

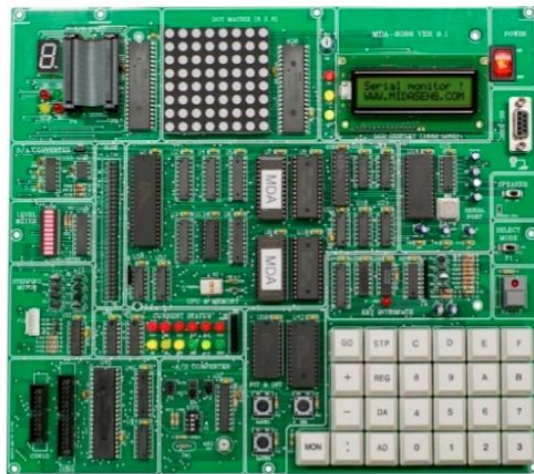
Experiment No: 01

Experiment Name: Familiarization of MDA-8086

Objective: 1) To understand the component of MDA-8086 trainer board
2) To know about 8086 Microprocessor

Equipment: MDA-8086 Kit

Figure:



Result:

Familiarizing with the MDA-8086 enhances understanding of 8086 architecture, assembly programming, hardware interfacing, and microprocessor system design, building essential skills for embedded systems and computer architecture.

Discussion:

Familiarization with the MDA-8086 provides hands-on experience with 8086 microprocessor architecture, programming, and interfacing, essential for learning embedded systems and hardware design.

Experiment No: 02

Experiment Name:Add, Substraction, Multiplication, Division using 8086 Compiler

Objective:1) To learn how to use arithmetic instructions in assembly language
2) To understand the register operation of 8086 for data manipulation
3) To gain hands-on experience in programming and debugging with the 8086 microprocessor

Equipment: 8086 Compiler(Online Compiler)/Emu8086

Code:

start:

; add,sub,mul,div

mov ax,7

mov bx,2

add ax,bx

mov ax,5

mov bx,2

sub ax,bx

mov ax,4

mov bx,2

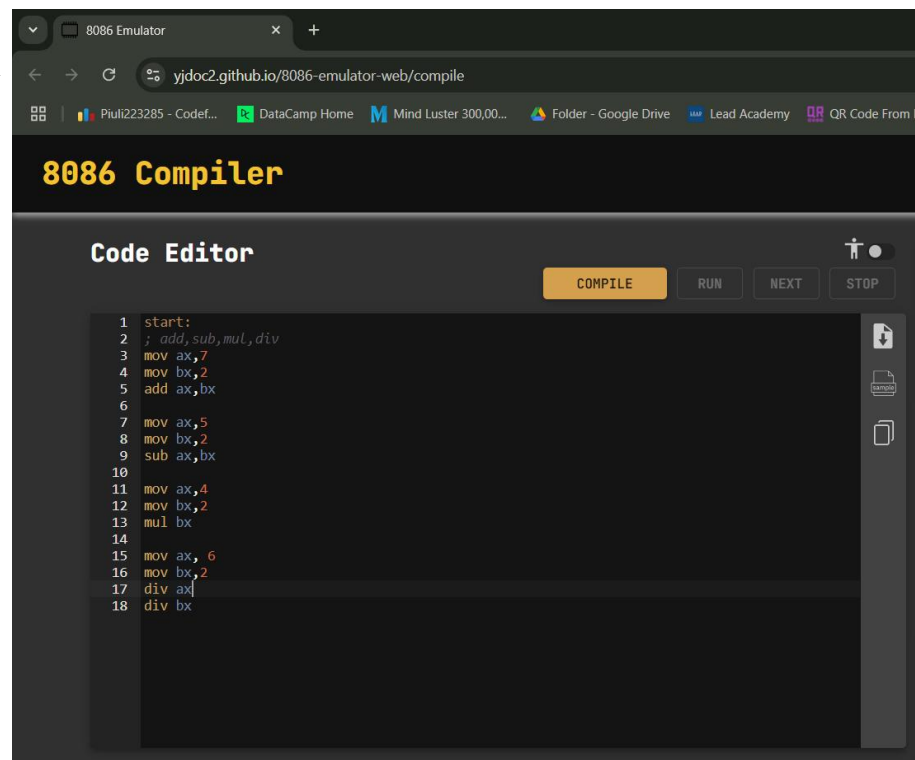
mul bx

mov ax, 6

mov bx,2

div ax

div bx



Result:

Add:

Reg	H	L
A	00	09
B	00	02
C	00	00
D	00	00

Sub:

Reg	H	L
A	00	03
B	00	02
C	00	00
D	00	00

Mul:

Reg	H	L
A	00	08
B	00	02
C	00	00
D	00	00

Div:

Reg	H	L
A	00	03
B	00	02
C	00	00
D	00	00

Discussion:

Experiment No: 03

Experiment Name:JMP and LOOP using 8086 Compiler

Objective:1) To learn how to use arithmetic instructions in assembly language
2)To understand the register operation of 8086 for data manipulation
3)To gain hands-on experience in programming and debugging with the 8086 microprocessor

Equipment: 8086 Compiler(Online Compiler)/Emu8086

Code:(i)JMP

start:

;Unconditional Jump

MOV AX,5

MOV BX,2

JMP CALC

BACK:JMP STOP

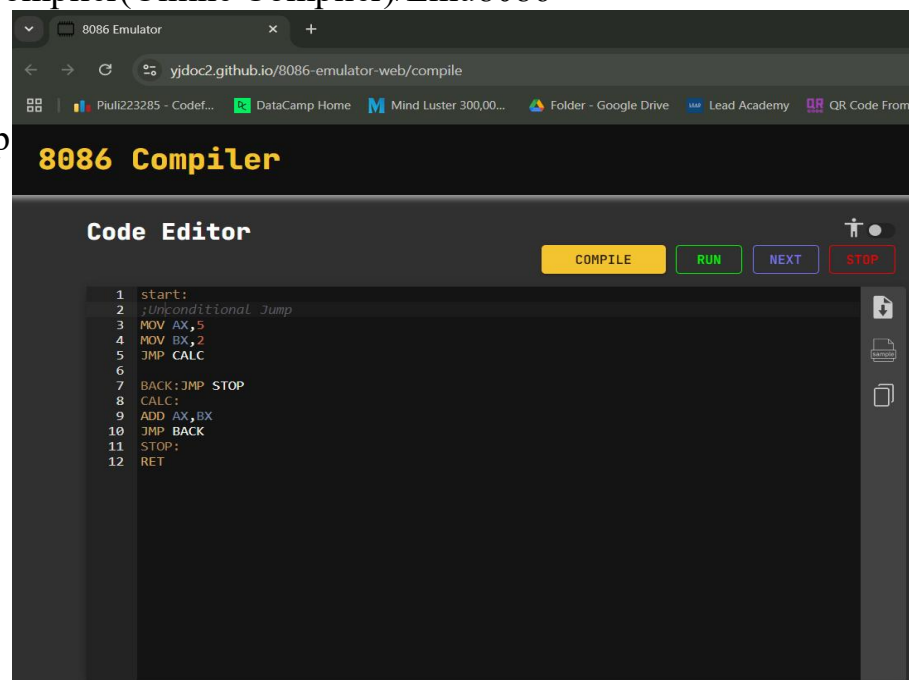
CALC:

ADD AX,BX

JMP BACK

STOP:

RET



(ii)LOOP

org 100h

; add your code here

MOV AX,05H

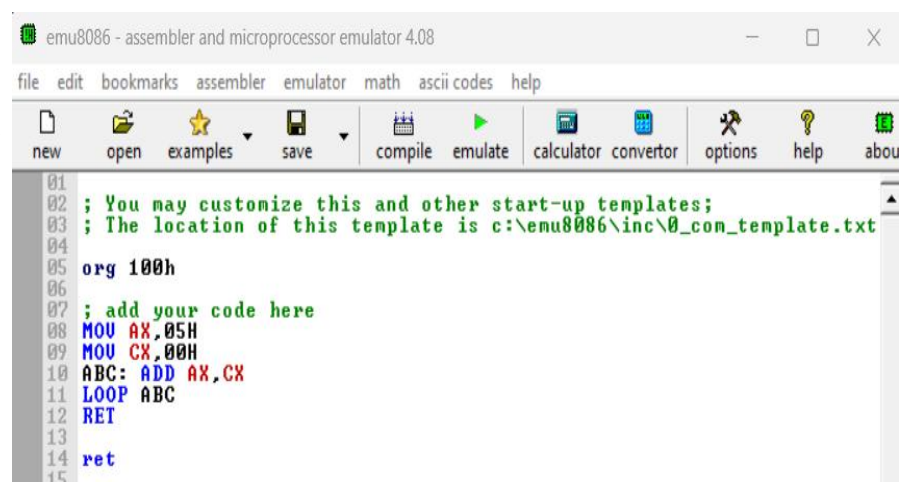
MOV CX,00H

ABC: ADD AX,CX

LOOP ABC

RET

ret

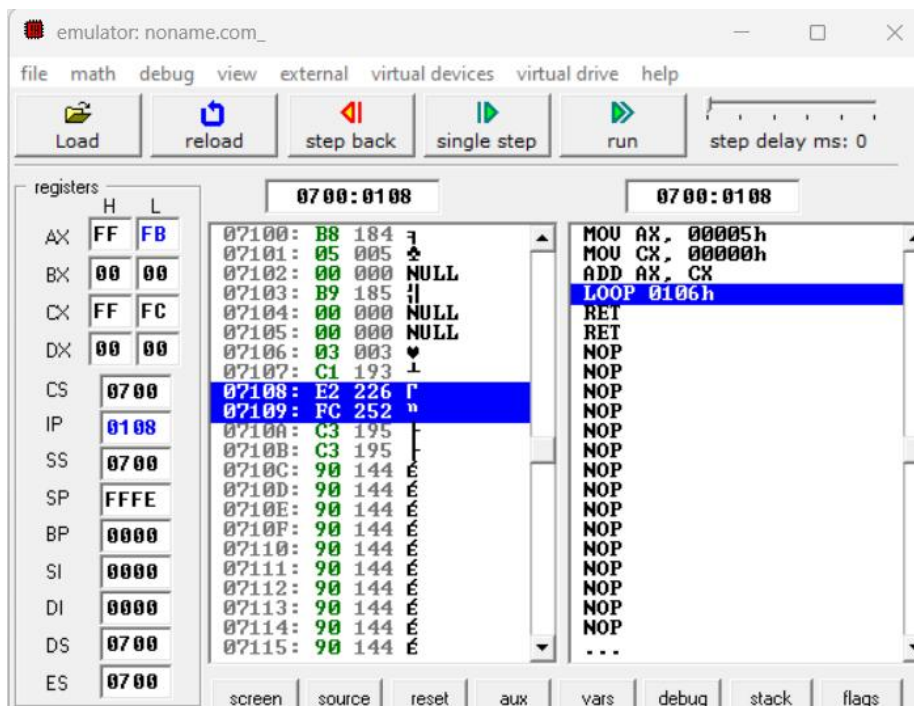


Result:

(i)JMP:

Reg	H	L
A	00	07
B	00	02
C	00	00
D	00	00

(ii)LOOP:



Discussion:

Experiment No: 04

Experiment Name:Shift Left, Shift Right, Rotate Left, Rotate Right using 8086 Compiler

Objective:1) To learn how to use arithmetic instructions in assembly language
2)To understand the register operation of 8086 for data manipulation
3)To gain hands-on experience in programming and debugging with the 8086 microprocessor

Equipment: 8086 Compiler(Online Compiler)/Emu8086

Code:

start:

;shift left and shift right

mov ax,8

shl ax,1

mov ax,8

shr ax,1

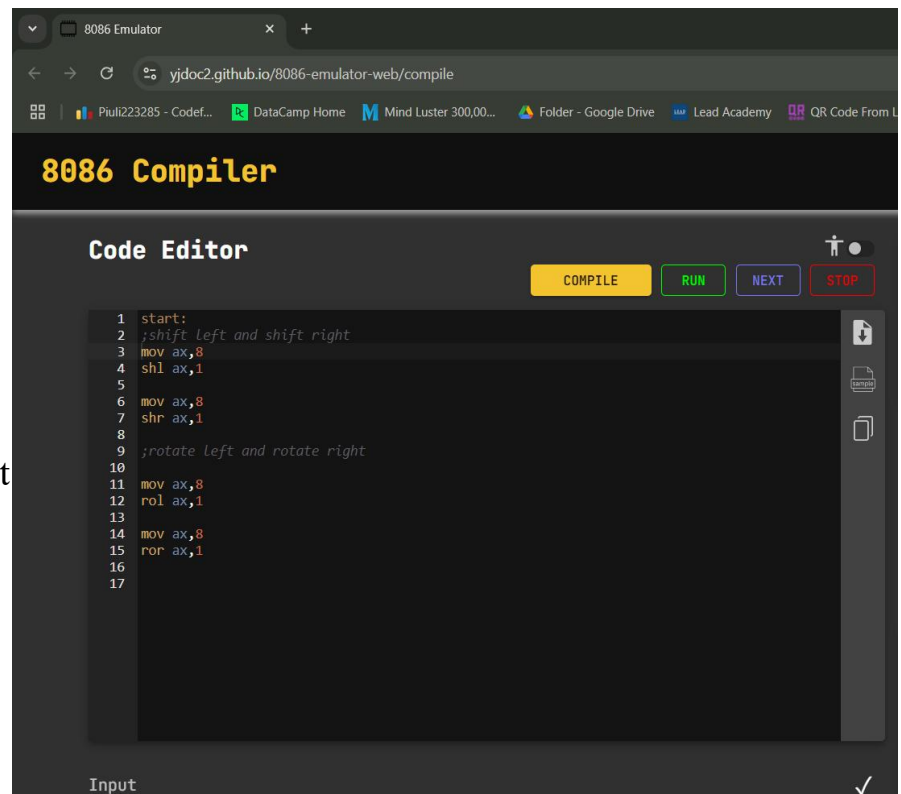
;rotate left and rotate right

mov ax,8

rol ax,1

mov ax,8

ror ax,1

The screenshot shows a web browser window titled "8086 Emulator" with the URL "ydoc2.github.io/8086-emulator-web/compile". The browser's address bar and tabs are visible. Below the browser, there is a dark-themed interface for the "8086 Compiler". The main area is a "Code Editor" with a text area containing assembly code. The code is as follows:

```
1 start:
2 ;shift left and shift right
3 mov ax,8
4 shl ax,1
5
6 mov ax,8
7 shr ax,1
8
9 ;rotate left and rotate right
10
11 mov ax,8
12 rol ax,1
13
14 mov ax,8
15 ror ax,1
16
17
```

At the top right of the code editor, there are four buttons: "COMPILE" (yellow), "RUN" (green), "NEXT" (blue), and "STOP" (red). On the right side of the code editor, there are icons for file operations: a download icon, a file icon, and a copy icon. At the bottom left of the code editor, there is an "Input" label. At the bottom right, there is a checkmark icon.

Result:

Shift Left:

Reg	H	L
A	00	10
B	00	00
C	00	00
D	00	00

Shift Right:

Reg	H	L
A	00	04
B	00	00
C	00	00
D	00	00

Rotate Left:

Reg	H	L
A	00	10
B	00	00
C	00	00
D	00	00

Rotate Right:

Reg	H	L
A	00	04
B	00	00
C	00	00
D	00	00

Discussion:

Experiment No: 05

Experiment Name: Simple Program Run using 8086 Compiler

Objective: 1) To learn how to use arithmetic instructions in assembly language
2) To understand the register operation of 8086 for data manipulation
3) To gain hands-on experience in programming and debugging with the 8086 microprocessor

