

ASSIGNMENT-1

S.PRAVEEN KUMAR

CH.EN.U4AIE22048

I) $C = A + B$

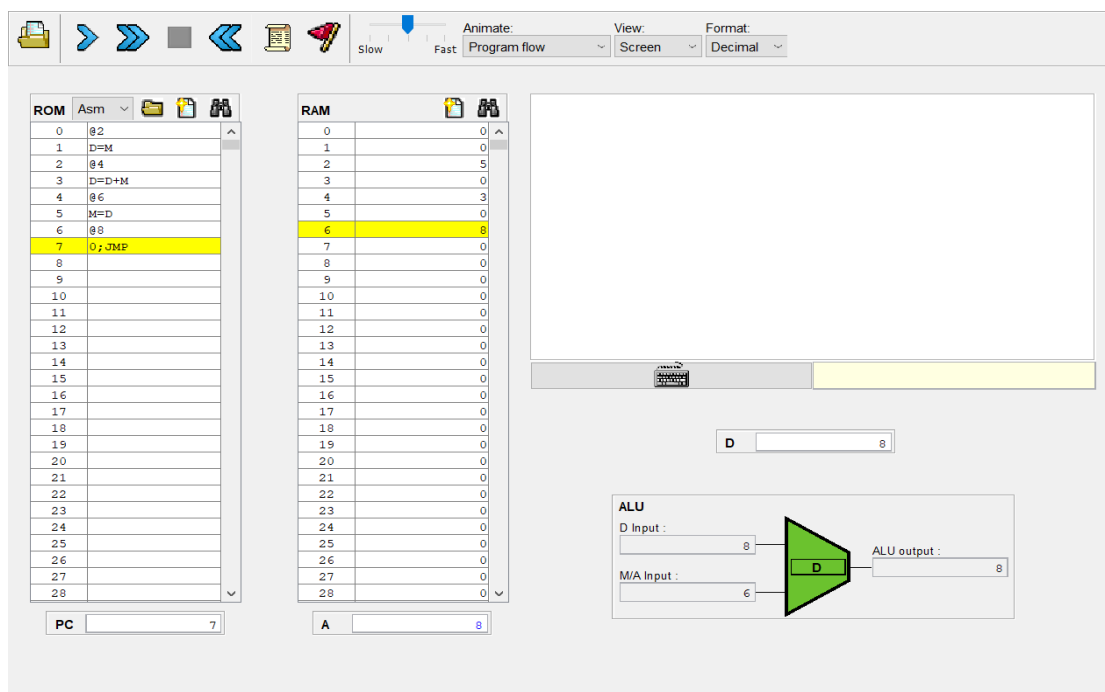
AIM:

To execute $C = A + B$ in CPU emulator using Assembly Language in Nand2tetris.

HACK ASSEMBLY CODE:

- ◆ 0 @2
- ◆ 1 D=M
- ◆ 2 @4
- ◆ 3 D=D+M
- ◆ 4 @6
- ◆ 5 M=D
- ◆ 6 @8
- ◆ 7 0;JMP

VERIFICATION SCREENSHOT:



RESULT:

The Addition of two numbers is executed successfully using CPU emulator in Nand2tetris using Assembly Language.

