Arithmetic-operation-using-8086

8086 Assembly Language Programs for Arithmetic Operations

AIM

To write and execute Assembly Language Programs to perform arithmetic operations for the 8086 microprocessor.

APPARATUS REQUIRED

- Personal Computer with MASM Software

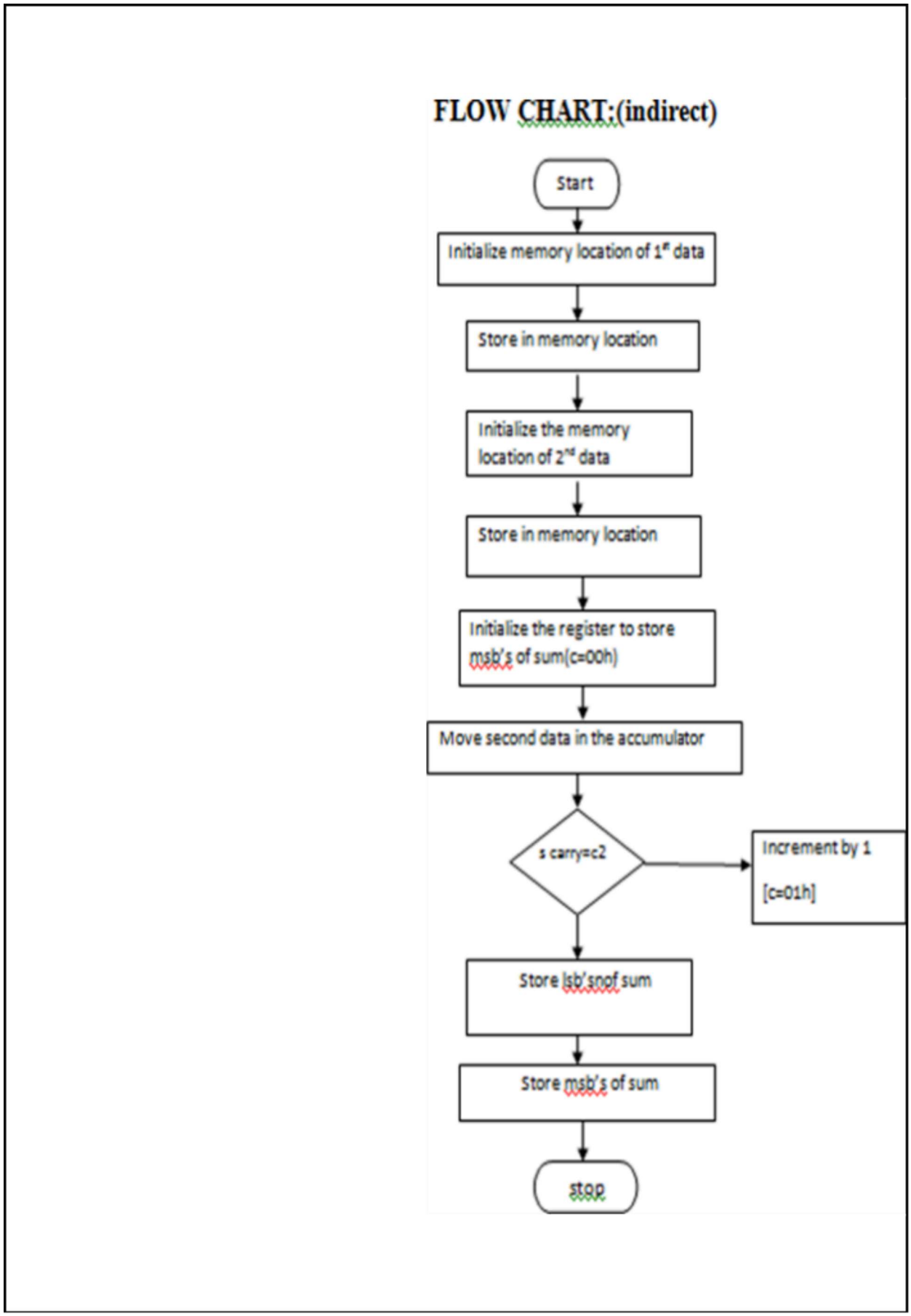
1. ADDITION

Algorithm

1. Initialize memory location in HL register.
2. Store 1st data.
3. Increment HL to enter 2nd data.
4. Move 2nd number to accumulator.
5. Decrement HL.

6. Add value in memory with accumulator.
7. Store result.
8. Stop.

FLOW CHART


FLOW CHART:(indirect)
Start
Initialize memory location of 1st data
Store in memory location
Initialize the memory location of 2nd data
Store in memory location
Initialize the register to store msb's of sum(c=00h)
Move second data in the accumulator
s carry=c2
Increment by 1
[c=01h]
Store lsb'snof sum
Store msb's of sum
stop

DIRECT ADDITION :

Program

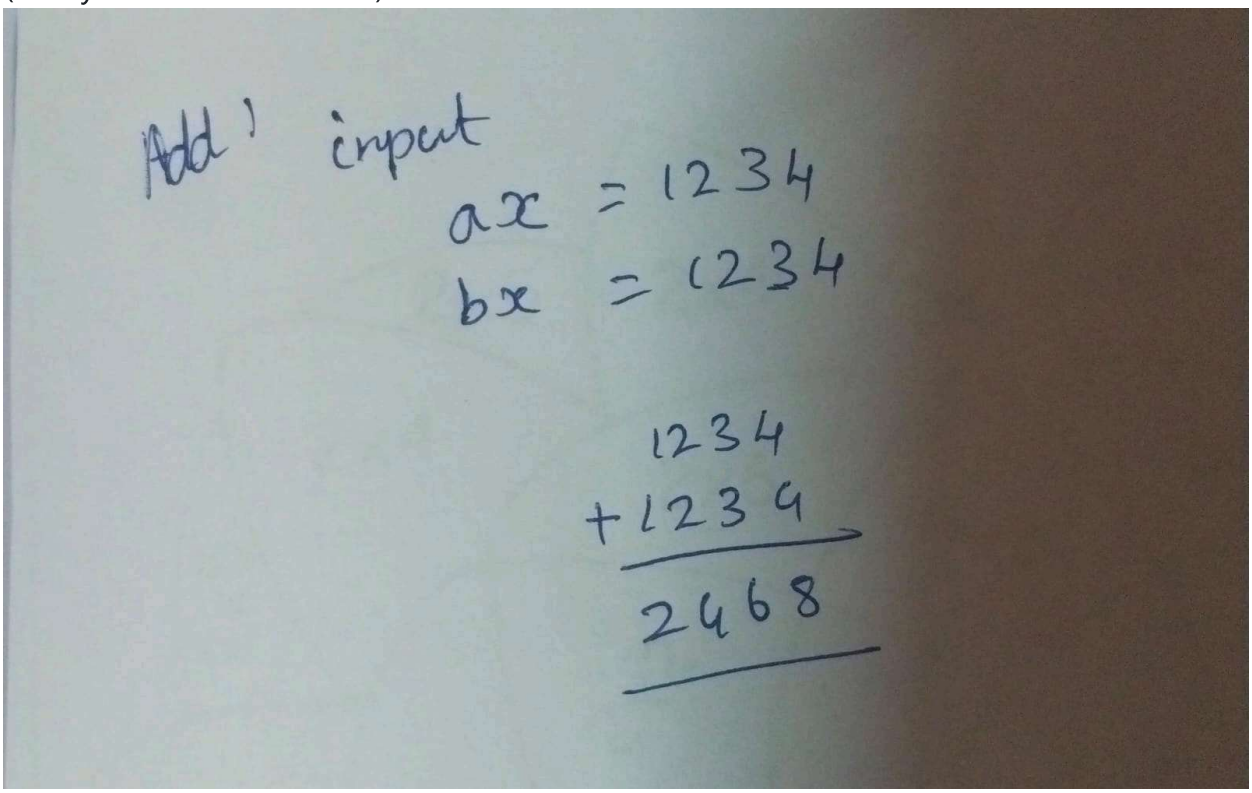
```
CODE SEGMENT
ASSUME CS: CODE, DS: CODE
ORG 1000H
MOV SI,2000H
MOV CL,00H
MOV AX,[SI]
MOV BX,[SI+02H]
ADD AX,BX
JNC L1
INC CL
L1:
MOV [SI+04H],AX
MOV [SI+06H],CL
MOV AH,4CH
INT 21H
CODE ENDS
END
```

Output Table

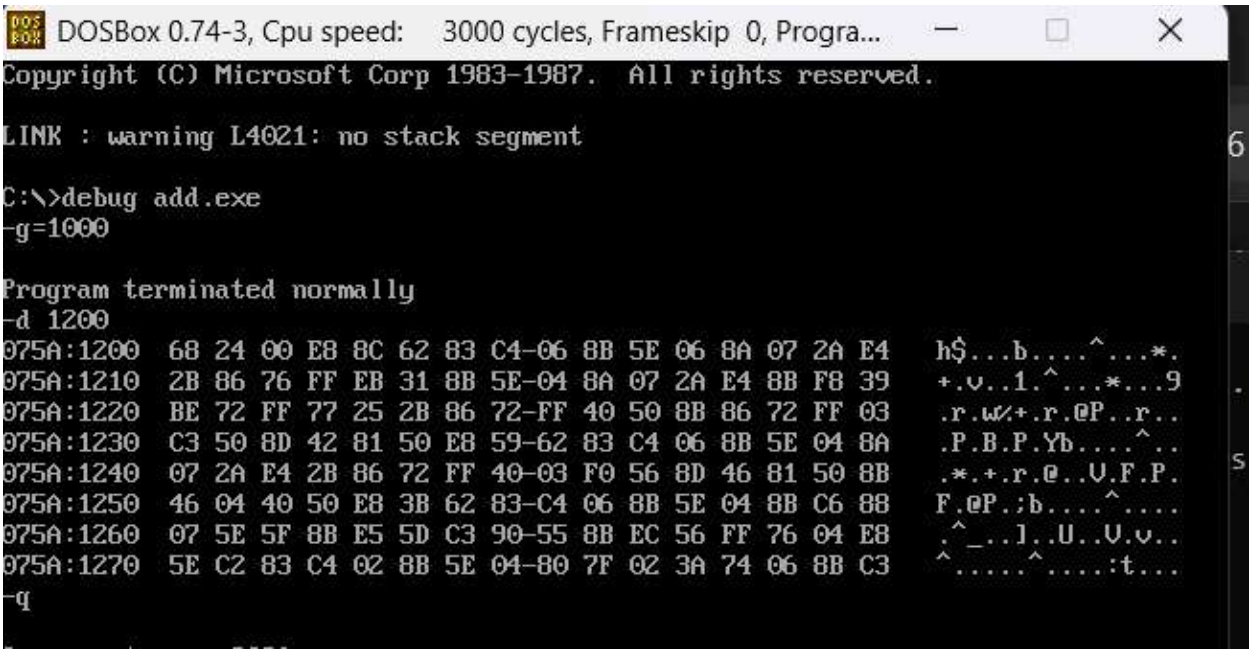
MEMORY LOCATION (INPUT)	MEMORY LOCATION (OUTPUT)
1200	12
1201	34
1202	12
1203	34
1204	24
1205	68

Manual Calculations :

(Add your calculation here)



OUTPUT IMAGE FROM MASM SOFTWARE :



INDIRECT ADDITION :

PROGRAM :

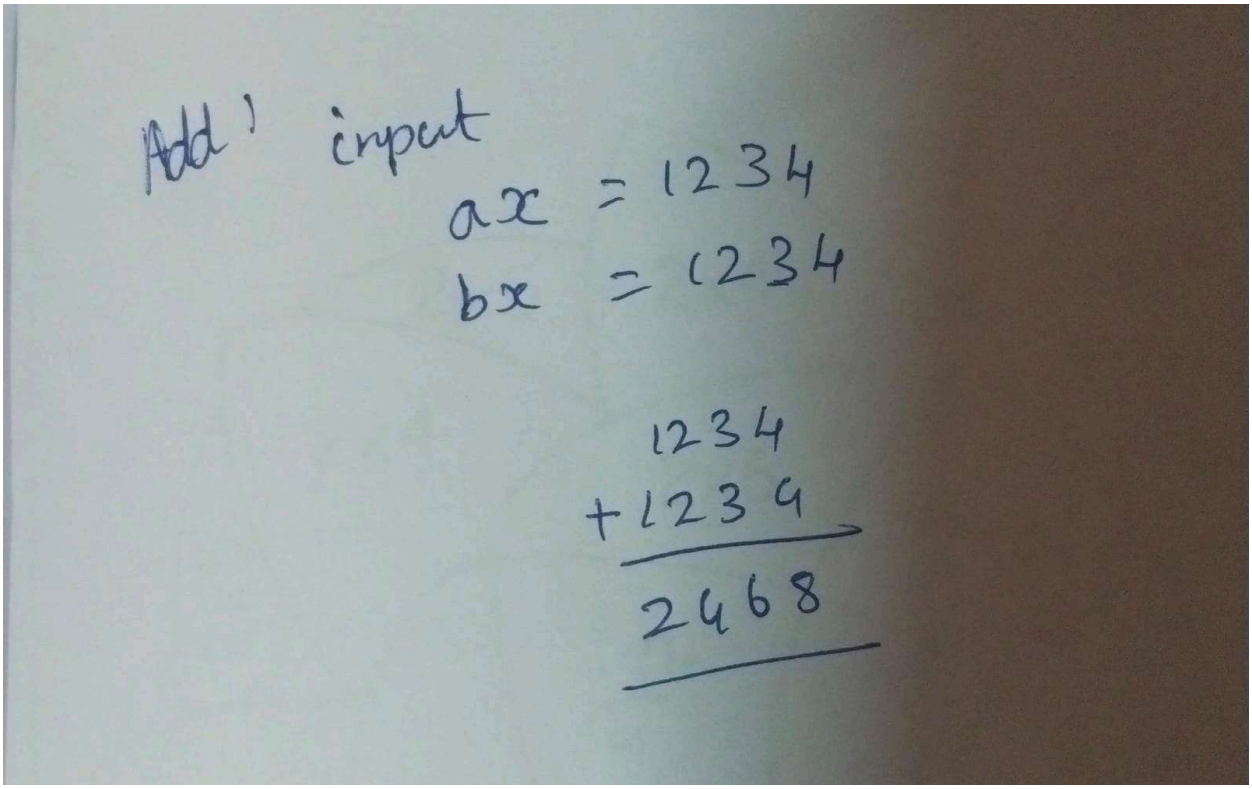
```
CODE SEGMENT
ASSUME CS: CODE, DS:CODE
ORG 1000H
MOV SI,2000H
```

```
MOV CL,00H
MOV AX,[SI]
MOV BX,[SI+02H]
ADD AX,BX
JNC L1
INC CL
L1:  MOV [SI+04H],AX
MOV [SI+06H],CL
MOV AH,4CH
INT 21H
CODE ENDS
END
```

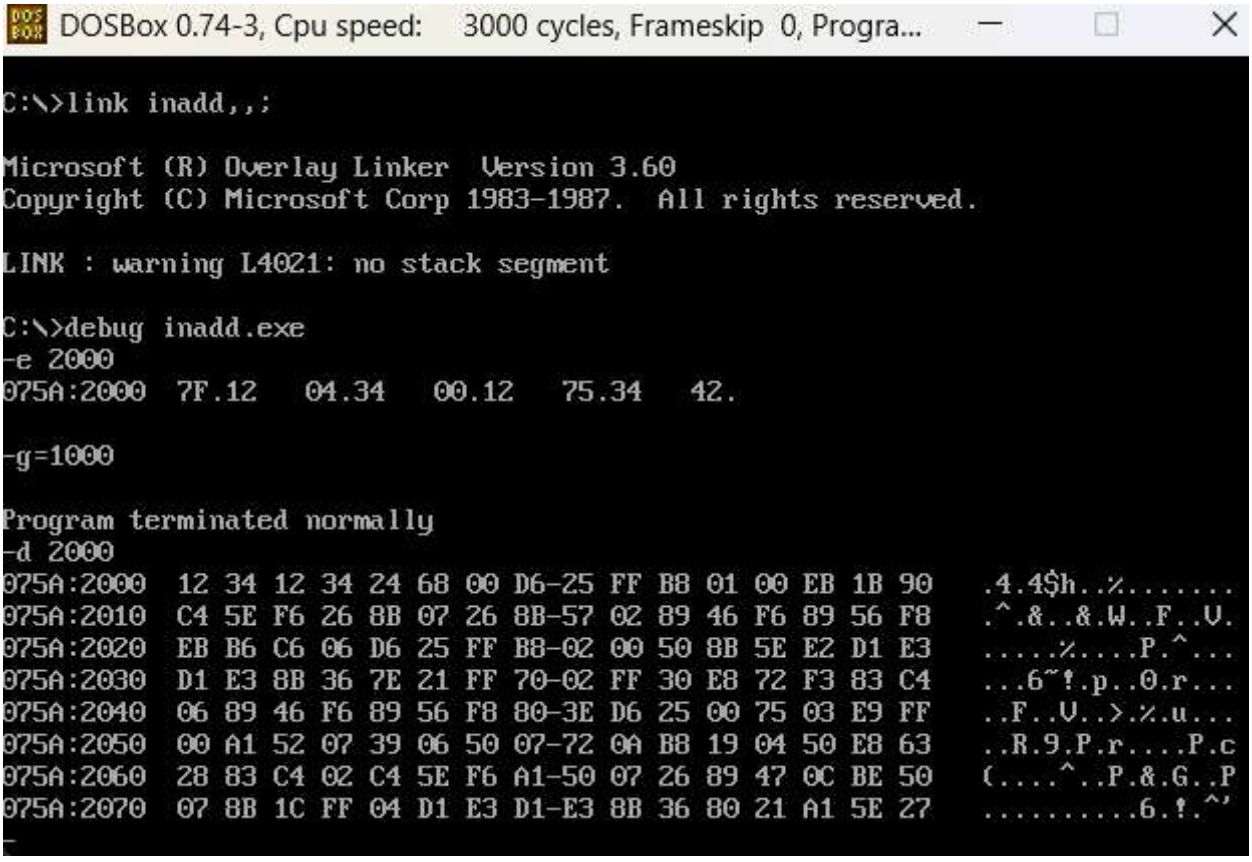
OUTPUT TABLE :

MEMORY LOCATION (INPUT)	MEMORY LOCATION (OUTPUT)
1200	78
1201	88
1202	23
1203	02
1204	9C
1205	8A

MANUAL CALCULATIONS :



OUTPUT IMAGE FROM MASM SOFTWARE :



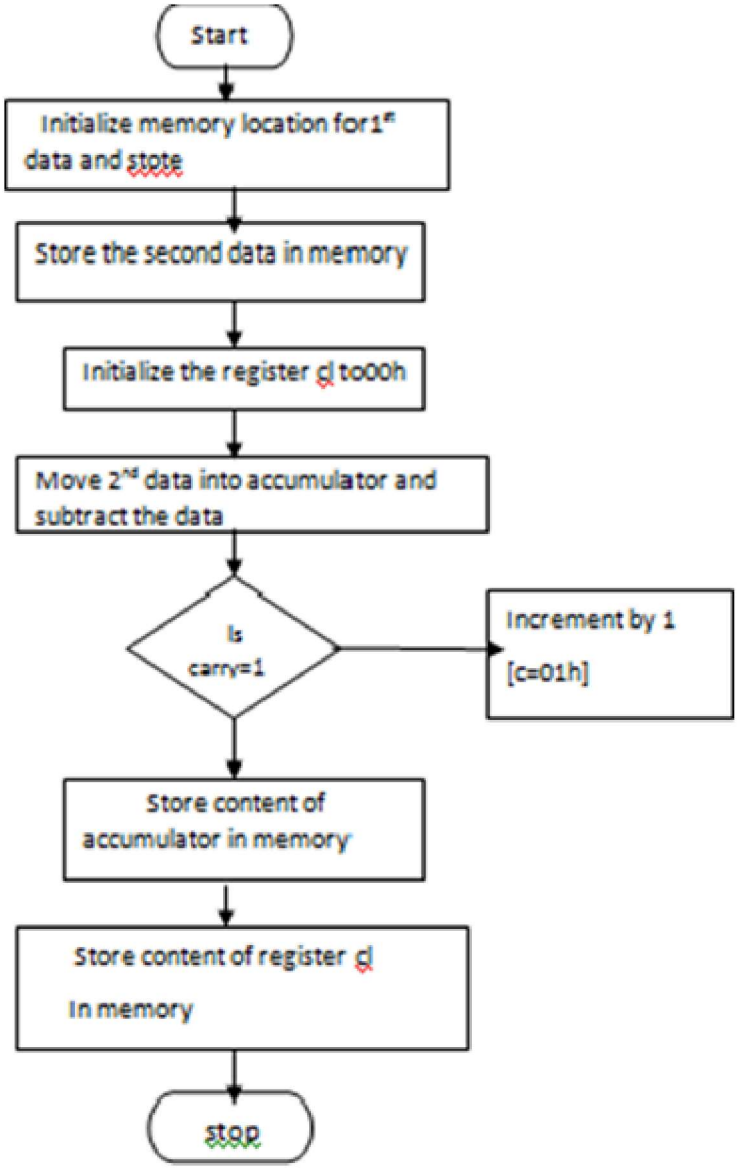
2. SUBTRACTION

Algorithm

1. Initialize memory and store 1st data.

2. Increment to get 2nd data.
3. Move 2nd data to accumulator.
4. Subtract memory content.
5. Store result.

FLOWCHART



Program

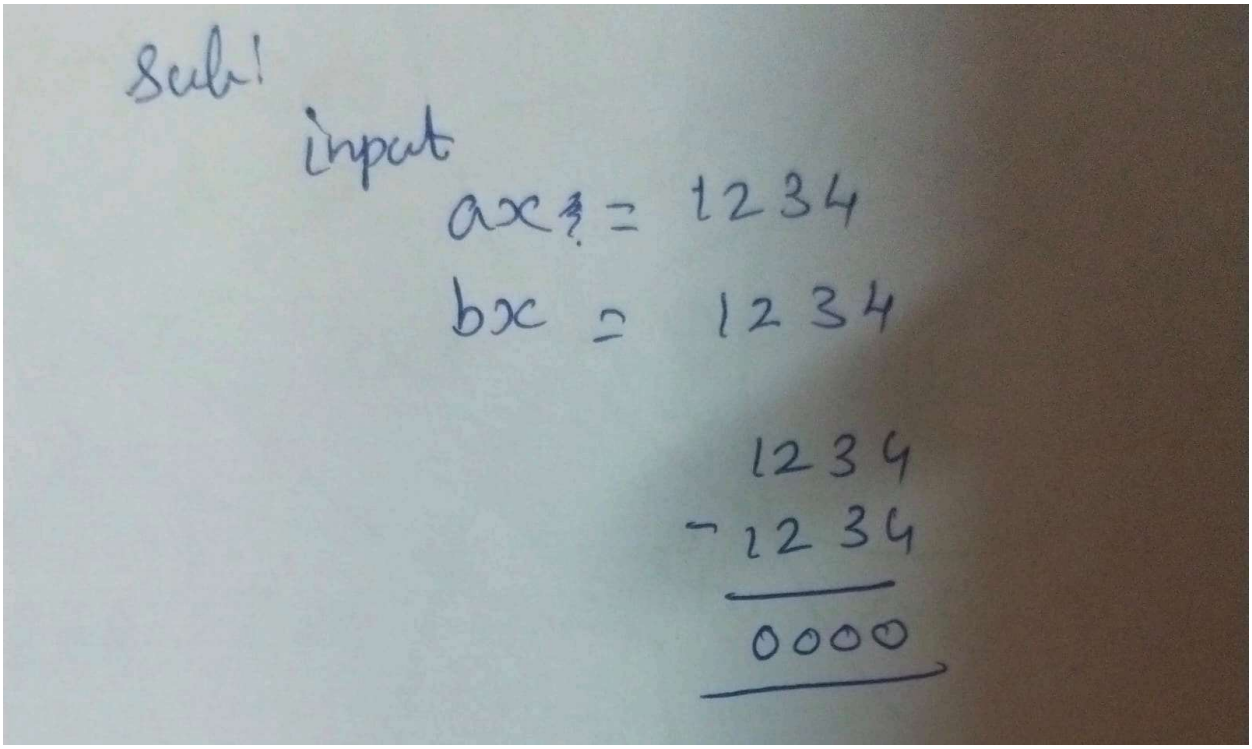
```
CODE SEGMENT
ASSUME CS: CODE, DS: CODE
ORG 1000H
MOV SI,2000H
MOV CL,00H
MOV AX,[SI]
MOV BX,[SI+02H]
SUB AX,BX
JNC L1
INC CL
L1:
MOV [SI+04H],AX
MOV [SI+06H],CL
MOV AH,4CH
INT 21H
CODE ENDS
END
```

Output Table

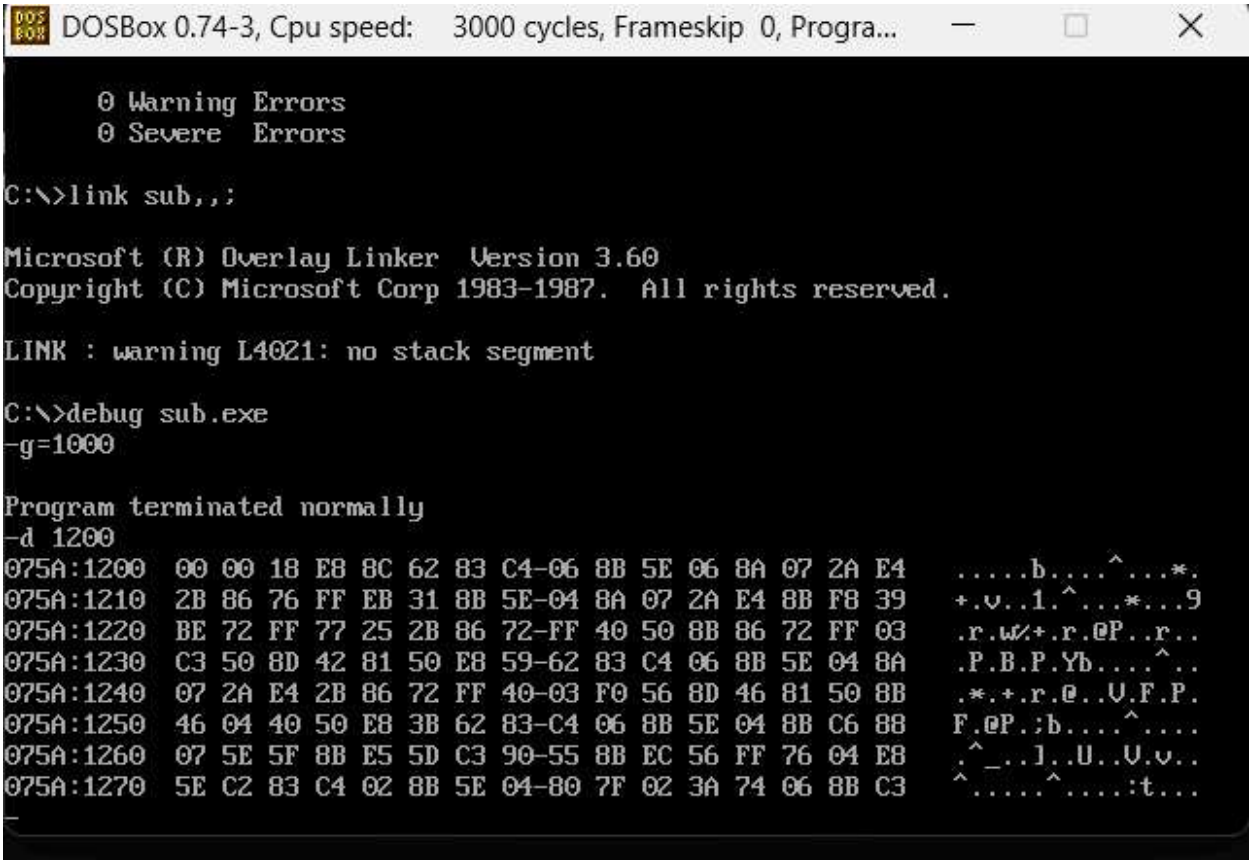
MEMORY LOCATION (INPUT)	MEMORY LOCATION (OUTPUT)
1200	12
1201	34
1202	12
1203	34
1204	00
1205	00

