

Jin Cheng,
Jt PP34.

Version B.

1. True or False.

(a) T

(b) F

(c) F

(d) T

(e) T

(f) F

(g) F

(h) ~~⊗~~ ~~⊗~~ F

(i) T

(j) F

2. Which of the following ... DFS are true?

C, d, ~~a~~ b. \leftarrow DFS still can find shortest path but
time complexity extreme.

3. Which of the following ... $O(V+E)$?

~~⊗~~ d

h 2 221

C.

(a) First r.

4. Which of the following algo efficient implementa

D. Floyd..

5. In --- selection problem...

~~a~~ a. b.

6. Which of the following -- shortest path are true?

a

7. Which of the following -- are true?

~~a~~ b

8.

中/英

cont

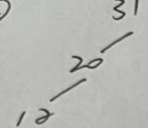
8.

①



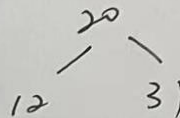
⇒

②



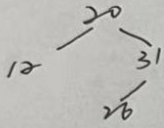
right
R
⇒

③



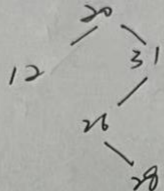
④

⇒

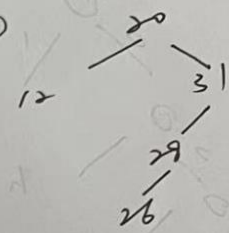


⑤

⇒



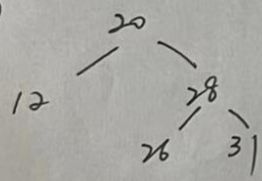
left rotate
⑥
⇒



RR

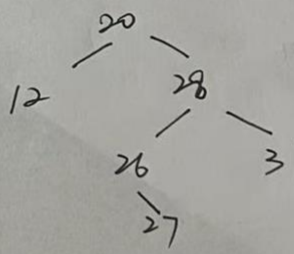
⑦

⇒

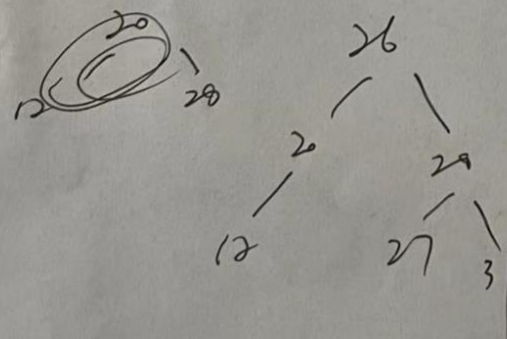


⑧

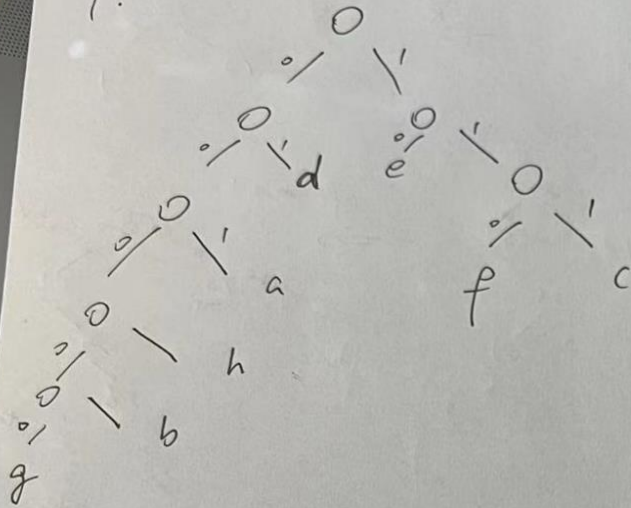
⇒



⇒



9.



a : 001

b : 0001

c : 111

d : 01

e : 10

f : 110

g : 00000

h : 0001

10.