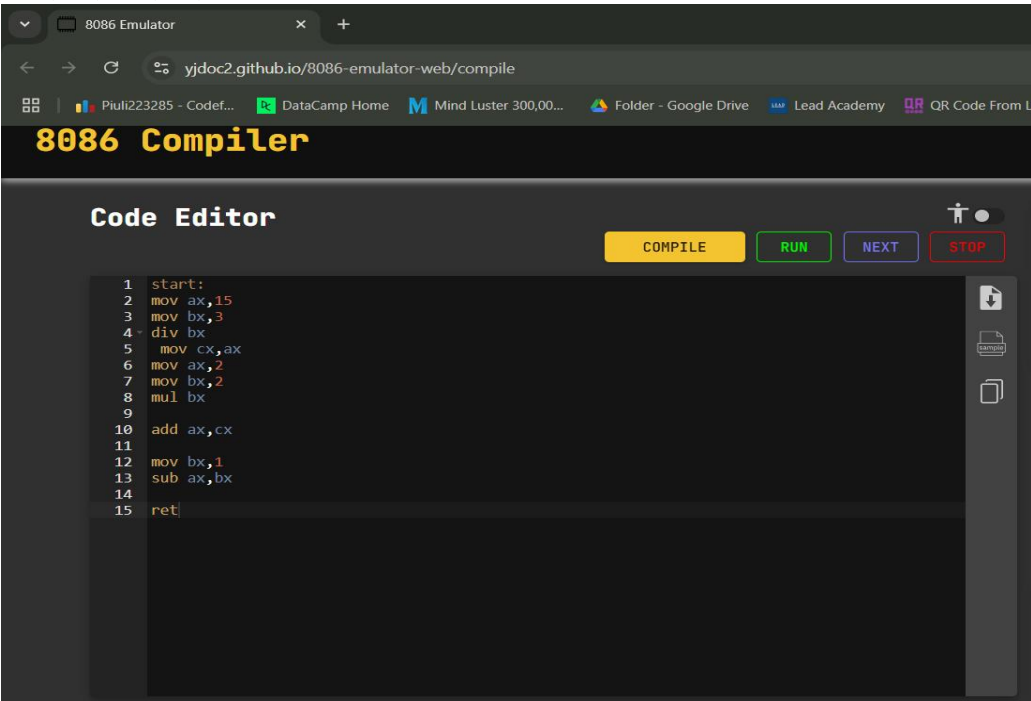
Equipment: 8086 Compiler(Online Compiler)/Emu8086

Code: (i) $((((15/3)+(2*2)-1))$

start:

```
mov ax,15
mov bx,3
div bx
mov cx,ax
mov ax,2
mov bx,2
mul bx
add ax,cx

mov bx,1
sub ax,bx
ret
```

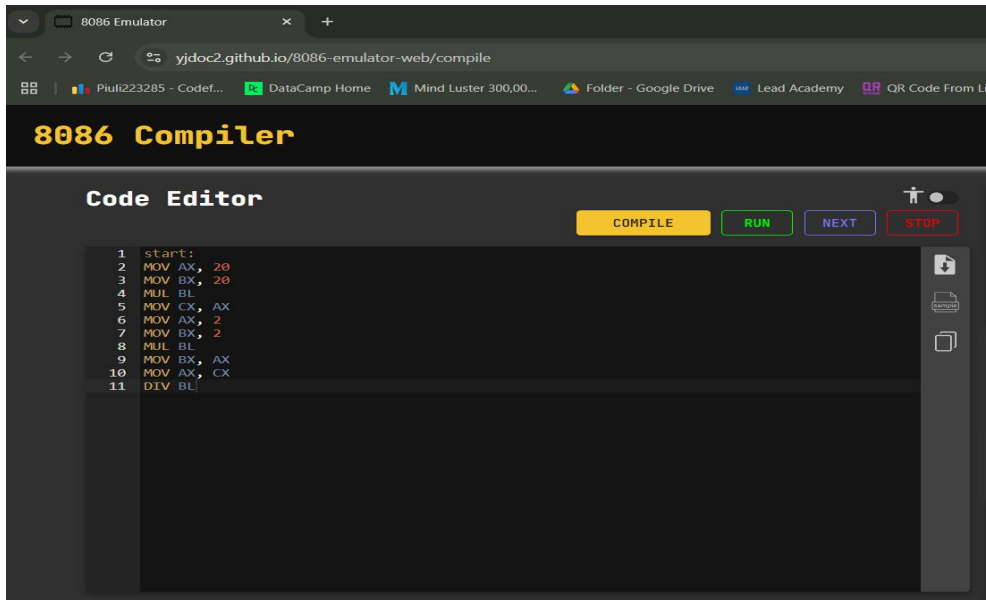
A screenshot of a web browser displaying the '8086 Compiler' interface. The browser's address bar shows 'yjdcc2.github.io/8086-emulator-web/compile'. The page has a dark theme with a yellow title '8086 Compiler'. Below the title is a 'Code Editor' section with a text area containing assembly code. To the right of the code editor are four buttons: 'COMPILE' (yellow), 'RUN' (green), 'NEXT' (blue), and 'STOP' (red). The code in the editor is as follows:

```
1 start:
2 mov ax,15
3 mov bx,3
4 div bx
5 mov cx,ax
6 mov ax,2
7 mov bx,2
8 mul bx
9
10 add ax,cx
11
12 mov bx,1
13 sub ax,bx
14
15 ret
```

(ii) Floor size 20*20, Tiles size 2*2. How many tiles are needed to cover up the floor?