Manual Calculations

(Add your calculation here)



OUTPUT SCREEN FROM MASM SOFTWARE :



INDIRECT SUBTRACTION :

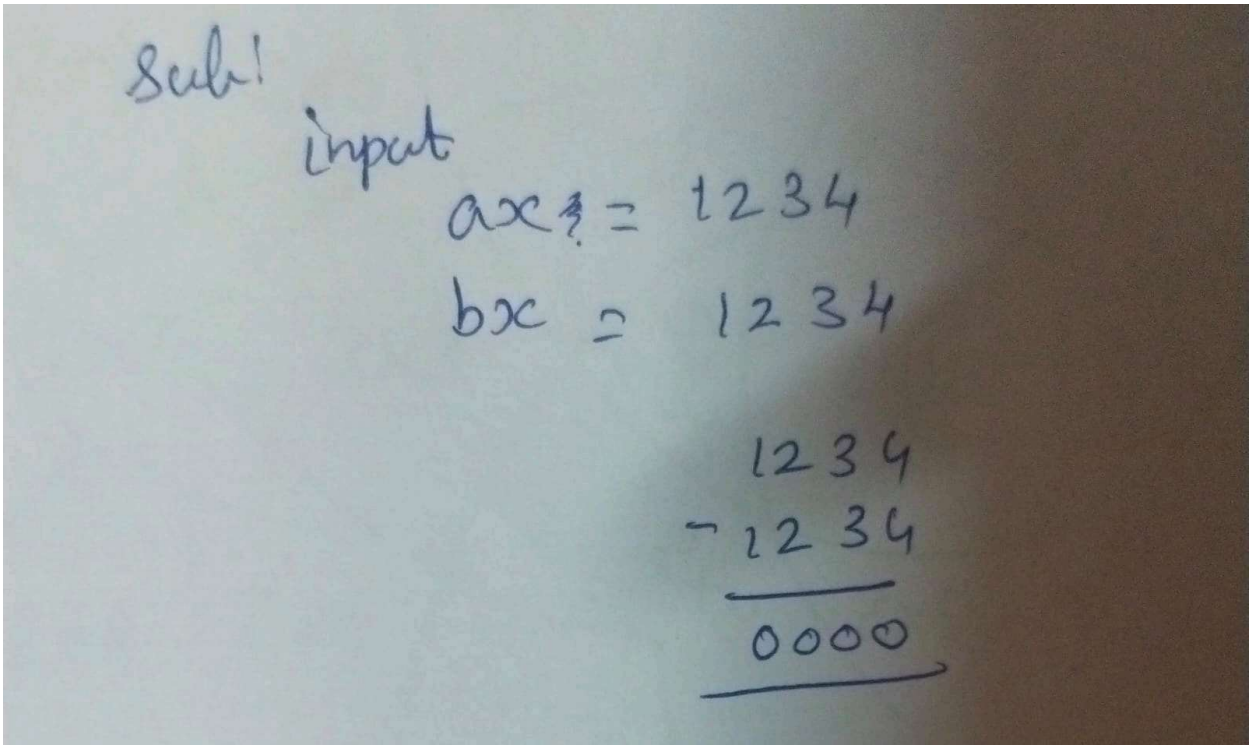
###PROGRAM :

```
ASSUME CS: CODE,DS:CODE
ORG 1000H
MOV SI,2000H
MOV CL,00H
MOV AX,[SI]
MOV BX,[SI+02H]
SUB AX, BX
JNC DOWN
INC CL
DOWN: MOV [SI+04H],AX
MOV [SI+06H],CL
MOV AH,4CH
INT 21H
CODE ENDS
END
```

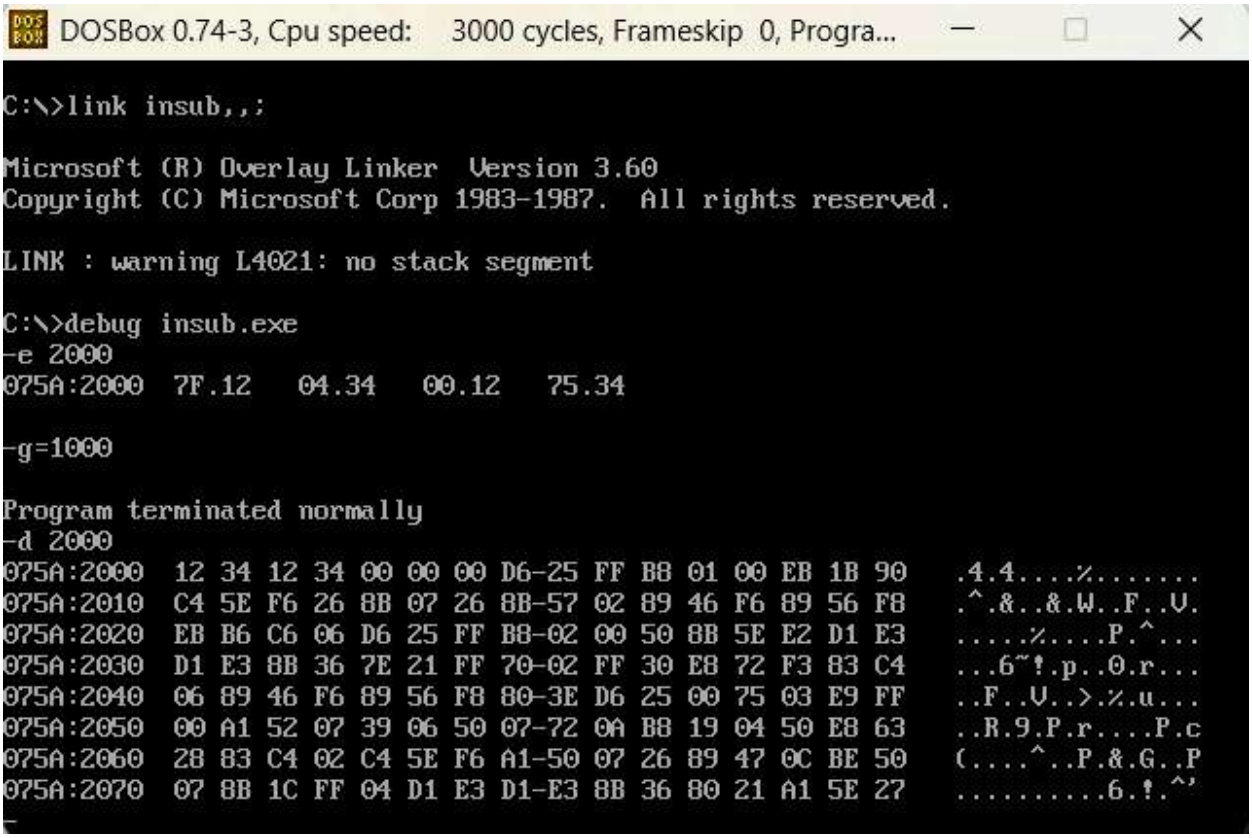
OUTPUT TABLE:

MEMORY LOCATION (INPUT)	MEMORY LOCATION (OUTPUT)
1200	12
1201	34
1202	12
1203	34
1204	00
1205	00

MANUAL CALCULATION :



OUTPUT SCREEN FROM MASM SOFTWARE :