(a) First Five:

$A \rightarrow E \rightarrow G \rightarrow C \rightarrow B \rightarrow F$

$\Rightarrow (A,E) \Rightarrow (E,G) \Rightarrow (G,C) \Rightarrow (C,B), (E,F)$

(b)

Kruskal started with the vertex with lowest weight.

first, sort the weights.

\therefore we have.

weight	source	Dest.
1	A	E
2	H	I
3	E	G
4	G	C
5	A	G
6	C	B
7	C	F
8	D	G

\Rightarrow when adding 5, there is a cycle skip.

\Rightarrow we will reject adding if cycle exists.

~~\therefore the order will be.~~

~~$(A,E) \Rightarrow (H,I) \Rightarrow (E,G) \Rightarrow (G,C) \Rightarrow (C,B)$~~

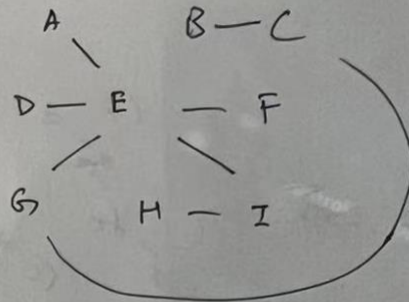
\therefore last five are.

$(G,C), (B,C), (E,F), (E,I), (D,E)$

$\Rightarrow (E, G) \Rightarrow (G, C) \Rightarrow (C, B), (EF)$

id with
weights

	10.6	1	A	E	✓
		2	H	I	✓
Dest		3	E	G	✓
E		4	G	C	✓
I		5	A	G	X
		6	B	C	✓
G	③	7	C	E	X
		8	E	F	✓
C		9	E	I	✓
G		10	D	E	✓
		11	G	H	X
B		12	A	B	✓
		13	B	E	X
F		14	F	I	X
G		15	C	F	X
		16	D	G	X



11. (a)

Function. Algorithm.

$m \leftarrow$ length of B

$n \leftarrow$ length of A

DP \leftarrow two-dimensional table with size $(m+1) \times (n+1)$
with initial value of zeros.

For $i = 1, 2, \dots, m+1$

For $j = 1, 2, \dots, n+1$

\Rightarrow iterate through the ^{two} DP list.

If $A[i-1] == B[j-1]$:

$$DP[i][j] = DP[i-1][j-1] + B[j-1]$$

\Downarrow
if equal, adding to the DP memo table

Else

$A[i-1]$ or $B[j-1]$ will have at least one that not in the LCS

$$DP[i][j] = \text{FIND-MAX}(DP[i-1][j], DP[i][j-1])$$

Return $DP[m][n]$.

(b) The final Result will be

$$LCS = \{ 2, 3, 4, 6, 5 \}$$

12.

(a). To prove that greedy is optimal for only nickel and pennies case, we will only be able to exchange for 4 pennies. since we will ~~be~~ ^{option} be able to use nickels until the total amount is ~~we~~ smaller than 5, ~~so~~ ~~so~~ therefore greedy provides the solution to reduce the chance of using small value if we still can use bigger one.

(b). for the case of dimes, nickels, and pennies, as our greedy algorithm, we will keep exchanging the biggest value first until we cannot use it, ~~which~~ ^{at most} which is dimes in this case. Therefore, I can only use ^{at most} 1 nickel here, since if the value is bigger than \$1, I can use dime ~~and~~ as many as until the remainder is smaller than one dime.

(c). for the case ~~where~~ where we can use quarter, dime, nickels, and pennies, as mentioned above, I can at most 2 dimes. otherwise I will be able to use quarter. ~~for example, the value of~~ if the remainder is above \$30, I will use quarter and nickels as many as I have. Then we will go from big ~~value~~ ^{case} to smaller ~~one~~.

13.

(a)

① First, we create a stack to track the vertex and DFS traverse the graph. ~~we~~ add a vertex into stack.

② Since this is a directed graph, we need to reverse the direction result after our ~~then~~ DFS traverse.

③ While stack is not empty.

we call DFS for each vertex from stack as the starting vertex,
and