II)D=A+B-C

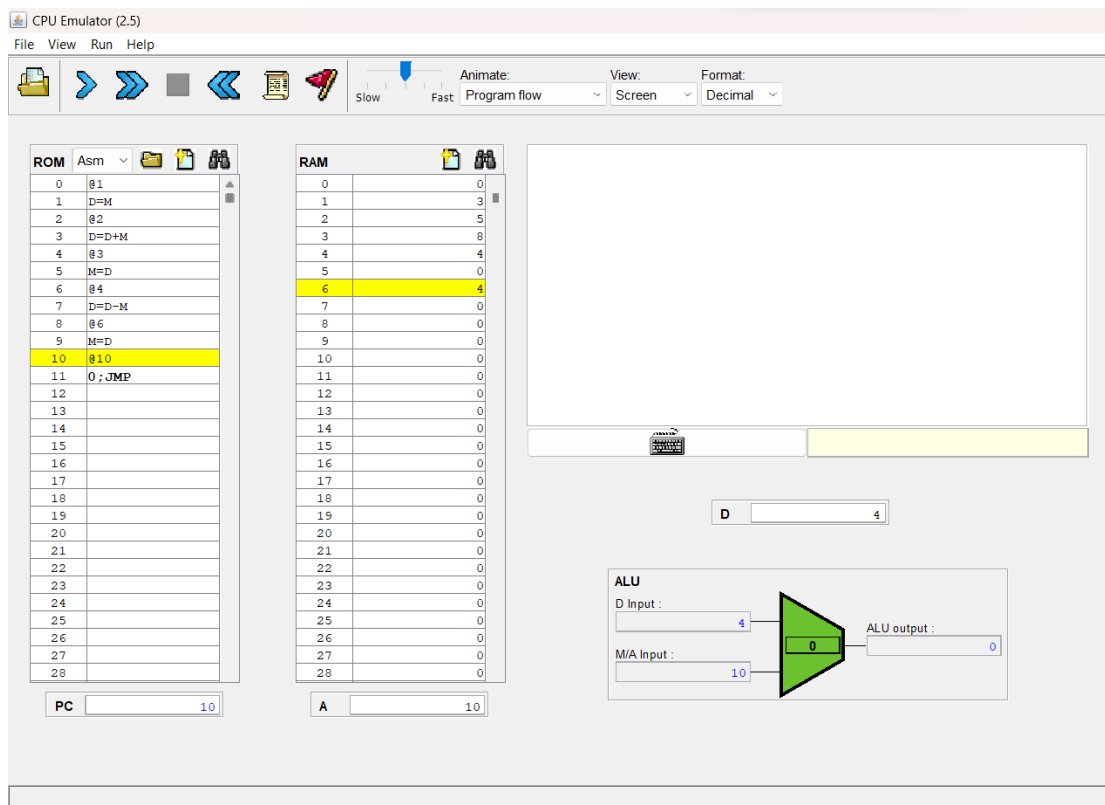
AIM:

To execute $D=A+B-C$ in CPU emulator using Assembly Language in Nand2tetris.

HACK ASSEMBLY CODE:

```
0 @1
1 D=M
2 @2
3 D=D+M
4 @3
5 M=D
6 @4
7 D=D-M
8 @6
9 M=D
10 @10
11 0;JMP
```

VERIFICATION SCREENSHOT:



RESULT:

Thus, the expression $A+B-C$ is executed successfully using CPU emulator in Nand2tetris using Assembly Language.

III) $E = (A + B) - (C + D)$

AIM:

To execute $E = (A + B) - (C + D)$ in CPU emulator using Assembly Language in Nand2tetris.

HACK ASSEMBLY CODE:

```
@0
D=M
@10
M=D
@1
D=M
@10
M=D+M
@2
D=M
@12
M=D
@3
D=M
@12
M=D+M
@10
D=M
@15
M=D
@12
D=M
@15
M=M-D
D=M
@25
D:JMP
```

VERIFICATION SCREENSHOT:

The screenshot displays a CPU emulator interface with the following components:

- ROM Table:**

Address	Instruction
0	@0
1	D=M
2	@10
3	M=D
4	@1
5	D=M
6	@10
7	M=D+M
8	@2
9	D=M
10	@12
11	M=D
12	@3
13	D=M
14	@12
15	M=D+M
16	@10
17	D=M
18	@15
19	M=D
20	@12
21	D=M
22	@15
23	M=M-D
24	D=M
25	@25
26	0;JMP
27	
28	

- RAM Table:**

Address	Value
0	5
1	7
2	1
3	2
4	0
5	0
6	0
7	0
8	0
9	0
10	12
11	0
12	3
13	0
14	0
15	9
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0

- PC (Program Counter):** 26
- A (Accumulator):** 25
- D Register:** 9
- ALU (Arithmetic Logic Unit):** Shows D Input: 9, M/A Input: 25, and ALU output: 0.

RESULT:

Thus, the expression $(A+B)-(C+D)$ is executed successfully using CPU emulator in Nand2tetris using Assembly Language.

EXP:02 SWAPPING TWO NUMBER

AIM:

To Swap two numbers in CPU emulator using Assembly Language in Nand2tetris.

HACK ASSEMBLY CODE:

- @1
- D=M
- @3
- M=D
- @2
- D=M
- @1
- M=D
- @3
- D=M
- @2
- M=D
- @12
- 0;JMP

VERIFICATION SCREENSHOT:

The screenshot displays a CPU emulator interface with the following components:

- ROM Table:**

Address	Instruction
0	@1
1	D=M
2	@3
3	M=D
4	@2
5	D=M
6	@1
7	M=D
8	@3
9	D=M
10	@2
11	M=D
12	@12
13	0; JMP
14	
15	
16	
17	
18	
19	
20	
21	
22	
23	
24	
25	
26	
27	
28	

- RAM Table:**

Address	Value
0	0
1	84
2	48
3	48
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0

- PC (Program Counter):** 13
- A (Accumulator):** 12
- D (Data Register):** 48
- ALU (Arithmetic Logic Unit):** Shows D Input: 48, M/A Input: 12, and ALU output: 0.