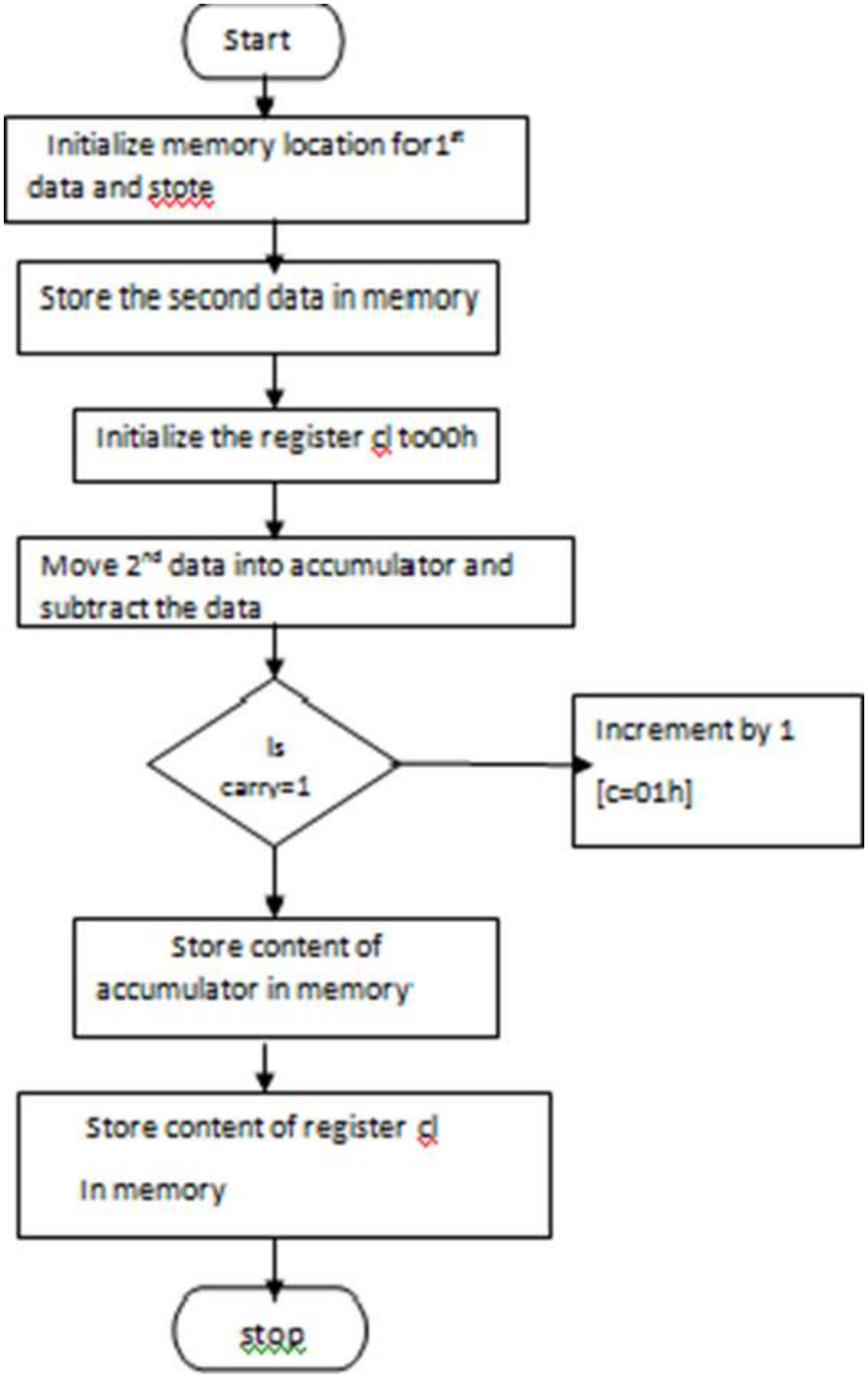
3. MULTIPLICATION

Algorithm

1. Initialize memory and store operands.
2. Move operands to registers.

3. Multiply.
4. Store result.

##FLOWCHART



Program

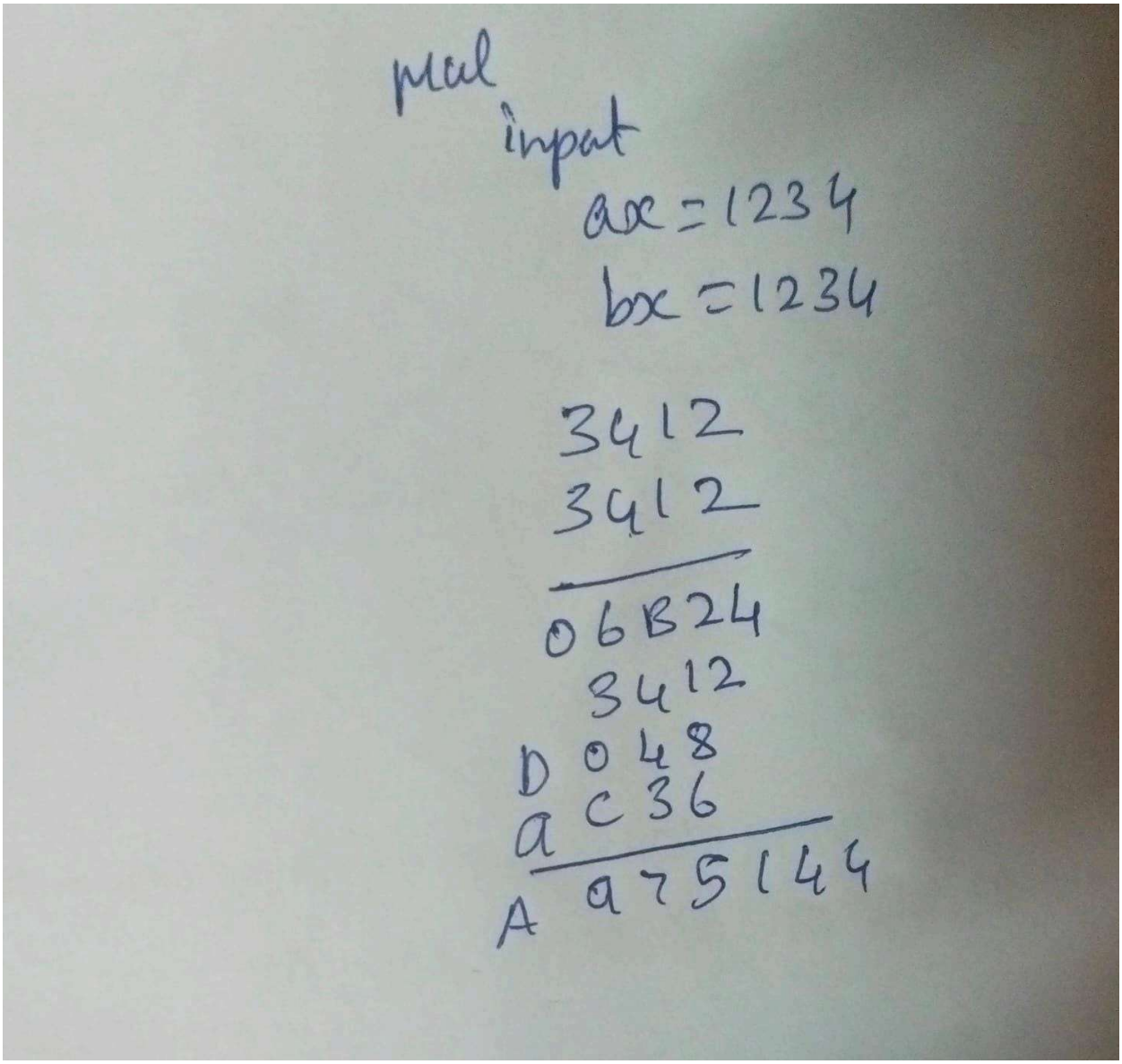
```
CODE SEGMENT
ASSUME CS: CODE, DS: CODE
ORG 1000H
MOV SI,2000H
MOV DX,0000H
MOV AX,[SI]
MOV BX,[SI+02H]
MUL BX
MOV [SI+04H],AX
MOV [SI+06H],DX
MOV AH,4CH
INT 21H
CODE ENDS
END
```

