Data Representation

- Data representation refers to the form in which data is processed, stored and transmitted

Binary Magnitude

from left side

Numbers starting with "0" represents Positive number

(01100011)

↳ first bit is binary magnitude

Numbers starting with "1" represents negative number

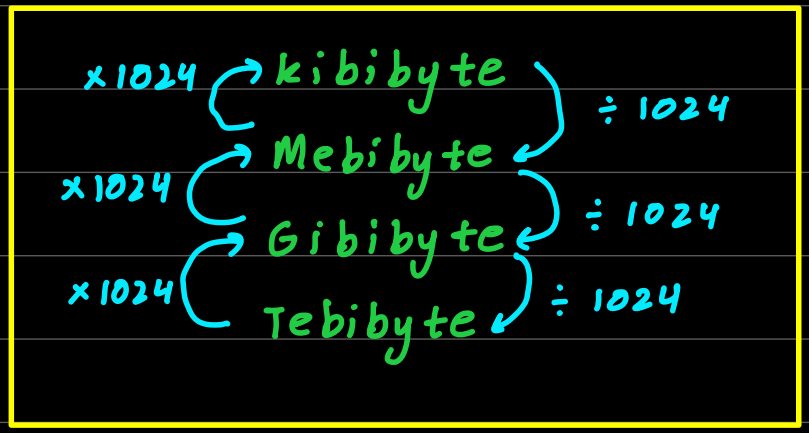
(10001100)

Prefixes

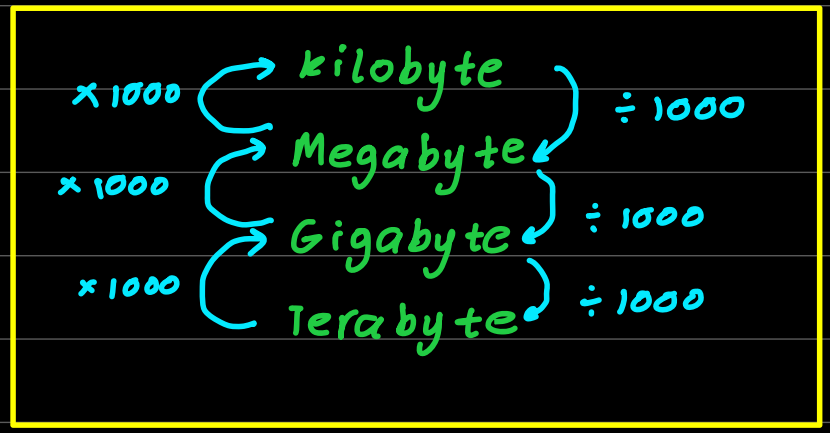
* Using decimal prefixes causes confusion when dealing with bigger numbers

Binary Prefix	Decimal Prefix
· kibibyte	· kilobyte
· Mebibyte	· Megabyte
· Gibibyte	· Gigabyte
· Tebibyte	· Terabyte
1024	1000

Binary Prefixes



Decimal Prefixes

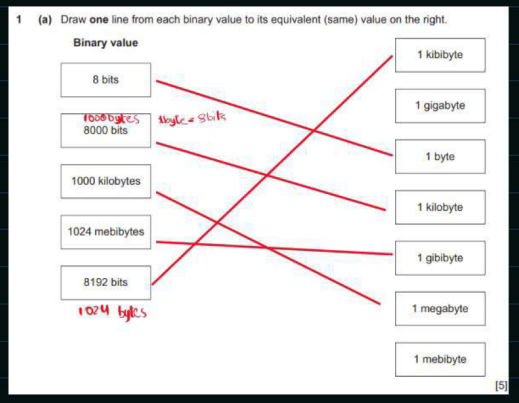


- Smallest unit is bit
- One byte contains 8 bits — 0110 1110 (1 byte)
- 4 bits are known as nibble — 0110 (1 nibble) 1110 (1 nibble)

- One kilobyte contains 1000 bytes
- One kibibyte contains 1024 bytes
- One Megabyte contains 1000 kilobytes
- One mebibyte contains 1024 kibibytes
- One gigabyte contains 1000 megabytes
- One gibibyte contains 1024 mebibytes
- " Terabyte " 1000 gigabytes
- " Tebibyte " 1024 gibibytes

Exam Style Question

1 (a) Draw **one** line from each binary value to its equivalent (same) value on the right.



[5]

Number Systems

- Base number: No. of ways a number can be represented.

