**Daily Log Sheet**

**User Id : 27292 Name: Sneha Saha**

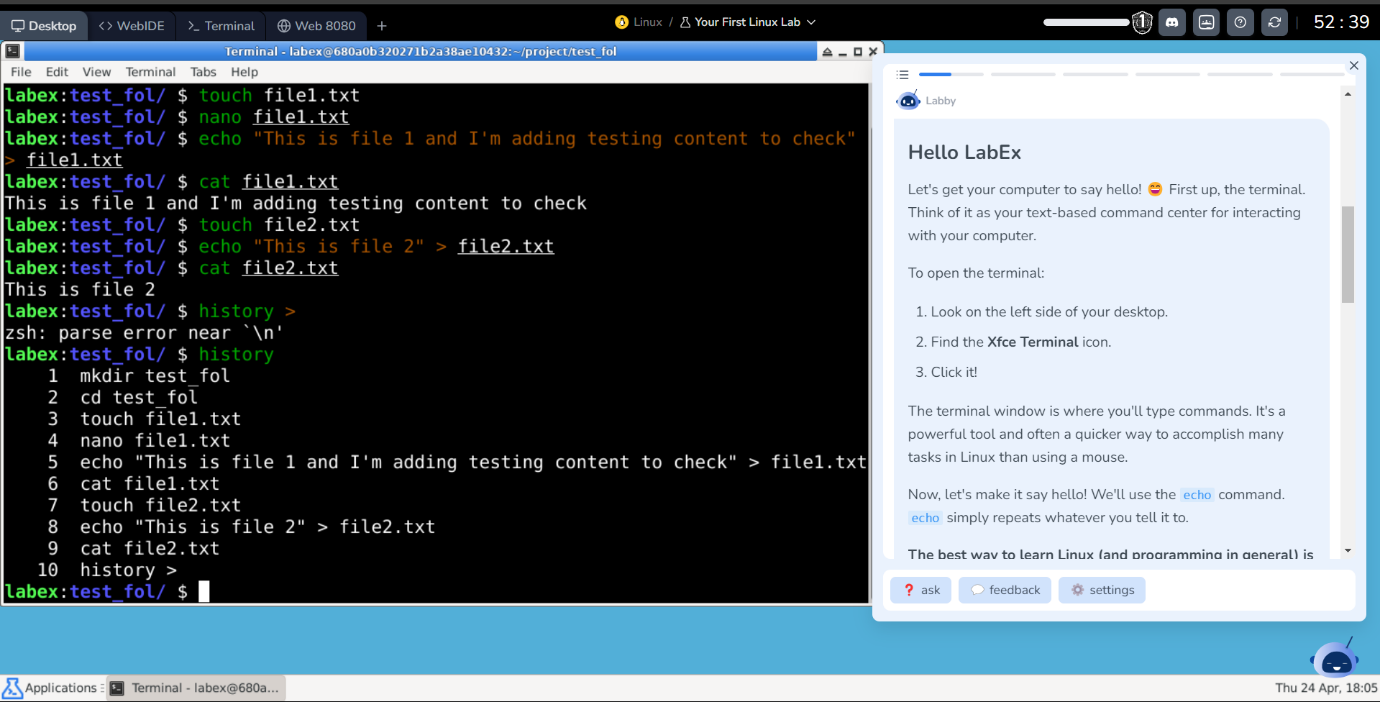
This is Day 1 training sheet

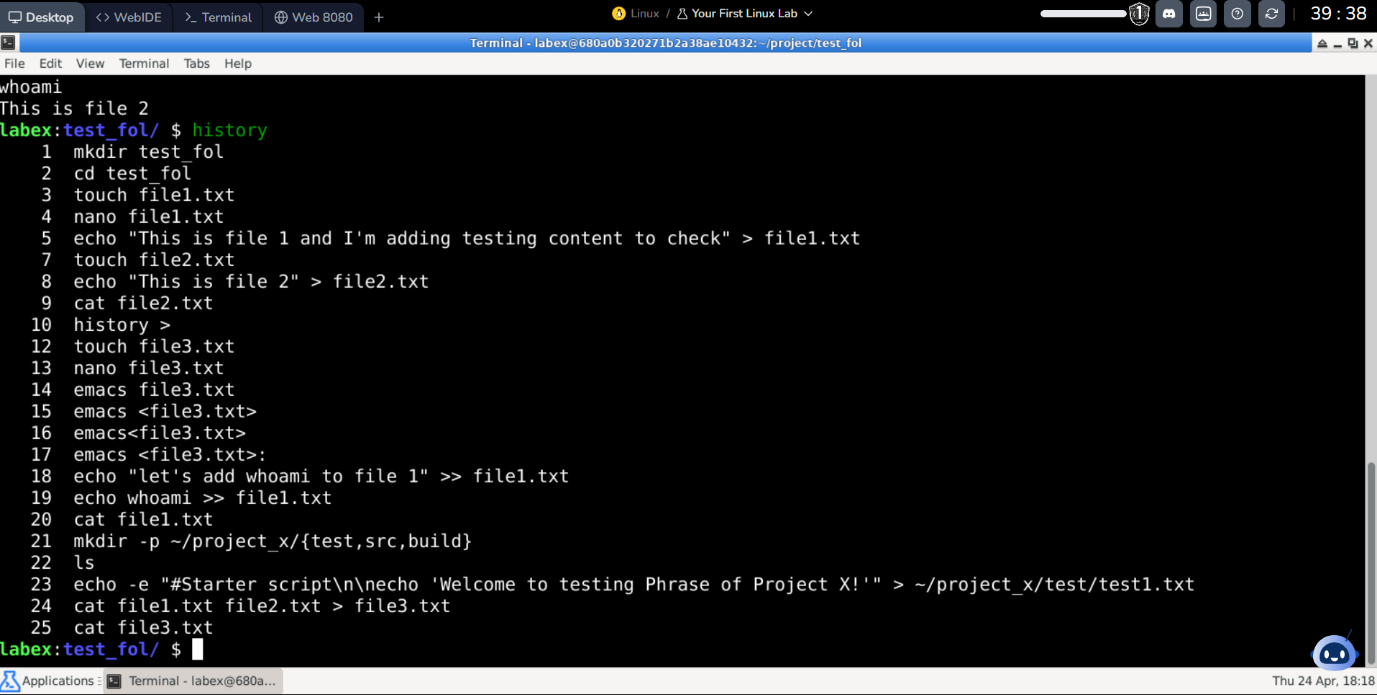
Overview: We learned about the use cases of Linus Operating System.  