Output Table

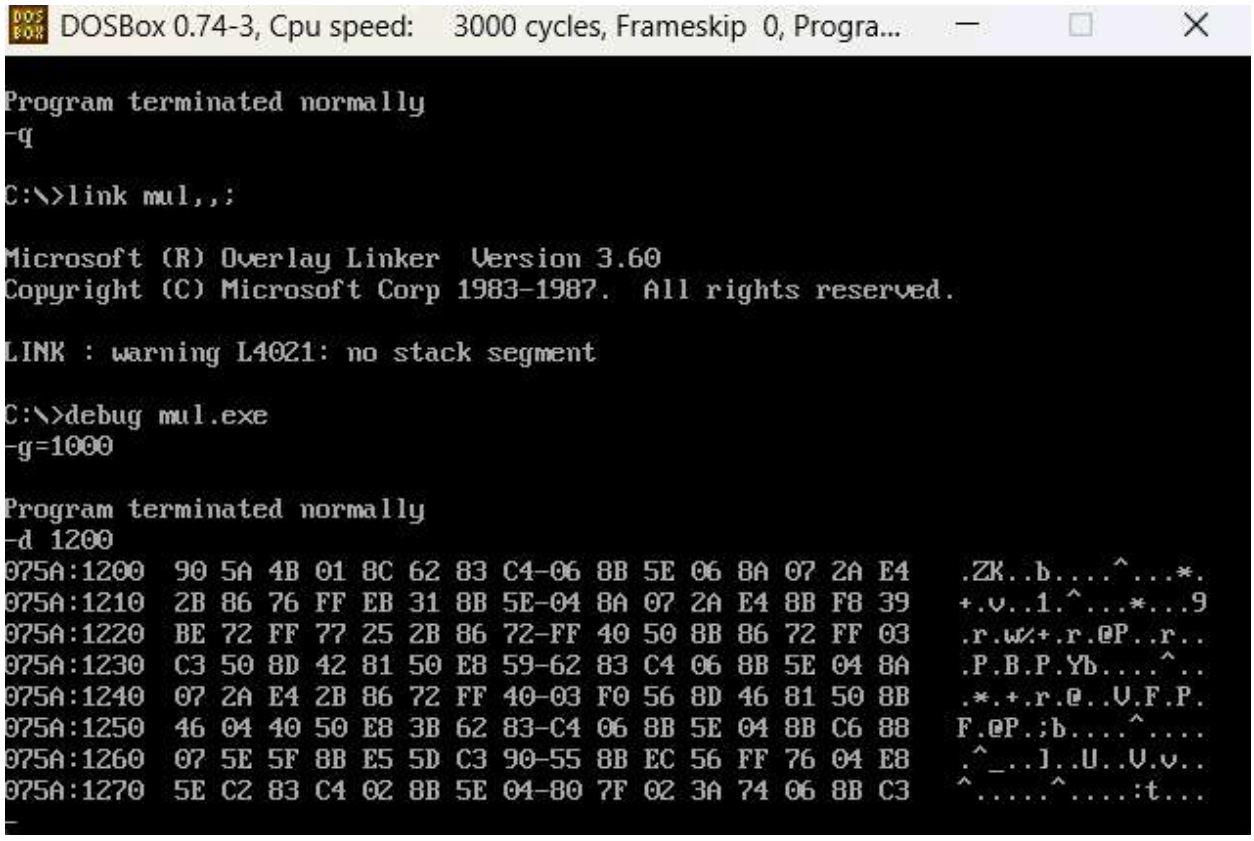
MEMORY LOCATION (INPUT)	MEMORY LOCATION (OUTPUT)
1200	12
1201	34
1202	12
1203	34
1204	09
1205	5A

Manual Calculations :

(Add your calculation here)


mul
input
ax = 1234
bx = 1234
3412
3412
06B24
3412
D048
AC36
A975144

OUTPUT SCREEN FROM MASM SOFTWARE :



INDIRECT MULTIPLICATION :

PROGRAM:

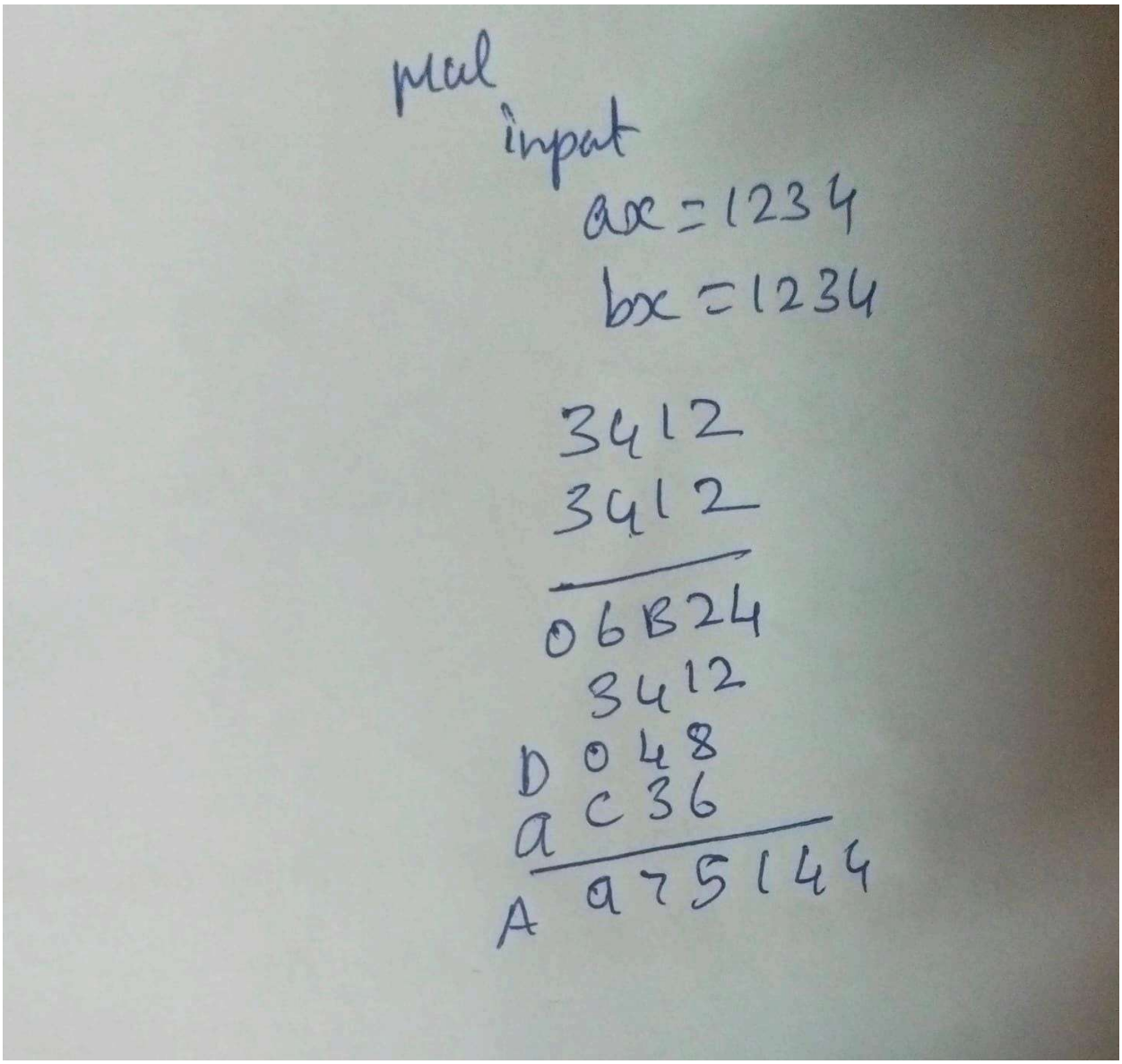
```
ASSUME CS: CODE,DS:CODE
ORG 1000H
MOV SI,2000H
MOV DX,0000H
 MOV AX,[SI]
MOV BX,[SI+02H]
MUL BX
MOV [SI+04H],AX
MOV [SI+06H],DX
MOV AH,4CH
INT 21H
 CODE ENDS
END
```

OUTPUT TABLE :

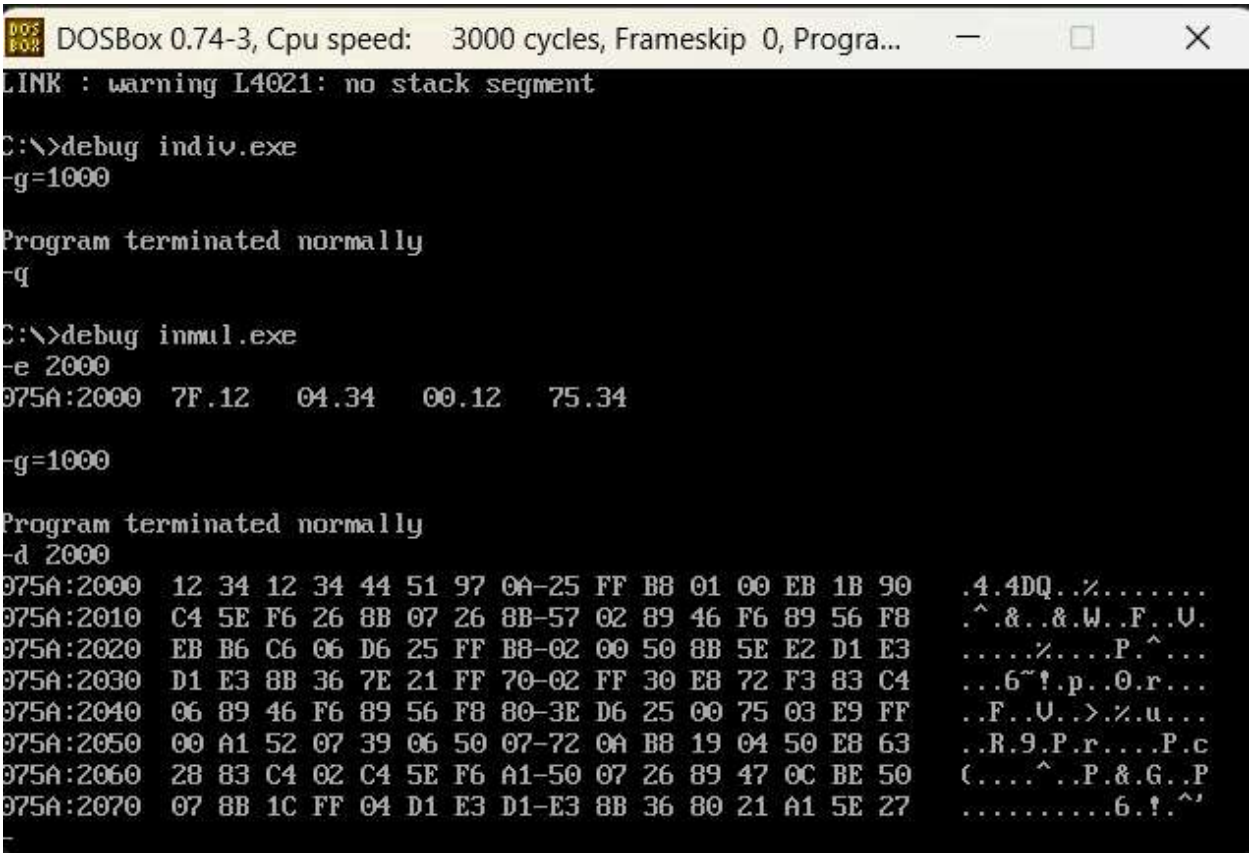
MEMORY LOCATION (INPUT)	MEMORY LOCATION (OUTPUT)
2000	12