Binary (base-2) | **Denary** (base-10) | **Hexadecimal** (base-16)

Note: Memorized list (128 64 32 16 8 4 2 1)

CASE 1: Denary to Binary

Q- Convert 65 into 8-bit binary

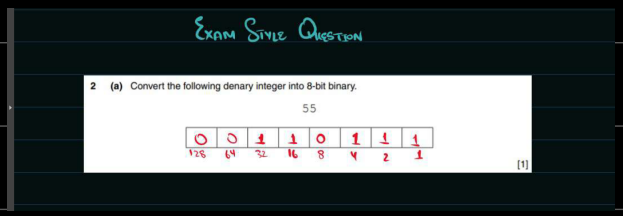
Step 1: Write down memorized list

Step 2: Place 1 on specific place so total could be the same as denary number given

16+16 8+8 4+4 2+2

128	64	32	16	8	4	2	1
0	1	0	0	0	0	0	1

→ Ans



CASE 2 : Binary to Denary

Q- Convert 01000001 into denary

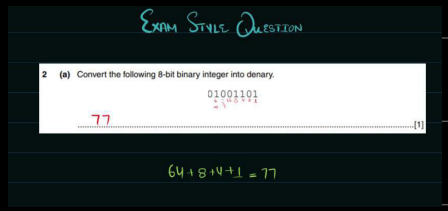
Step 1 : Write down memorized list on the binary number.

Step 2 : Add only the numbers with "1"

128	64	32	16	8	4	2	1
0	1	0	0	0	0	0	1

★ Write memorized list from right hand side

64 + 1 = 65 Ans



CASE 3 : Binary to Hexadecimal

Q- Convert 11011001 into Hexadecimal

Step 1 : From right side, divide the binary value into groups of four. → 1101 1001

A	=	10
B	=	11
C	=	12
D	=	13
E	=	14
F	=	15

Step 2: Write the memorized list seperately for each group

8 4 2 1 | 8 4 2 1
1 1 0 1 | 1 0 0 1

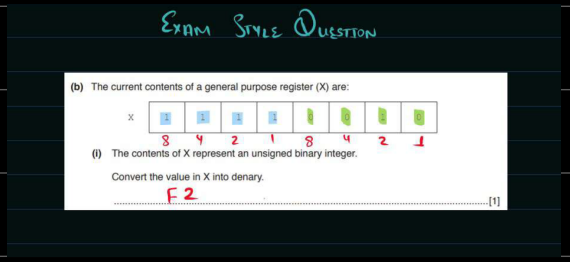
Step 3: Add numbers with 1 and if it is greater than 9, then use value box.

8 4 2 1 | 8 4 2 1
1 1 0 1 | 1 0 0 1

8+4+1 | 8+1

13 | 9

D9 Ans



CASE 4 : Denary to Hexadecimal

Q- Convert 195 to Hexadecimal

Step 1 : First convert the denary number to binary

128 64 32 16 8 4 2 1
1 1 0 0 0 0 1 1

Step 2: Repeat steps in CASE 3

- Complete groups will be made

8 4 2 1 | 8 4 2 1

1 1 0 0 | 0 0 1 1

8+4 | 2+1

12 | 3

C3 Ans

- If complete groups are not forming, add zeroes at left side.

CASE 5 : Hexadecimal to Denary

- Reverse steps in CASE 3

Q- convert 4E into denary

8 4 2 1 | 8 4 2 1

0 1 0 0 | 1 1 1 0

128 64 32 16 8 4 2 1

0 1 0 0 1 1 1 0 ⟶ 8+4+2+64 = 78 Ans

Binary Coded Decimal

- Uses a 4 bit code to represent each denary digit.

0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

1001
8 4 2 1

* A type of format of Binary number system

Note: Any number greater than 9 can not be represented in BCD (4-bit)

CASE 1: Denary to BCD

Q- Convert 54 into BCD

Step 1: Convert each digit into BCD format.

5	4
8 4 2 1	8 4 2 1
0 1 0 1	0 1 0 0

→ 0101 0100 Ans

Exam Style Question

(b) (i) Convert the following denary integer into Binary Coded Decimal (BCD).

653

0110 0101 0011 [1]

CASE 2 : BCD to Denary

Q- Convert 01010100 BCD to Denary

Step 1: Make groups of 4

*Make groups from right hand side.

0101 0100

Step 2: Write memorized list on each group and add the numbers with 1

5	4
8 4 2 1	8 4 2 1
0 1 0 1	0 1 0 0

→ 54 Ans

Exam Style Question

(b) Convert the following Binary Coded Decimal (BCD) number into denary.

10000011

83 [1]

Q- Describe how denary integers larger than 9 can be converted into BCD. Give an example in your answer.

* Always give example

- Each Denary digit is written as 4-bit binary number (1)
- 46 → 0100 0110 (2)

Q- Describe how a 8-bit BCD can be converted into a denary integer. Give an example in your answer

- Binary number is split into groups of 4 bits, starting from the right.
- Each group of 4-bits is converted into a denary digit
- E.g: 0011 0111 → 37

Q- Identify practical applications where BCD is used.