The syntax for the CLI or Terminal used :

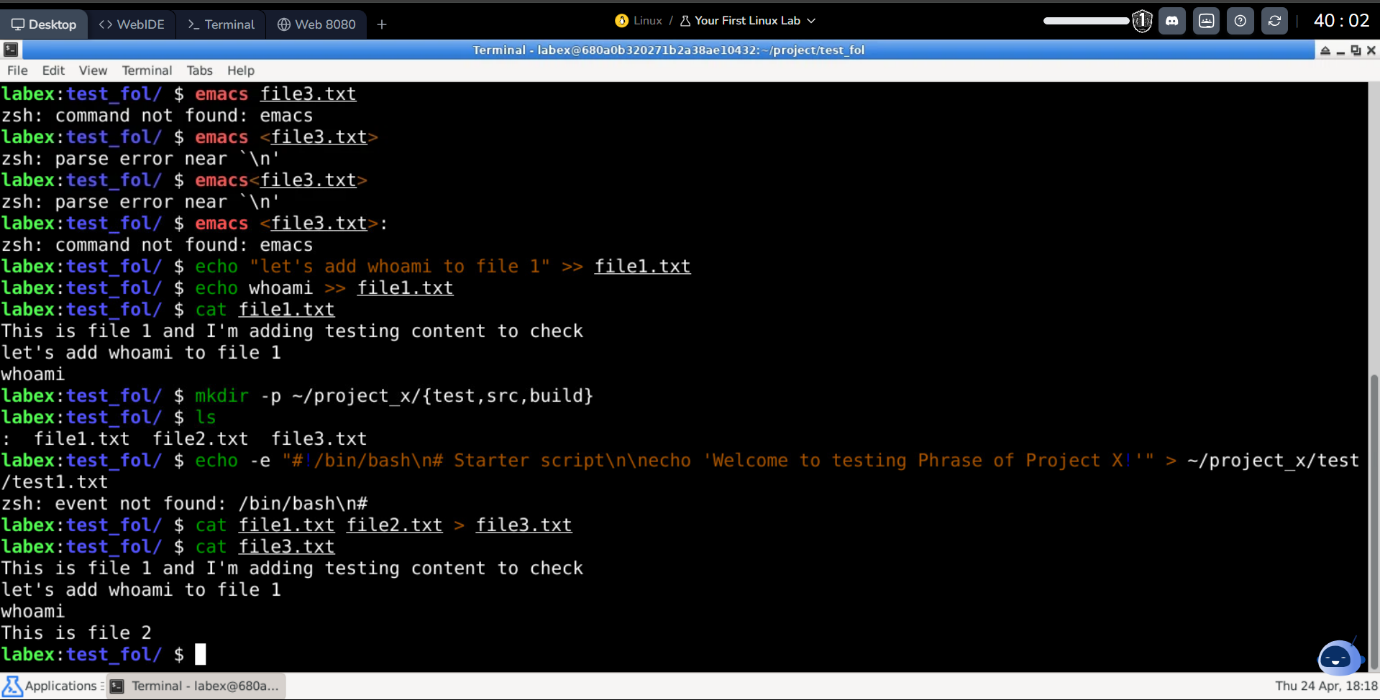
* Cat
* History
* Nano
* Vi/Vim
* Echo
* Alias
* Mkdir
* Rmdir
* Rm -r file1.txt
* Touch
* Cd

I tried few others like bin, bash, emacs, but the current labex.io wasn’t supporting such.