MEMORY LOCATION (INPUT)	MEMORY LOCATION (OUTPUT)
2001	
2002	12
2003	34
2004	44
2005	51
2006	97
2007	0A

Manual Calculations :



OUTPUT SCREEN FROM MASM SOFTWARE :

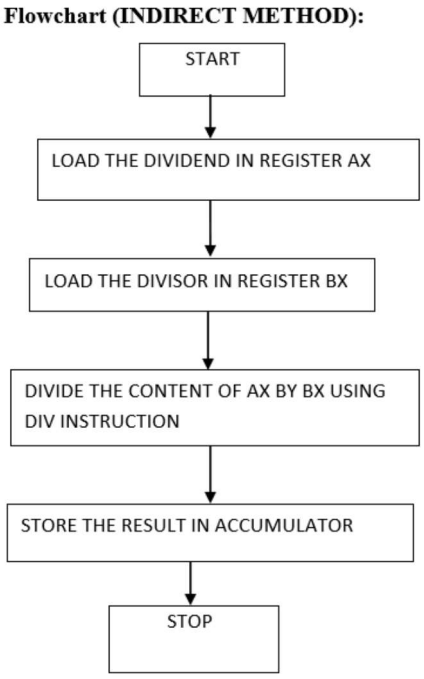


4. DIVISION

Algorithm

1. Load memory location of operands.
2. Perform division.
3. Store result.

FLOWCHART



Program

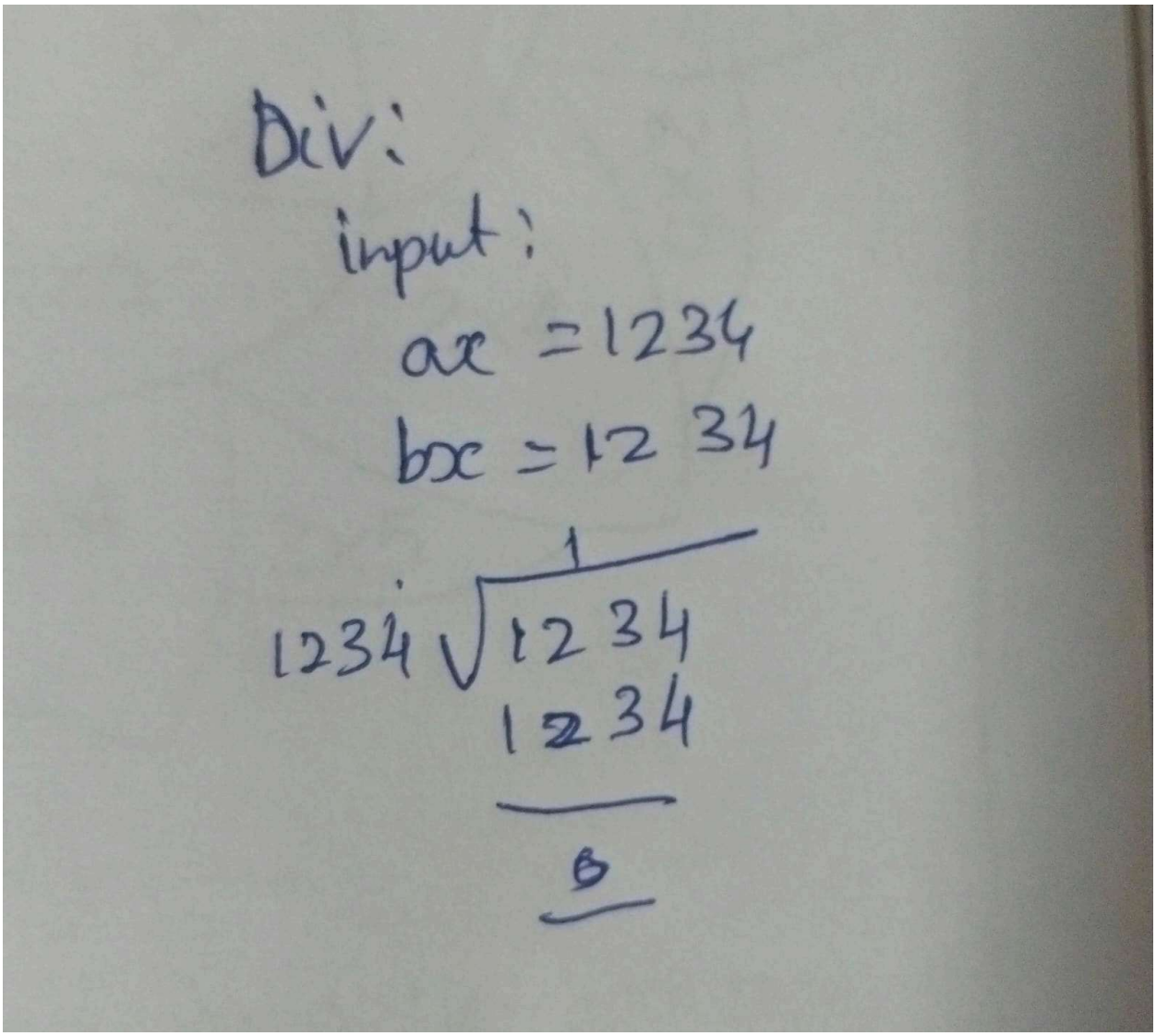
```
CODE SEGMENT
ASSUME CS: CODE, DS: CODE
ORG 1000H
MOV SI,2000H
MOV DX,0000H
MOV AX,[SI]
MOV BX,[SI+02H]
DIV BX
MOV [SI+04H],AX
MOV [SI+06H],DX
MOV AH,4CH
INT 21H
CODE ENDS
END
```

Output Table

MEMORY LOCATION (INPUT)	MEMORY LOCATION (OUTPUT)
2000	34

MEMORY LOCATION (INPUT)	MEMORY LOCATION (OUTPUT)
2001	
2002	34
2003	12
2004	01
2005	00

Manual Calculations :