- Any scenario where a single digit needs to be transmitted / displayed

- calculator, digital clock.

Binary Addition

	Sum	Carry
0 0	0	0
0 1	1	0
1 0	1	0
1 1	0	1
1 1 1	1	1

(ii) The amount of green in binary is `00100011`. This has the denary number `15` added to it to create a second colour.

Add the denary number `15` to the binary number `00100011` and give your answer in binary.

Perform the addition in binary. Show your working.

Working ..... 00100011 + 00001111 = 00110010

Answer (in binary) 00110010

[3]

(b) (i) Perform the following binary addition. Show your working.

```
  10101010
+ 00110111
------------
 11100001
```

[2]

Conversion of Negative Numbers

Q- Convert -125 to Binary Form.

Step 1: Ignore '-' sign and convert it to Binary.

128	64	32	16	8	4	2	1
0	1	1	1	1	1	0	1

0 1 1 1 1 1 0 1

* If answer is not forming, make different combonition.

Step 2: Apply 2's compliment method.

0 1 1 1 1 1 0 1

1 0 0 0 0 0 1 0 → Inverse (One's compliment)

1 → 2's compliment

1 0 0 0 0 0 1 1

Answer: 10000011

Q- Convert the denary number -194 into 12-bit two's compliment

128	64	32	16	8	4	2	1
1	1	0	0	0	0	1	0

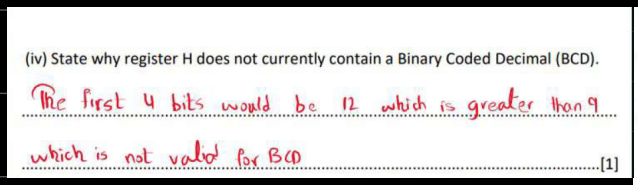
means two's compliment of all 12-bits

=> 000011000010

```
  111100111101
+             1
--------------
  111100111110
```

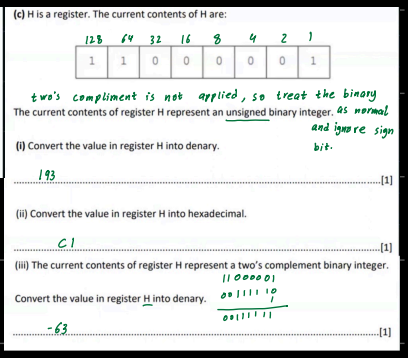
→ 111100111110

Note: All 2's compliment numbers that start with 1 are negative numbers



(iv) State why register H does not currently contain a Binary Coded Decimal (BCD).

The first 4 bits would be 12 which is greater than 9 which is not valid for BCD [1]



(c) H is a register. The current contents of H are:

128	64	32	16	8	4	2	1
1	1	0	0	0	0	0	1

The current contents of register H represent an unsigned binary integer.

two's compliment is not applied, so treat the binary as normal and ignore sign bit.

(i) Convert the value in register H into denary.

193 [1]

(ii) Convert the value in register H into hexadecimal.

C1 [1]

(iii) The current contents of register H represent a two's complement binary integer.

Convert the value in register H into denary.

11000001
00111110
+ 1
00111111

-63 [1]

Binary Subtraction

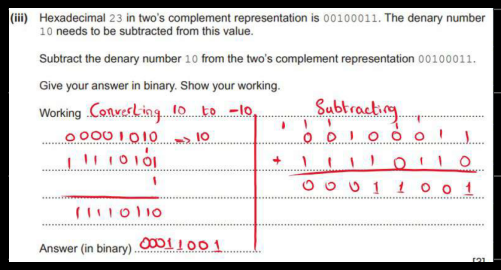
Explaination:

40 — 15 = 25

OR

40 + -15 = 25

Note: Apply two's compliment and then same working as binary addition



(iii) Hexadecimal 23 in two's complement representation is 00100011. The denary number 10 needs to be subtracted from this value.

Subtract the denary number 10 from the two's complement representation 00100011.

Give your answer in binary. Show your working.

