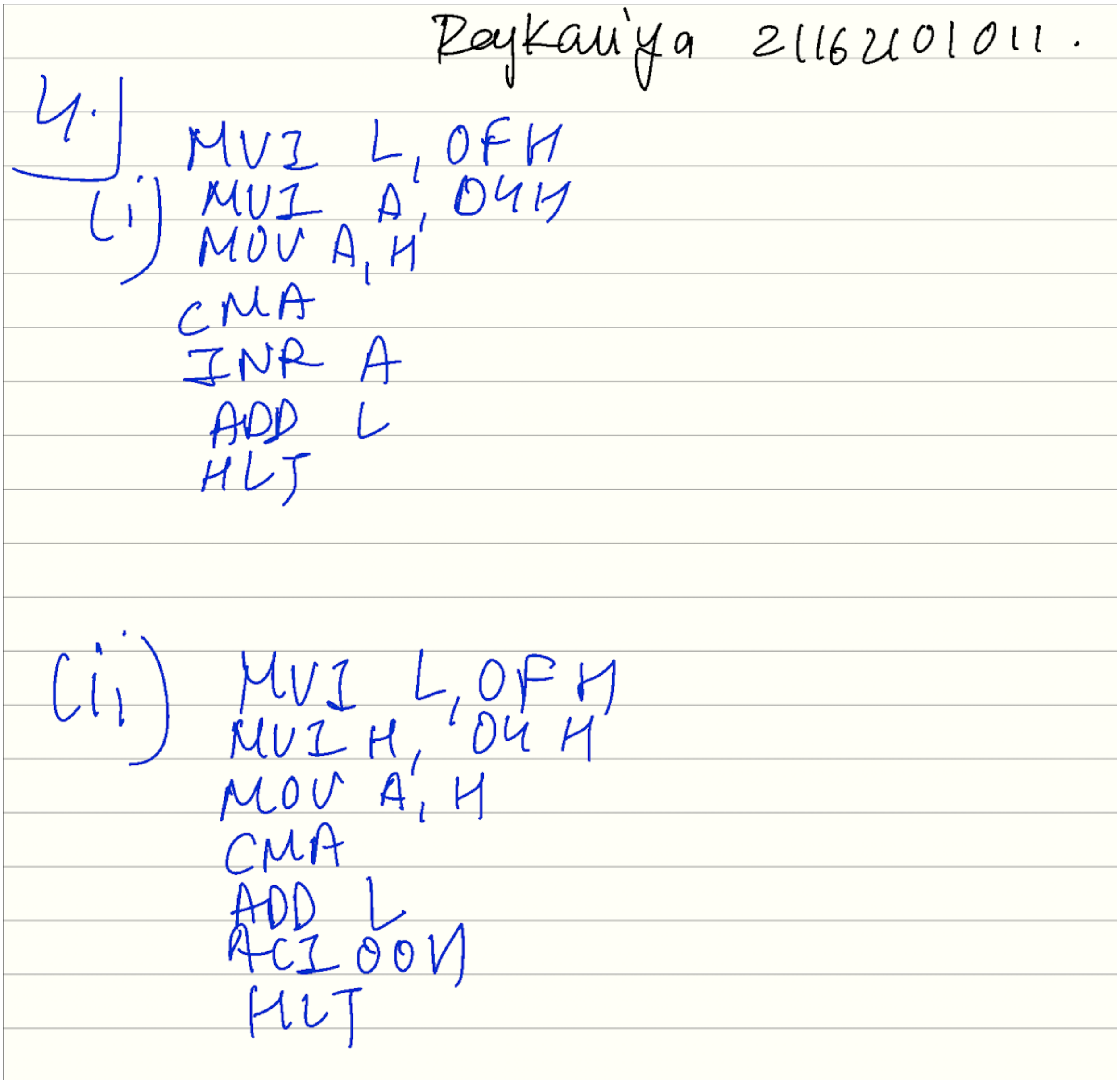
SOD	SID	INTR	TRAP	R7.5	R6.5	R5.5
0	0	0	0	0	0	0

For SIM instruction

SOD	SDE	*	R7...	MSE	M...	M...	M...
0	0	0	0	0	0	0	0

For RIM instruction

SID	I7.5	I6.5	I5.5	IE	M...	M...	M...
0	0	0	0	0	0	0	0



Rajkariya 21162101011.

4.

(i)

```
MVI L, 0FH
MVI A, 04H
MOV A, H
CMA
INR A
ADD L
HLT
```

(ii)

```
MVI L, 0FH
MVI H, 04 H
MOV A, H
CMA
ADD L
ACI 00H
HLT
```