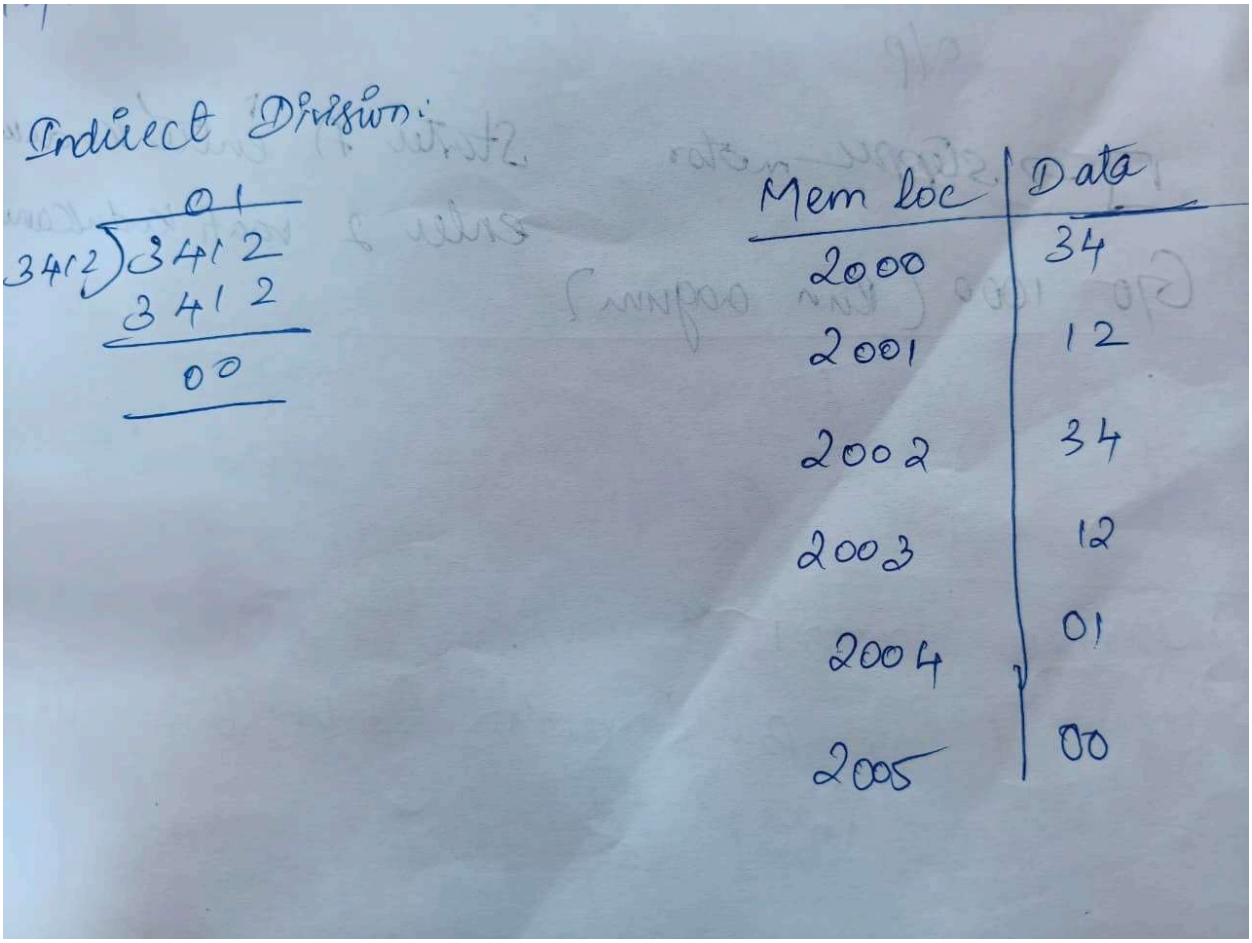


(Add your calculation here)

OUTPUT FROM MASM SOFTWARE :

```
DOSBox 0.74-3, Cpu speed:   3000 cycles, Frameskip  0, Progra...

     0 Warning Errors
     0 Severe  Errors

C:\>link div,,;

Microsoft (R) Overlay Linker  Version 3.60
Copyright (C) Microsoft Corp 1983-1987.  All rights reserved.

LINK : warning L4021: no stack segment

C:\>debug div.exe
-g=1000

Program terminated normally
-d 1200
075A:1200  01 00 00 00 8C 62 83 C4-06 8B 5E 06 8A 07 2A E4   .....b....^...*.
075A:1210  2B 86 76 FF EB 31 8B 5E-04 8A 07 2A E4 8B F8 39   +.v..1.^...*...9
075A:1220  BE 72 FF 77 25 2B 86 72-FF 40 50 8B 86 72 FF 03   .r.w%+.r.@P..r..
075A:1230  C3 50 8D 42 81 50 E8 59-62 83 C4 06 8B 5E 04 8A   .P.B.P.Yb....^..
075A:1240  07 2A E4 2B 86 72 FF 40-03 F0 56 8D 46 81 50 8B   .*.+.r.@..V.F.P.
075A:1250  46 04 40 50 E8 3B 62 83-C4 06 8B 5E 04 8B C6 88   F.@P.;b....^....
075A:1260  07 5E 5F 8B E5 5D C3 90-55 8B EC 56 FF 76 04 E8   .^_..]..U..V.v..
075A:1270  5E C2 83 C4 02 8B 5E 04-80 7F 02 3A 74 06 8B C3   ^.....^....:t...
-
```

INDIRECT DIVISION :



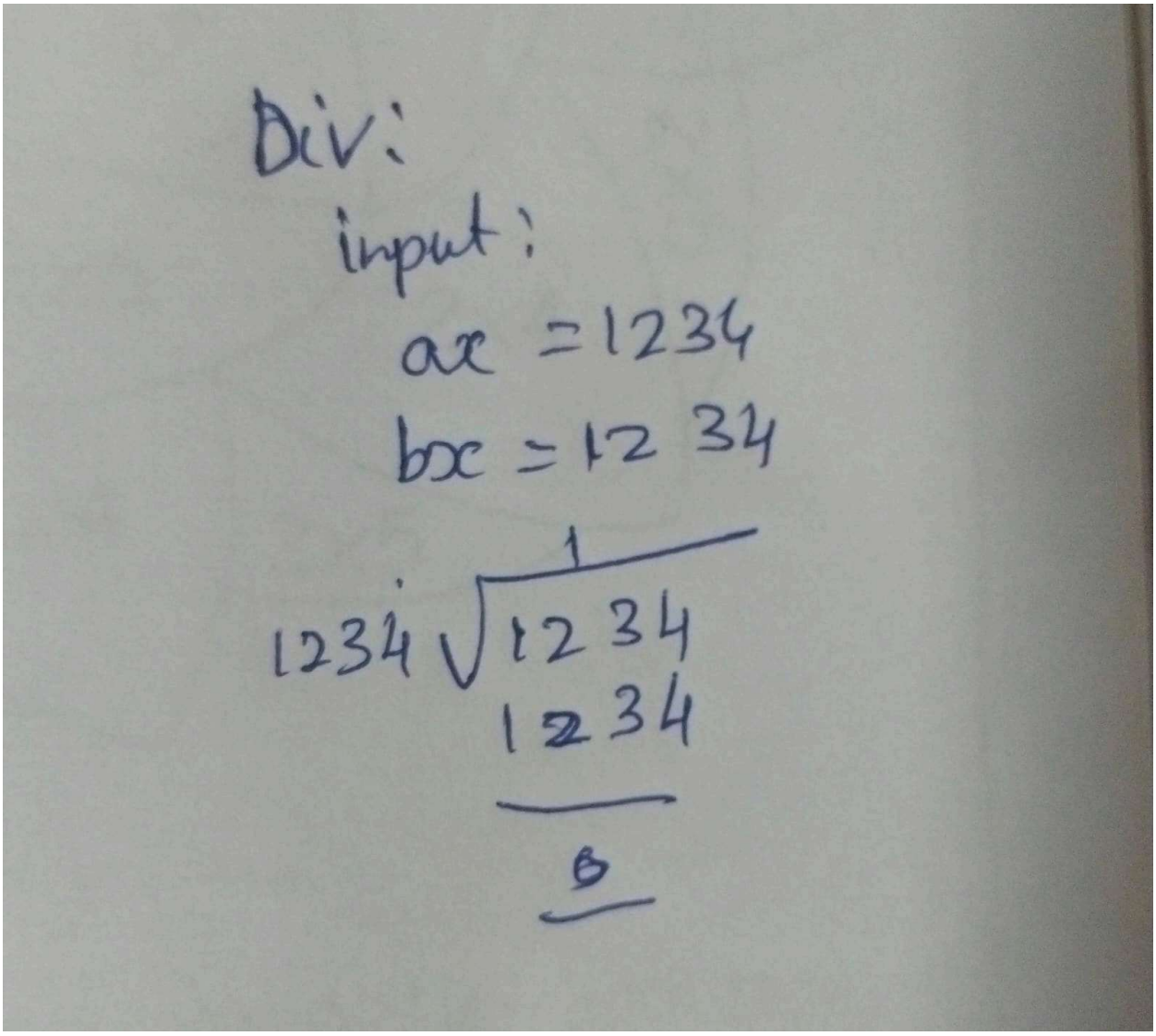
PROGRAM :

```
CODE SEGMENT
ASSUME CS: CODE,DS:CODE
ORG 1000H
MOV SI,2000H
MOV DX,0000H
MOV AX,[SI]
MOV BX,[SI+02H]
DIV BX
MOV [SI+04H],AX
MOV [SI+06H],DX
MOV AH,4CH
INT 21H
CODE ENDS
END
```

OUTPUT TABLE :

MEMORY LOCATION (INPUT)	MEMORY LOCATION (OUTPUT)
2000	34
2001	
2002	34
2003	12
2004	01
2005	00

MANUAL CALCULATIONS :



OUTPUT FROM MASM SOFTWARE :

```
DOSBox 0.74-3, Cpu speed:   3000 cycles, Frameskip  0, Progra...
-e 2000
075A:2000  43.

-q

C:\>e 2000
Illegal command: e.

C:\>debug indiv.exe
-e 2000
075A:2000  43.34   04.12   00.34   75.12

-g=1000

Program terminated normally
-d 2000
075A:2000  34 12 34 12 01 00 00 00-25 FF B8 01 00 EB 1B 90   4.4.....%.......
075A:2010  C4 5E F6 26 8B 07 26 8B-57 02 89 46 F6 89 56 F8   .^.&..&.W..F..V.
075A:2020  EB B6 C6 06 D6 25 FF B8-02 00 50 8B 5E E2 D1 E3   .....%....P.^...
075A:2030  D1 E3 8B 36 7E 21 FF 70-02 FF 30 E8 72 F3 83 C4   ...6~!.p..0.r...
075A:2040  06 89 46 F6 89 56 F8 80-3E D6 25 00 75 03 E9 FF   ..F..V..>.%.u...
075A:2050  00 A1 52 07 39 06 50 07-72 0A B8 19 04 50 E8 63   ..R.9.P.r....P.c
075A:2060  28 83 C4 02 C4 5E F6 A1-50 07 26 89 47 0C BE 50   (....^..P.&.G..P
075A:2070  07 8B 1C FF 04 D1 E3 D1-E3 8B 36 80 21 A1 5E 27   ..........6.!.^'
-
```

RESULT

Thus, the Assembly Language Programs for 8086 to perform arithmetic operations (Addition, Subtraction, Multiplication, and Division) using both direct and indirect methods were successfully written and executed using MASM.