Huffman Coding.

↳ fixed length code

↳ variable length code.

1) fixed length.

Message – BCCABBDDAECCBBAEDDCC

length = 20.

It has to be sent by ASCII codes. (8 bits)

If 8 bits are used to encode 20 alphabets 160 bits are required.

character	count		code
A	3	3/20	000
B	5	5/20	001
C	6	6/20	010
D	4	4/20	011
E	2	2/10	100
	20		

8 bits are for 128 characters

to represent 5 alphabets 3 bits are sufficient.

010.

001010 . . . . . .

$20 \times 3 = 60$ bits → encoder requires 60 bits

But each symbol need to be decoded at the receiving end. ∴ Table also should be transmitted.

5 alphabets → Requires ASCII codes → $5 \times 8 + 5 \times 3 = 40 + 15 = 55$

5×8 → alphabets; 5×3 → codes

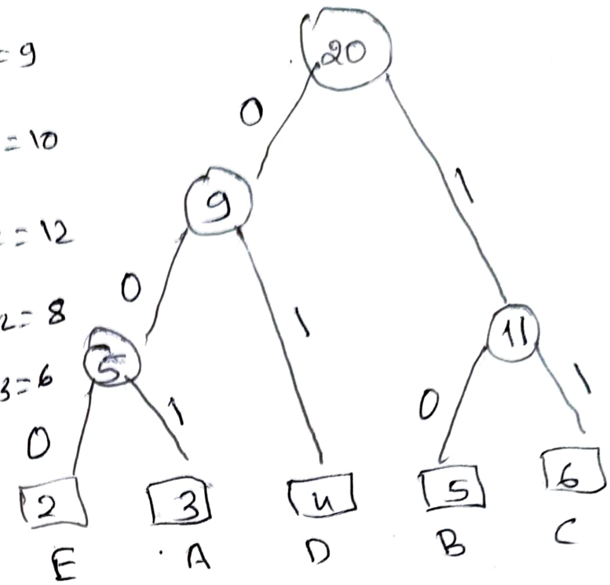
Message	60 bits
Table	55 bits
	115

115 : 160

variable length code.

(2)

char	count	code	
A	3	001	3×3=9
B	5	10	5×2=10
C	6	11	6×2=12
D	4	01	4×2=8
E	2	000	2×3=6



optimal merge pattern
(greedy method)

B	C	C	. . . . .	C
10	11	11		11

Total bits required to send = 45 bits.

message size — 45 bits.

Table / tree size →	Alphabets		codes
	5×8=40	+	12 = 52 bits

45 + 52 = 97 bits to transmit the message.

$\boxed{97 : 160}$

Run length coding

00000 11111 00 1 0000 1 0 1

5 5 2 1 4 1 1 1

$\Rightarrow$ (0,5) (1,5) (0,2) (1,1) (0,4) (1,1) (0,1) (1,1)

1 2 3 4 5 6 7 8

$\Rightarrow \log_2 8 = \log_2 2^3 = 3 \log_2 2 = 3$ bit

$\Rightarrow$ (0, 101), (1, 101) (0, 010), (1, 001) (0, 100), (1, 001) (0, 001), (1, 001)

final code.

0101 1101 0010 1001 0100 1001 0001 1001

1111111111111110000000000000000000111

(15,1), (19,0), (4,1)

$\boxed{2^5 = 32}$

$2^4 = 16$

(01111, 1) (10011, 0) (00100, 1)

Total bits = 38

011111 100110 001001

18 bits.

encoded bits = 18

compression ratio = 18/38

= 1/2.11

$\Rightarrow$ 1 : 2.11