Also, got to learn about the > and >> and also ~ uses, we can overwrite using the ‘>’ but to stop doing so and go to the next line we can use ‘>>’

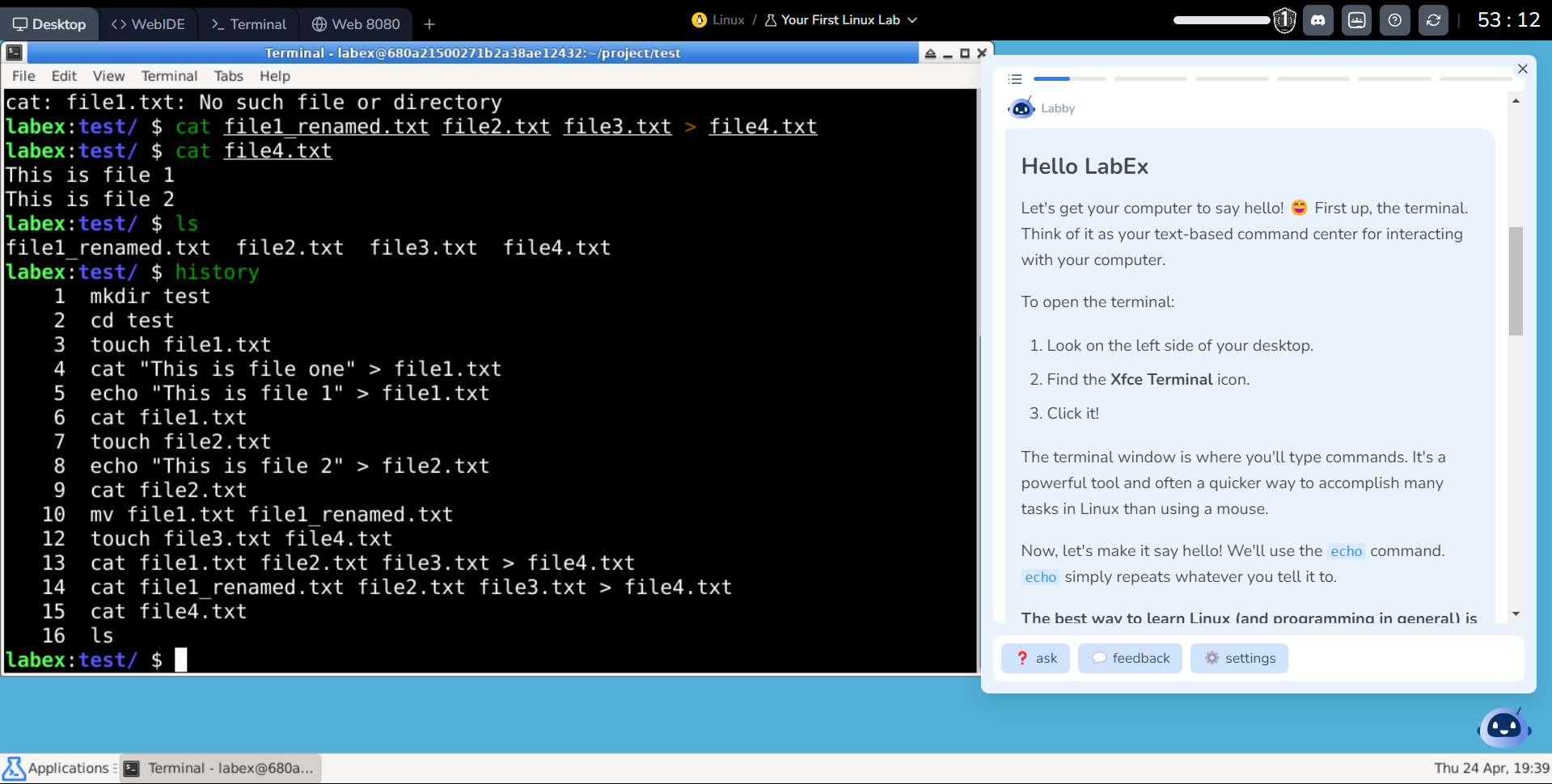


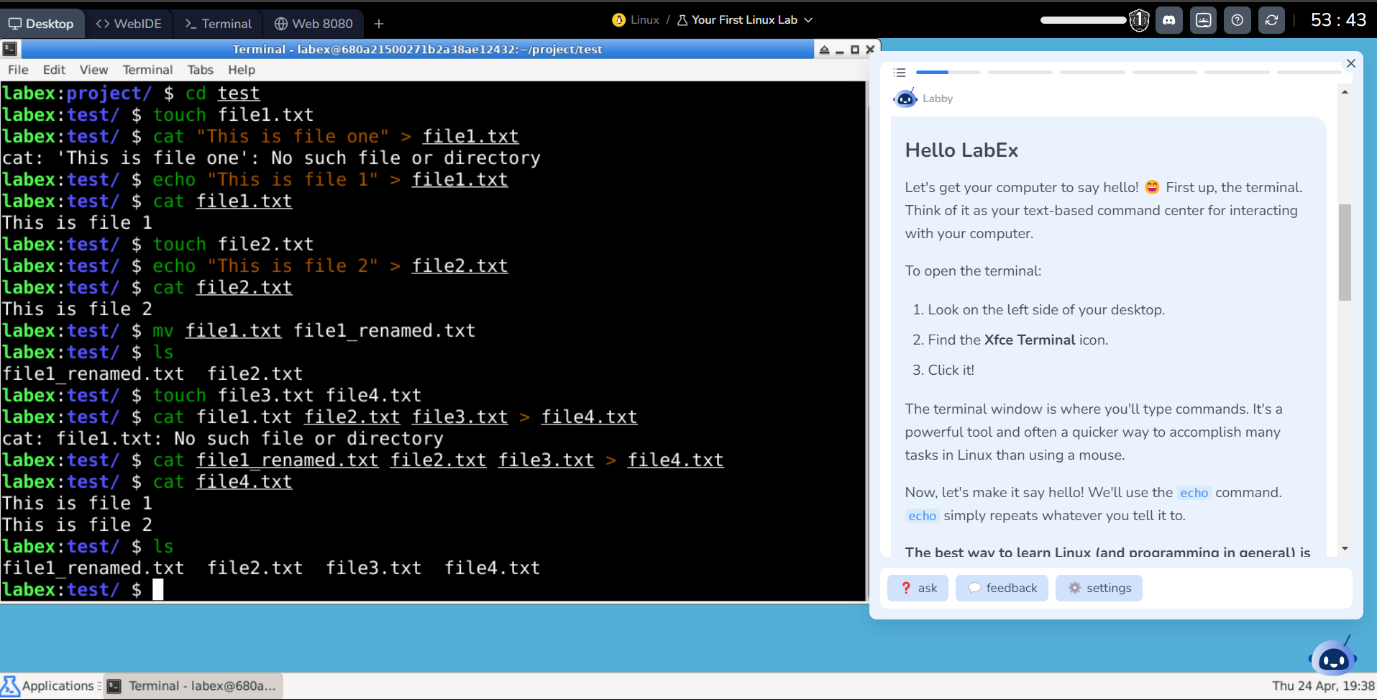




Task for the day :

* Create a directory and a few text files in it.
* Rename one of the files.
* Create multiple Files in single command
* Combine multiple files in a single command and redirect their output to a single file





This is day 2 training sheet

Overview: The number bit representation

Binary system is base 2, so:

2⁰ = 1

2¹ = 2

2² = 4

…

2¹⁰ = 1024

0000000000 (binary) = 0

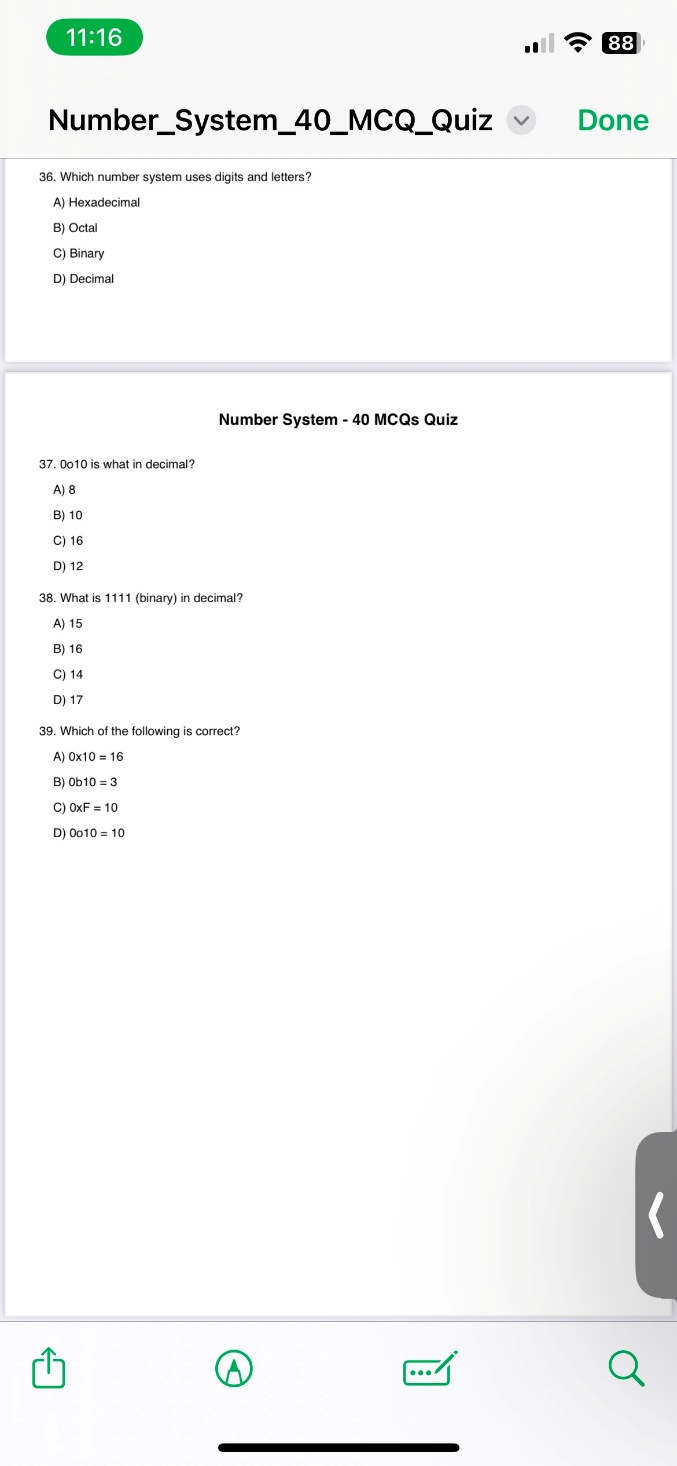
to

1111111111 (binary) = 1023 so it adds upto 1024

512 256 128 64 32 16 8 4 2

Total values: 2^n

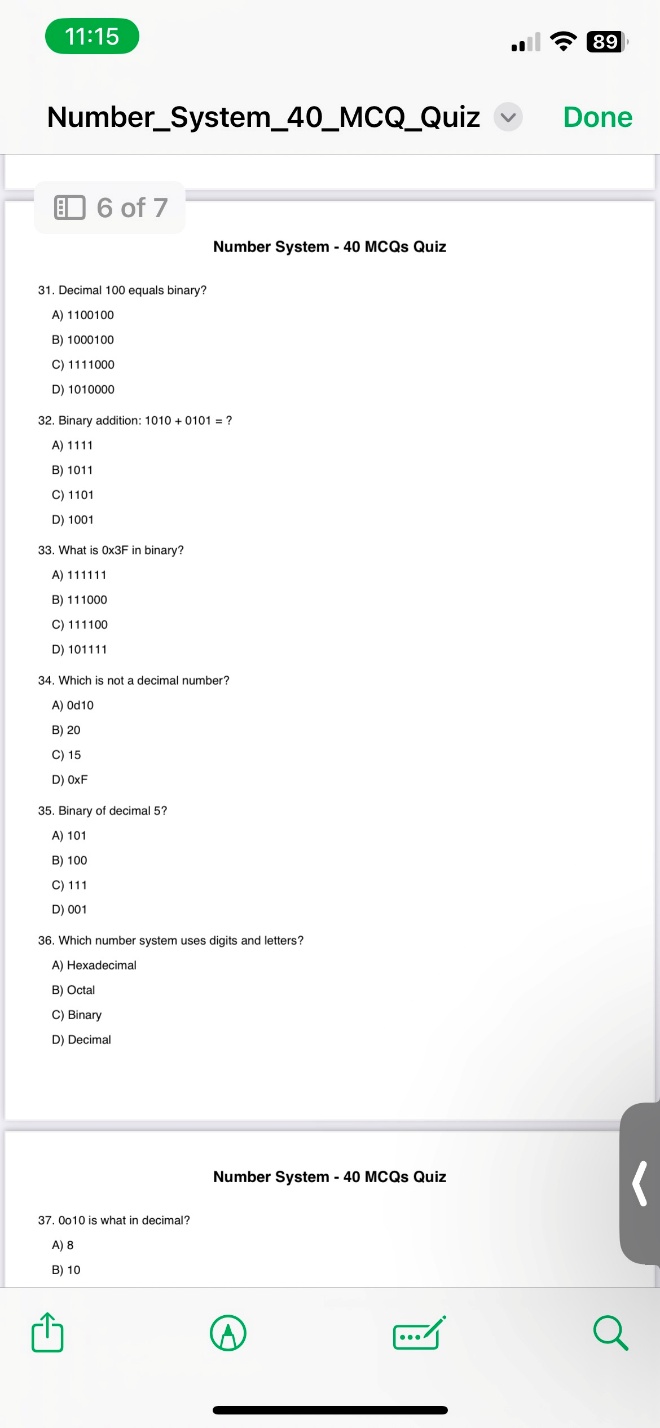
Maximum value: 2^n – 1



37) (B) 10

38) (A) 15

39) (A) 0x10 = 16



31) (A) 1100100

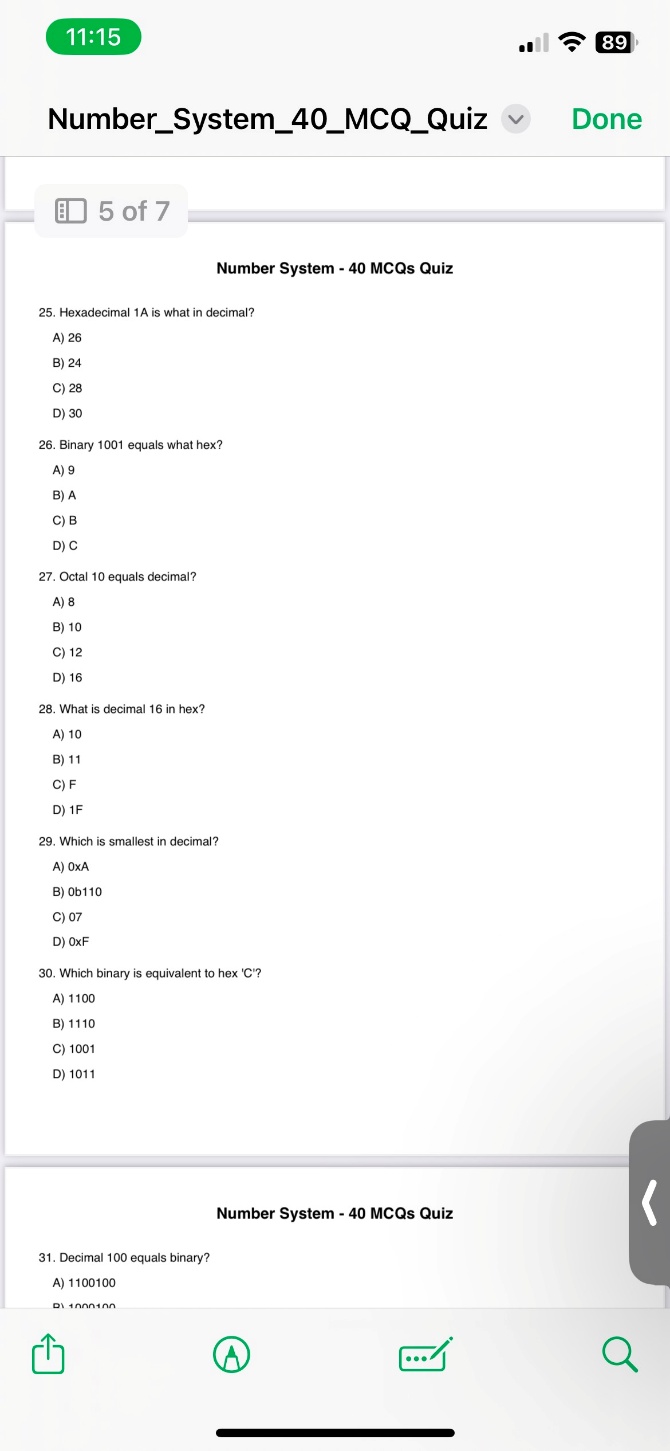
32) (A) 1111

