Practice problems CSE 2320, Spring 2018 : Dynamic Programming, Greedy.

Book (CLRS) problems:

1. Dynamic programming: 15.4-1, 15.4-2, 15.4-3 and 15.4-5 (here justify the complexity of all steps you take). (page 396)
2. Greedy: 16.1-3 (page 422)

Other problems:

1. **Other sources:** Problems/issues discussed in class, Homework problems
2. **Huffman encoding** (another example)

Assume that the numbers given below represent counts of letters in the hundreds from a file (similar to the CLRS example). For example, in the file there will be exactly 20 * 100 occurrences of the letter 'a' , 11*100 occurrence of the letter 'c', etc. **a: 20, c:11, d:2, e: 10, o:15, m:8, s:10, t:22, u: 2**

a) What is an optimal Huffman code based on the following set of frequencies?
    a. Draw the tree. Show your work at every step.
    b. Give the Huffman encoding for each letter.
    c. We encode the file using the Huffman codes produced above. How much memory will the file require with this encoding?

b) Fixed-length encoding:
    a. Give a fixed-length encoding for the characters in this file.
    b. What will be the file size when the fixed-length encoding is used?

3. Dynamic programming (smaller problem versions, and new problems):

a. Given solution partially filled out, finish filling it out.

i. Fill out the remaining of the solution table (for columns 13 and 14). **You must show your work in the rows for A,B,C,D as shown in class** (even though for the provided solution (columns 0 to 12) the work is not shown in rows A,B,C,D):

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Sol:	0	0	0	0	10	10	21	21	21	21	33	33	42	**42**	**43**
Picked					A	A	B	B	B	B	C	C	B	**B**	**A**
A,4,10														**9,31**	**10, 43**
B,6,21														**7,42**	**8,42**
C,10,33														**3,33**	**4,43**
D,12,36														**1,36**	**2,36**

ii. Fill out the last two rows in the table below:

		N	**O**	**N**	**S**	**T**	**O**	**P**
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**

R	1	1	2	3	4	5	6	7
O	2	2	1	2	3	4	5	6
U	3	3	2	2	3	4	5	6
N	4	3	3	2	3	4	5	6
D	5	4	4	3	3	4	5	6

iii. Fill-out the last two rows in the edit distance table below. (Here STEM is the complete second word. ME is the end of the first word. For example the first word could be NAME, TESTME).

		S	T	E	M
…	…	…	…	…	…
…	3	3	2	2	3
M	4	4	3	3	2
E	5	5	4	3	3

iv. Similar for other DP problems: be able to finish an iterative solution given a partial answer.

b. Given the edit distance table (with the cost or the distance), add the retracing arrows for:

i. all cells (this is a bit long for an exam, but may still be asked for a small example). Give the time and space complexity for this process. Time: Θ(NM) Space: Θ(NM)

ii. **only the cells visited to recover the solution choices**. Give the time and space complexity for this this process. Time: O(N+M) Space: Θ(N+M)

iii. **the last two rows.** Give the time and space complexity for this process. **Time Space: Θ(M) where M is the length of the horizontal string in the table.**

		N	O	N	S	T	O	P
	0 ↖	1	2	3	4	5	6	7
R	1	1 ↖	2	3	4	5	6	7
O	2	2	1 ↖	2 ←	3 ←	4	5	6
U	3	3	2	2	3	4 ↖	5	6
N	4 ↑	3 ↖	3 ↑	2 ↖	3 ↖	4 ↖	5 ↖	6 ↖
D	5 ↑	4 ↑	4 ↖	3 ↑	3 ↖	4 ↖	5 ↖	6 ↖

c. Similar for other DP problems: given the gain/cost solution, recover the **solution choices** that gave this optimal value.

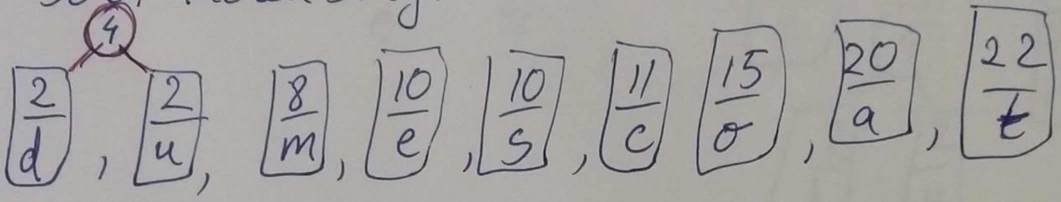
Solution to Problem 2 (Huffman encoding):