RESULT:

Swapping of two numbers is executed successfully using CPU emulator in Nand2tetris using Assembly Language.

EXP NO:03 SUM OF NTH NUMBERS

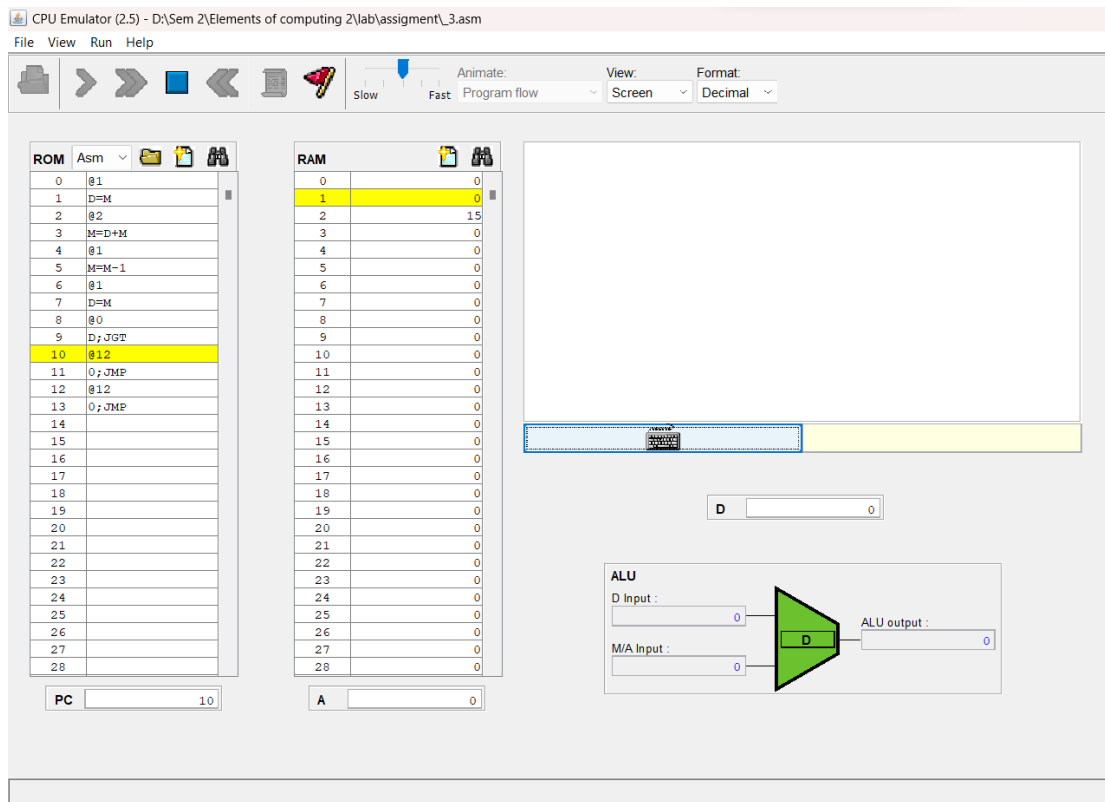
AIM:

To Sum of nth numbers in CPU emulator using Assembly Language in Nand2tetris.

HACK ASSEMBLY CODE:

```
0 @1
1 D=M
2 @2
3 M=D+M
4 @1
5 M=M-1
6 @1
7 D=M
8 @0
9 D;JGT
10 @12
11 0;JMP
12 @12
13 0;JMP
```

VERIFICATION SCREENSHOT:



RESULT:

Sum of "n"th numbers is executed successfully using CPU emulator in Nand2tetris using Assembly Language.

EXP NO: 04 IF A<0 PRINT 1 ELSE PRINT 0

AIM:

to print 1 if number>0 else print 0 USING a hack assembly language code

HACK ASSEMBLY CODE:

```
@0
```

```
D=M
```

```
@0
```

```
M=1
```

```
D;JGT
```

```
@0
```

```
M=0
```

```
D;JGT
```

```
@8
```

```
0;JMP
```

VERIFICATION SCREENSHOT:

IF PART

The screenshot shows a digital logic simulator interface. On the left, the ROM is populated with assembly code. The RAM is empty. The ALU output is 0. The PC register is 0. The A register is 0. The D register is 0. The M/A input is 0. The ALU output is 0.