33) (D) 101111

34) (D) 0xF

35) (A) 101

36) (A) Hexadecimal



25) (A) 26

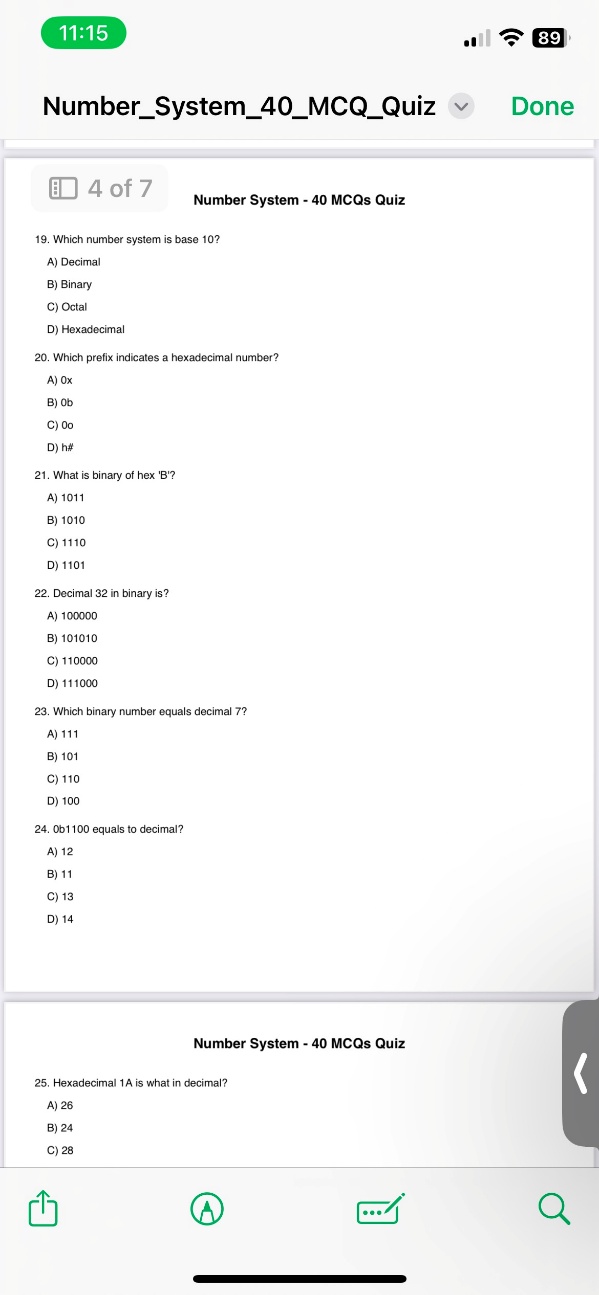
26) (A) 9

27) (A) 8

28) (A) 10

29) (C) 07

30) (C) 07



19) (A) Decimal

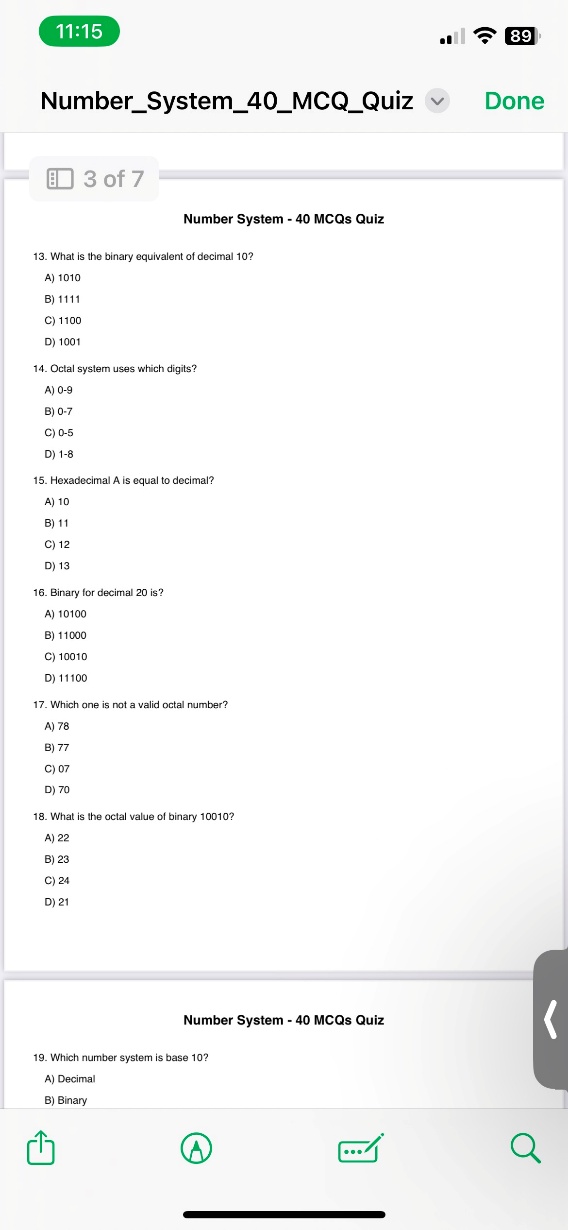
20) (A) 0x

21) (A) 1011

22) (A) 100000

23) (A) 111

24) (A) 12



13) (A) 1010

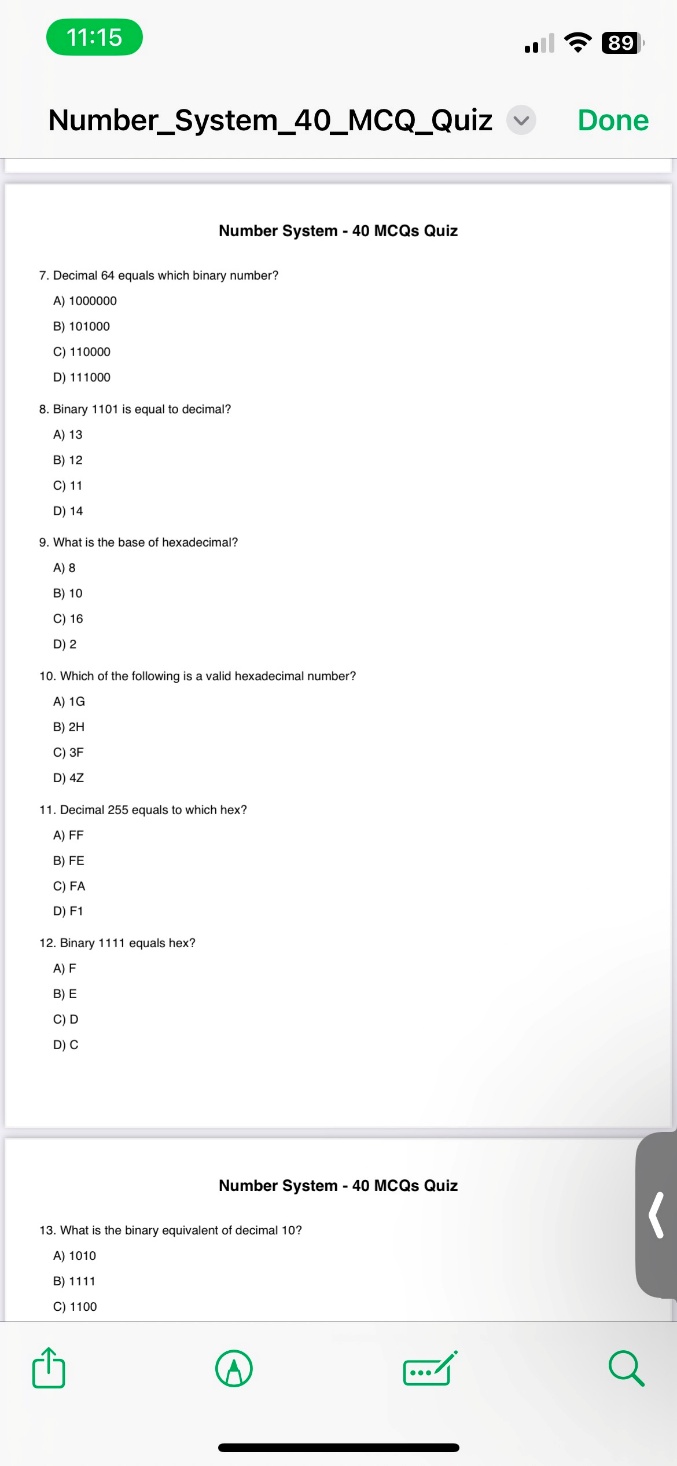
14) (B) 0-7

15) (A) 10

16) (A) 10100

17) (A) 78

18) (A) 22



7) (A) 1000000

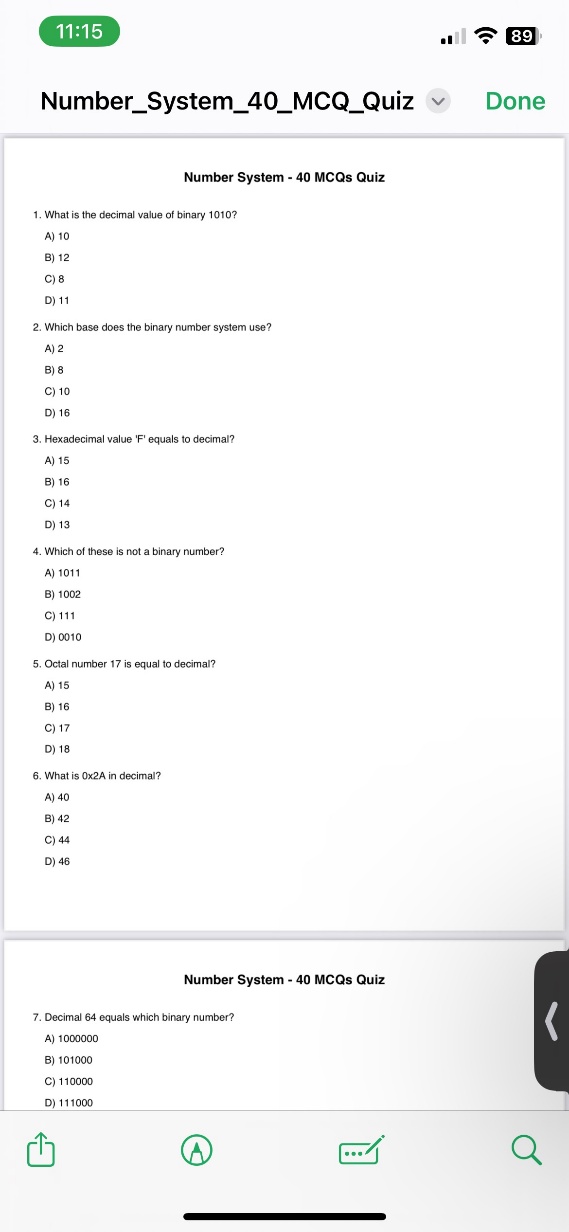
8) (A) 13

9) (C) 16

10) (C) 3F

11) (A) FF

12) (A) F



1) (A) 10

2) (A) 2

3) (A) 15

4) (B) 1002

5) (A) 15

6) (B) 42

8051 Architecture :

We need to write two programs using the 8051-architecture assembly syntax language –

1. Addition

ORG 00000

MOV A, #0FH ; Load 15 (decimal) into accumulator A

MOV R1, #05H ; Load 5 into register R1

ADD A, R1 ; A = A + R1 → A = 15 + 5 = 20

1. If else Conditioning using Jump (SJMP)

ORG 00000

MOV A, #00H ; Load A with a value

JZ IS\_ZERO ; If A == 0, jump to IS\_ZERO

MOV P1, #0FFH ; If A ≠ 0 → set P1 to HIGH (else part)

SJMP END\_IF ; Jump to end of if-else

IS\_ZERO:

MOV P1, #00H ; If A == 0 → set P1 to LOW

END