Working Converting 10 to -10 | Subtracting

00001010 → 10 | 00100011

11110101 | + 11111010

1 | 00011001

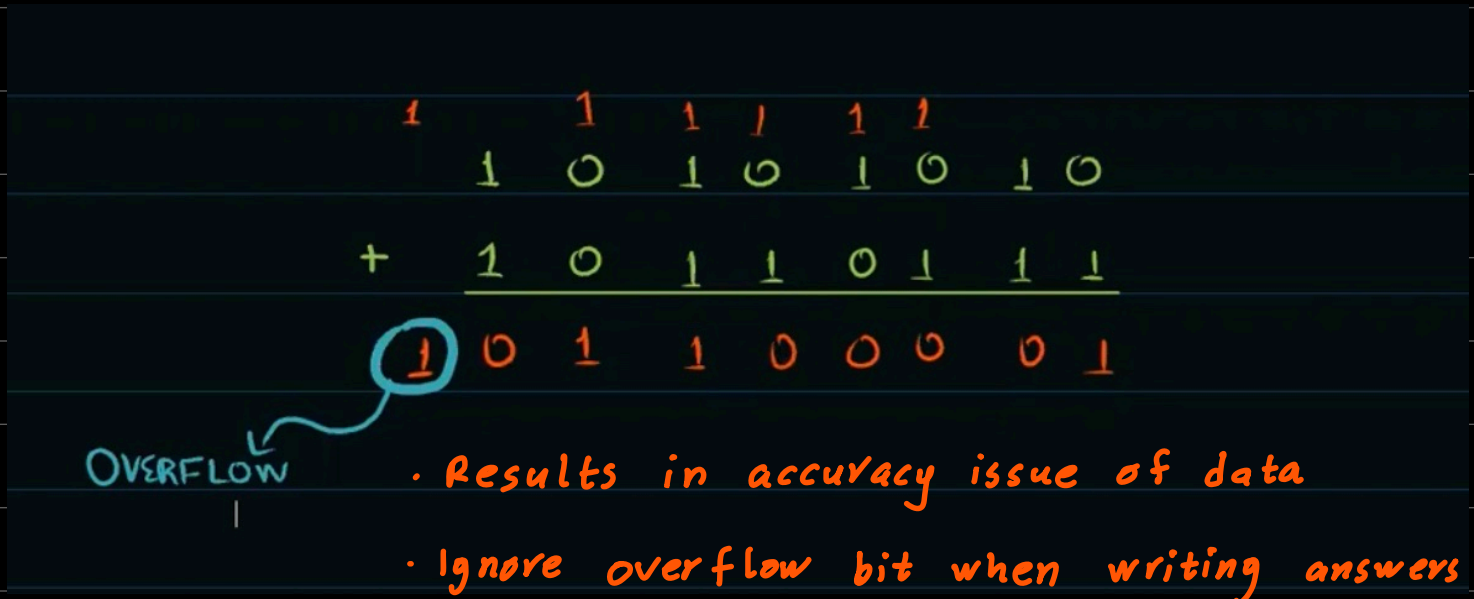
11110110

Answer (in binary) 00011001

[2]

Overflow

Concept: More bits less space to store



Q- State how an overflow can occur when adding two binary integers

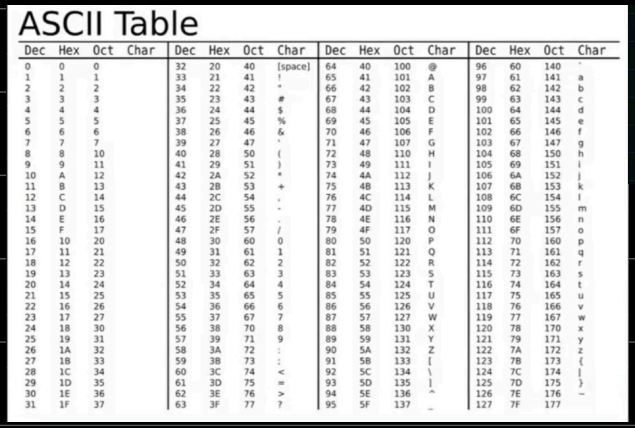
· The result is a larger number that can not be stored in the given number of bits

ASCII , Extended ASCII , Unicode

Note: The numbers could be easily represented into binary form, but there was no way to represent alphabets / characters

A = 65

a = 97



ASCII

- 7 bits are used for character representation
- 128 characters can be represented (127 + 1)

64 32 16 8 4 2 1 = A
1 0 0 0 0 0 1

Extended ASCII

- 8 bits for character representation
- ASCII codes were 7 bit long, but then it was extended to have bit length of 8
- 256 characters can be represented.

Unicode

Problem:

→ Less number of bits for representation of characters of other languages.

- Uses 16, 24, 32 bits to represent characters.

Q- What are the disadvantages of using ASCII code?

- Only 128 characters can be represented
- Uses values 0 to 127
- Many characters in other languages can not be represented
- In extended ASCII, the characters from 128 to 255 may be coded differently in different systems. (No standard set)

Q- How Unicode is designed to overcome disadvantages of ASCII?

- Uses 16, 24, 32 bits
- Unicode is designed to be superset of ASCII
- Designed so that most characters in other languages can be represented

(b) Explain how a word such as 'HOUSE' is represented by the ASCII character set.

- Each character has its own unique code
- Each character in the word is replaced by its code
- The codes are stored in the order in the word.

[2]