ROM	Asm
0	@0
1	D=M
2	@0
3	M=1
4	D;JGT
5	@0
6	M=0
7	D;JGT
8	@8
9	0;JMP
10	
11	
12	
13	
14	
15	
16	
17	
18	
19	
20	
21	
22	
23	
24	
25	
26	
27	
28	

RAM	
0	5
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0

PC: 0, A: 0, D: 0, M/A Input: 0, ALU output: 0

The screenshot shows the same digital logic simulator interface after the first instruction. The RAM is updated. The ALU output is 5. The PC register is 0. The A register is 0. The D register is 5. The M/A input is 0. The ALU output is 5.

ROM	Asm
0	@0
1	D=M
2	@0
3	M=1
4	D;JGT
5	@0
6	M=0
7	D;JGT
8	@8
9	0;JMP
10	
11	
12	
13	
14	
15	
16	
17	
18	
19	
20	
21	
22	
23	
24	
25	
26	
27	
28	

RAM	
0	1
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0

PC: 0, A: 0, D: 5, M/A Input: 0, ALU output: 5

ELSE PART:

The screenshot displays a computer architecture simulator interface. On the left, the ROM table lists instructions: 0: @0, 1: D=M, 2: @0, 3: M=1, 4: D;JGT, 5: @0, 6: M=0, 7: D;JGT, 8: @8, 9: 0;JMP, and rows 10-28 are empty. Below the ROM table, the PC register is set to 3. In the center, the RAM table shows address 0 containing -1, with addresses 1-28 all containing 0. Below the RAM table, the A register is set to 0. On the right, a large empty box represents the main memory. Below it, a D register is set to -1. At the bottom right, the ALU section shows the D Input as 0 and the M/A Input as -1, with the ALU output also being -1.

ROM	Asm
0	@0
1	D=M
2	@0
3	M=1
4	D;JGT
5	@0
6	M=0
7	D;JGT
8	@8
9	0;JMP
10	
11	
12	
13	
14	
15	
16	
17	
18	
19	
20	
21	
22	
23	
24	
25	
26	
27	
28	

PC: 3

RAM	
0	-1
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0

A: 0

D: -1

ALU

D Input: 0

M/A Input: -1

ALU output: -1

EXPNO:05 MULTIPLICATION OF TWO NUMBERS

AIM:

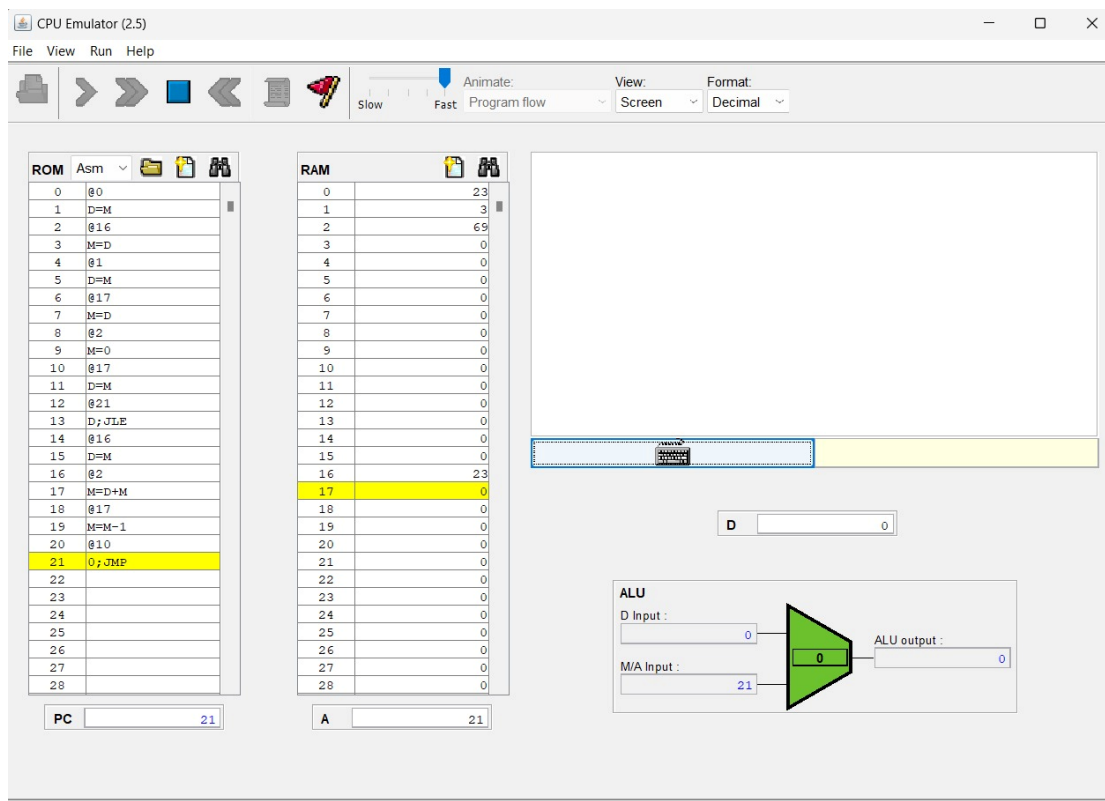
To multiplication in CPU emulator using Assembly Language in Nand2tetris.

HACK ASSEMBLY CODE:

1. @0
2. D=M
3. @16
4. M=D
5. @1
6. D=M
7. @17
8. M=D
9. @2
10. M=0
11. @17
12. D=M
13. @21
14. D;JLE
15. @16

16. D=M
17. @2
18. M=D+M
19. @17
20. M=M-1
21. @10
22. 0;JMP

VERIFICATION SCREENSHOT:



RESULT:

The Multiplication of two numbers is executed successfully using CPU emulator in Nand2tetris using Assembly Language.

DIVISION OF TWO NUMBERS

AIM:

To DIVISION of two numbers in CPU emulator using Assembly Language in Nand2tetris.

HACK ASSEMBLY CODE:

@0

D=M

@2

M=D

@7

M=0

@1

D=M

@2

M=M-D

@7

M=M+1

@2

D=M

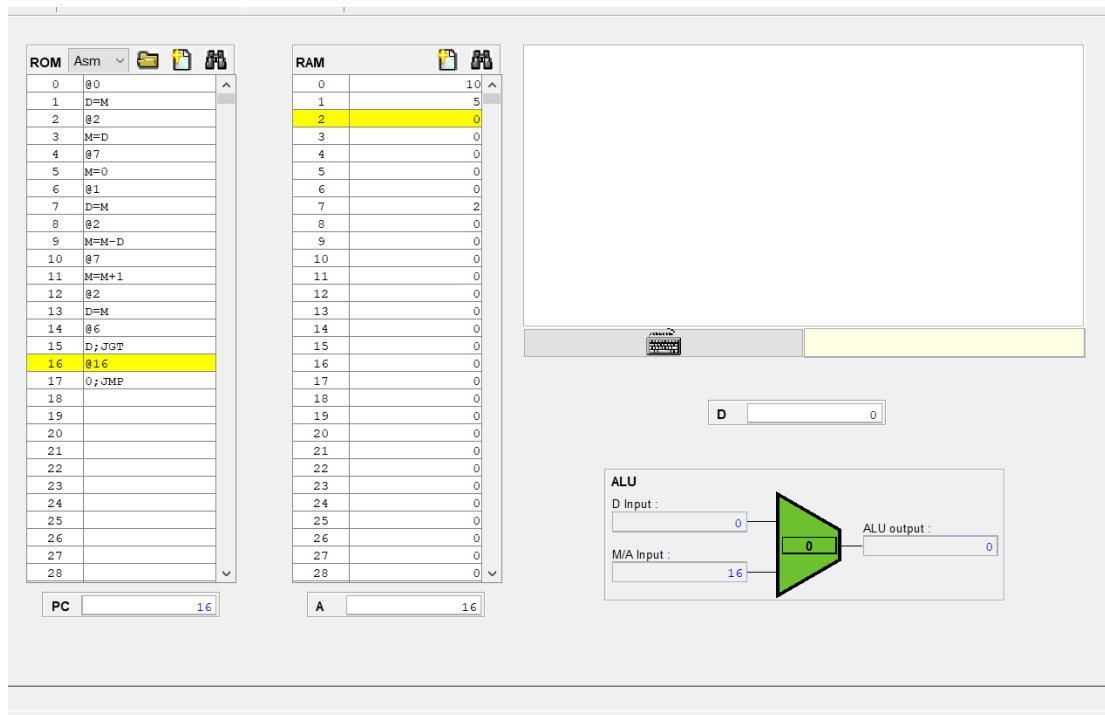
@6

D;JGT

@16

D;JMP

VERIFICATION SCREENSHOT:



RESULT:

The Division of two numbers is executed successfully using CPU emulator in Nand2tetris using Assembly Language.