Greedy

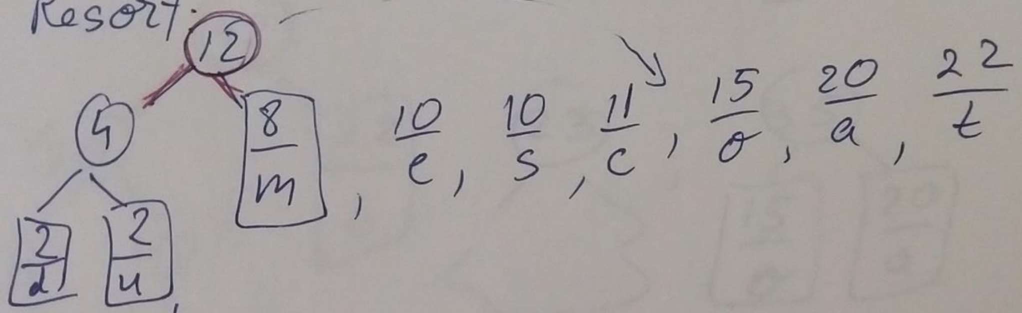
2. Huffman encoding

a:20, c:11, d:2, e:10, o:15, m:8, s:10, t:22, u:2

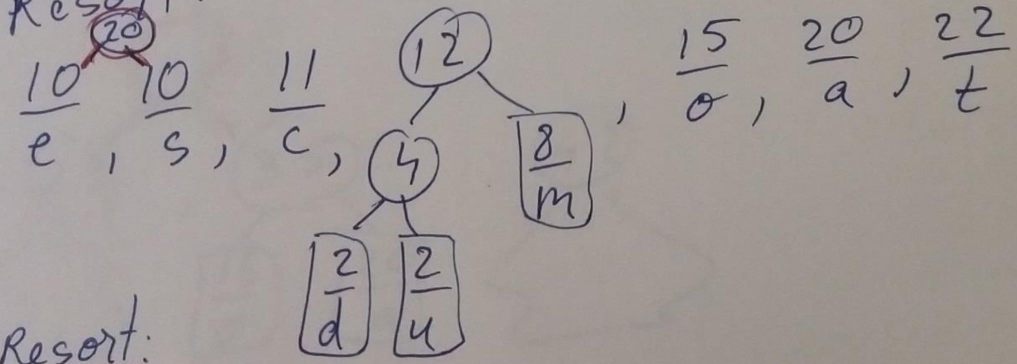
Sort increasing:



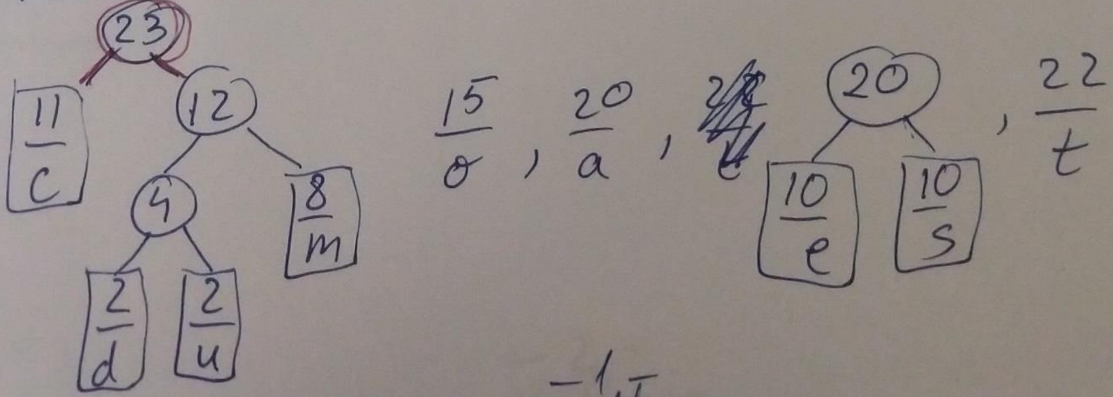
Resort:



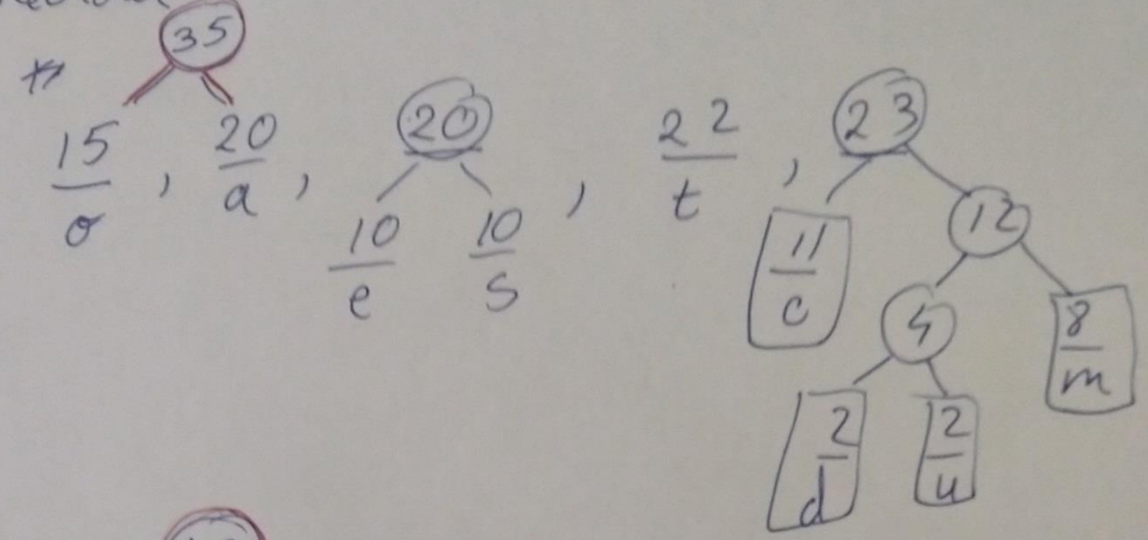
Resort:

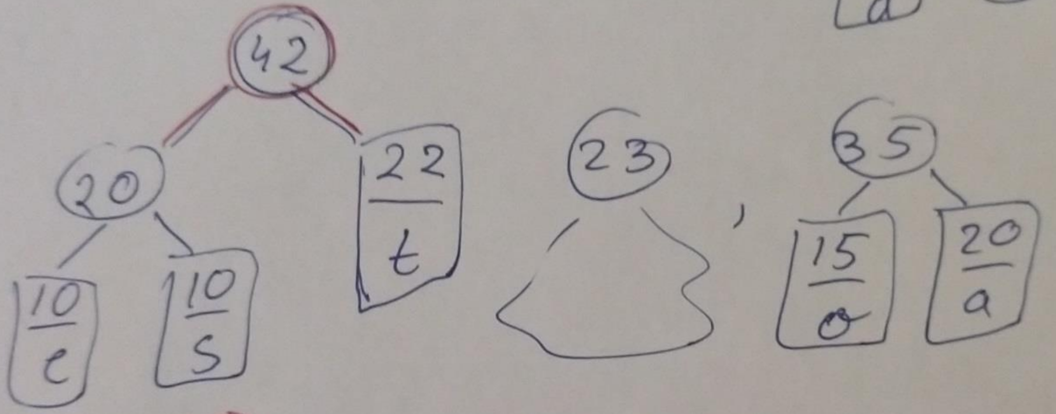


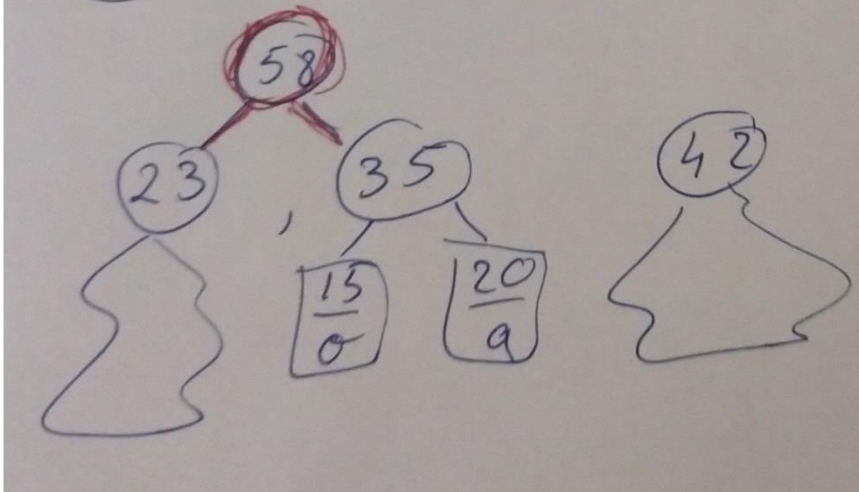
Resort:

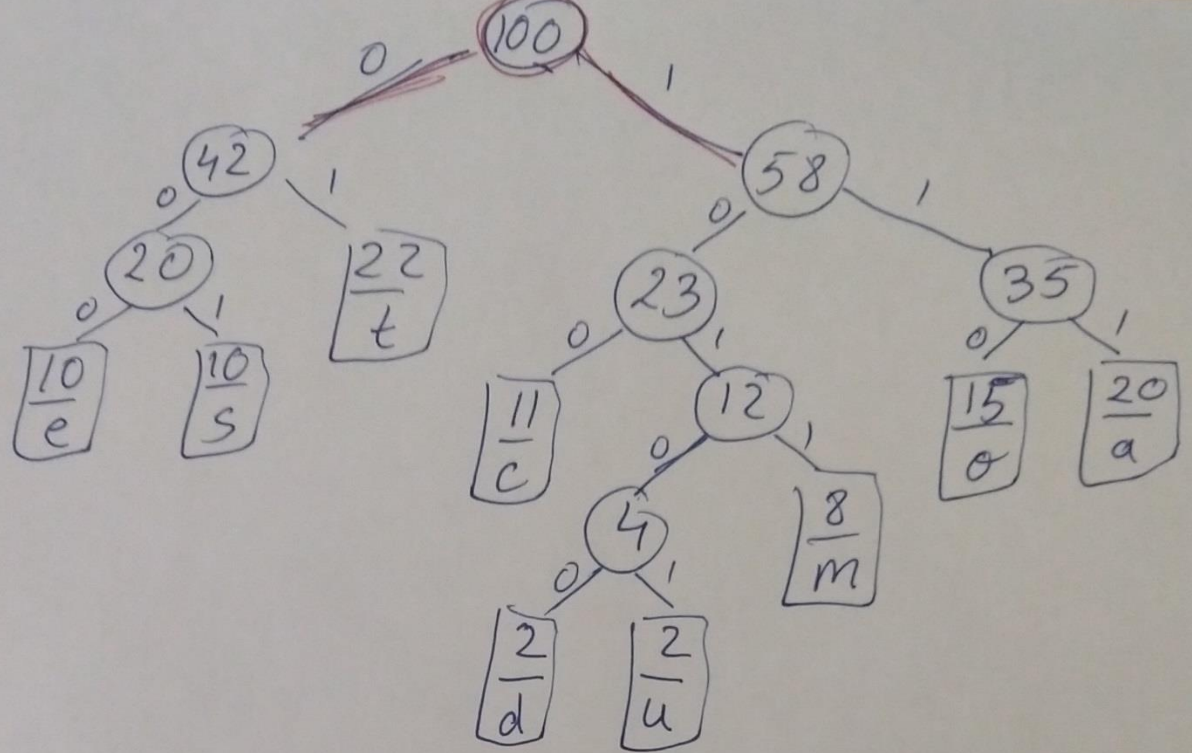


Reorder:


35
15
o
20
a
20
10
e
10
s
22
t
23
11
c
12
4
2
d
2
u
8
m


42
20
10
e
10
s
22
t
23
35
15
o
20
a


58
23
35
15
o
20
a
42



Tree is complete. ~~Ret~~,
Label 0/1 on branches.

ⓐⓑ encoding:

e	000 (3bits)
s	001
t	01
c	100
d	10100
u	10101
m	1011
o	110
a	111

=> ⓐⓒ $\text{Space} = \sum_{\text{letter}} \text{count} * \text{bits}$

$= 10\cdot 100\cdot 3 +$
$10\cdot 100\cdot 3 +$
$22\cdot 100\cdot 2 +$
$11\cdot 100\cdot 3 +$
$2\cdot 100\cdot 5 +$
$2\cdot 100\cdot 5 +$
$8\cdot 100\cdot 4 +$
$15\cdot 100\cdot 3 +$
$20\cdot 100\cdot 3$
$= 29400$ bits

(Note for e, we have: 10·100 count · 3bits)

–3
Huffman

(b) Fixed length:

a) 9 symbols => need 9 different encoding

=> need 4 bits

a	0000
b	0001
d	0010
e	0011
o	0100
m	0101
s	0110
t	0111
u	1000

b) Size: $= \left(\sum_{\text{letter}} \text{count}\right) \cdot 4 =$

$= 4(20+11+2+10+15+8+10+22+2) \cdot 100$

$= 4 \cdot 100 \cdot 100 = 40{,}000$ bits

Huffman