->start:

```
MOV AX, 20
MOV BX, 20
MUL BL
MOV CX, AX
MOV AX, 2
MOV BX, 2
MUL BL
MOV BX, AX
MOV AX, CX
DIV BL
```

A screenshot of the same '8086 Compiler' web interface. The code editor now contains assembly code for part (ii) of the experiment:

```
1 start:
2 MOV AX, 20
3 MOV BX, 20
4 MUL BL
5 MOV CX, AX
6 MOV AX, 2
7 MOV BX, 2
8 MUL BL
9 MOV BX, AX
10 MOV AX, CX
11 DIV BL
```


Result:

(i)

The screenshot shows the 8086 Compiler interface. The Code Editor contains the following assembly code:

```
1 start:
2 mov ax,15
3 mov bx,3
4 div bx
5 mov cx,ax
6 mov ax,2
7 mov bx,2
8 mul bx
9
10 add ax,cx
11
12 mov bx,1
13 sub ax,bx
14
15 ret
```

The right panel displays the Register (Reg) values:

Reg	H	L
A	00	08
B	00	01
C	00	05
D	00	00

The Flags section shows:

OF	DF	IF
0	0	0

(ii)

The screenshot shows the 8086 Compiler interface after execution. The Code Editor contains the following assembly code:

```
1 start:
2 MOV AX, 20
3 MOV BX, 20
4 MUL BL
5 MOV CX, AX
6 MOV AX, 2
7 MOV BX, 2
8 MUL BL
9 MOV BX, AX
10 MOV AX, CX
11 DIV BL
```

The right panel displays the Register (Reg) values:

Reg	H	L
A	00	64
B	00	04
C	01	90
D	00	00

The Flags section shows:

OF	DF	IF
0	0	0

Discussion:

Experiment No: 06

Experiment Name: Temperature, Area, Factorial using 8086 Compiler

Objective: 1) To learn how to use arithmetic instructions in assembly language
2) To understand the register operation of 8086 for data manipulation
3) To gain hands-on experience in programming and debugging with the 8086 microprocessor

Equipment: 8086 Compiler(Online Compiler)/Emu8086

Code:

Factorial: $(5! / 3!) + 4!$

start:

;Factorial

MOV AX, 1

MOV CL, 5

L1:MUL CL

LOOP L1

MOV DX, AX

MOV AX, 1

MOV CL, 3

L2:MUL CL

LOOP L2

MOV BX, AX

MOV AX, DX

DIV BL

MOV DX, AX

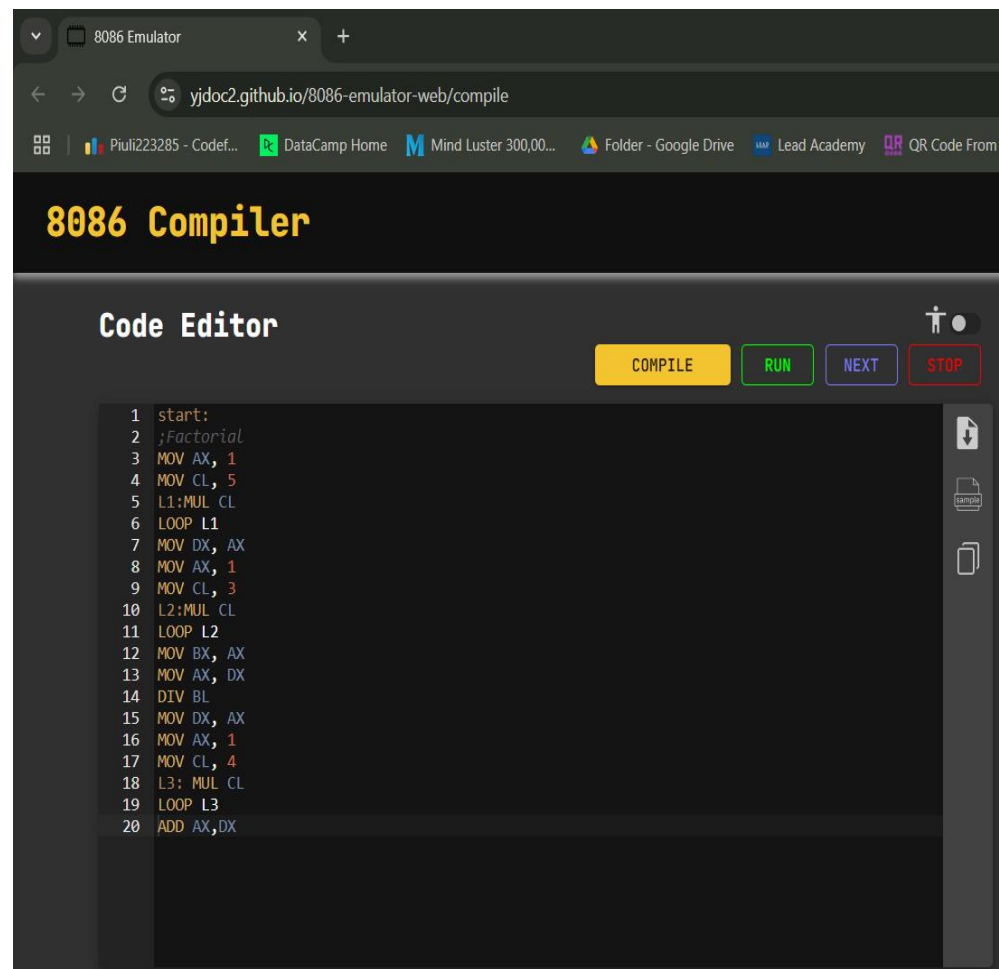
MOV AX, 1

MOV CL, 4

L3: MUL CL

LOOP L3

ADD AX,DX



Area: Trapezium

start:

;Area of trapezium

MOV AX,1

MOV BX,3

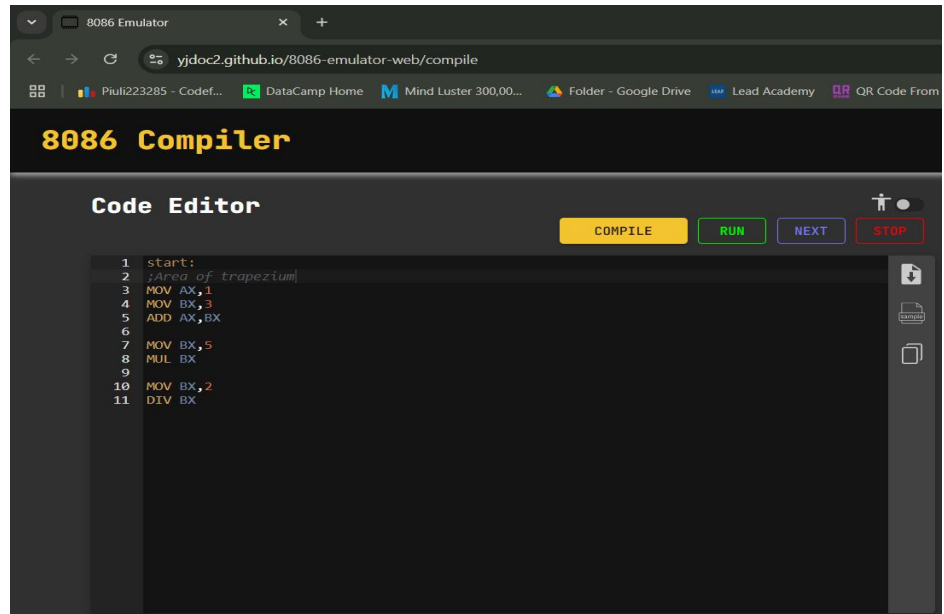
ADD AX,BX

MOV BX,5

MUL BX

MOV BX,2

DIV BX



Temperature:

Result:

Factorial:

Reg	H	L
A	00	2c
B	00	06
C	00	00
D	00	14

Area:

Trapezium

Reg	H	L
A	00	0a
B	00	02
C	00	00
D	00	00

Temperature: