

Problem 1.

(a)

$$C[i, j, k] = \begin{cases} 0 & , i=0 \text{ or } j=0 \text{ or } k=0 \\ C[i-1, j-1, k-1], & \text{if } W[i] = X[j] = Y[k] \\ \max\{C[i-1, j, k], C[i, j-1, k], C[i, j, k-1]\}, & \text{otherwise.} \end{cases}$$

(b)

case 1 (initial conditions): if $i=0$ or $j=0$ or $k=0$, there is no common sequence
 $\therefore C[i, j, k] = 0$

case 2: if $W[i] = X[j] = Y[k]$, match them.

the Longest Common Subsequence of these 3 sequence can contain them

\therefore longest common subsequence of $W[1 \dots i], X[1 \dots j], Y[1 \dots k]$ is one more greater than $W[1 \dots i-1], X[1 \dots j-1], Y[1 \dots k-1]$

\therefore so length $C[i, j, k] = C[i-1, j-1, k-1] + 1$

case 3: "otherwise" means if it is not the initial case and they don't match.

we need to find the maximum of Longest Common Sequence in $\{W[1 \dots i-1], X[1 \dots j], Y[1 \dots k]\}$

or $\{W[1 \dots i], X[1 \dots j-1], Y[1 \dots k]\}$ or $\{W[1 \dots i], X[1 \dots j], Y[1 \dots k-1]\}$

assign the maximum to $C[i, j, k]$.

$\therefore C[i, j, k] = \max\{C[i-1, j, k], C[i, j-1, k], C[i, j, k-1]\}$.

(c) Alg:

let $C[0 \dots p, 0 \dots m, 0 \dots n]$ be an new array of all 0. // initialization

for $i \leftarrow 1$ to p :

 for $j \leftarrow 1$ to m :

 for $k \leftarrow 1$ to n :

 if $W[i] = X[j] = Y[k]$:

// case for matching $W[i], X[j], Y[k]$

$C[i, j, k] = C[i-1, j-1, k-1] + 1$

 else

// case for not matching

$C[i, j, k] < \max\{C[i-1, j, k], C[i, j-1, k], C[i, j, k-1]\}$

(d) in side " $k=1 \dots n$ " loop, we use $O(1)$ time.

3 nested loop: $O(pmn)$ time.

total: $O(pmn)$ time.

Problem 2:

(a) Denote: $C[i, j]$ is the money we get or pay when we are at (i, j)

$$C[i, j] = \begin{cases} P[1, 1], & \text{if } i=j=1 \\ P[i, j] + C[i-1, j], & \text{if } i=1, j>1 \\ P[i, j] + C[i, j-1], & \text{if } j=1, i>1 \\ \max\{(C[i-1, j] + P[i, j]), (C[i, j-1] + P[i, j]), (C[i-1, j-1] + \frac{3}{2}P[i, j])\}, & \text{otherwise} \end{cases}$$

(b) The solution is a particular table entry. Every entry contains the maximum value when walking to the corresponding place.

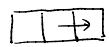
(c) Proof:

base case ($i=j=1$): when walking in $(1, 1)$, the maximum prize we get is just $P[1, 1]$

case for $i=1, j>1$: when we are at $(1, j)$ column, we can only move down, so every (i, j) is from $(i-1, j)$
 $\therefore C[i, j] = P[i, j] + C[i-1, j]$



case for $i>1, j=1$: when we are at $(i, 1)$ row, every (i, j) is from $(i, j-1)$



$$\therefore C[i, j] = P[i, j] + C[i, j-1]$$

case for "otherwise": as showed below, (i, j) is from $(i-1, j)$ or $(i, j-1)$ or $(i-1, j-1)$



$$\therefore C[i, j] = \max\{(C[i-1, j] + P[i, j]), (C[i, j-1] + P[i, j]), (C[i-1, j-1] + \frac{3}{2}P[i, j])\}$$

since we need to find the maximum value of $C[i, j]$.

from all above, we can get all maximum prize when we walk from $(1, 1)$ to (i, j)

so we will get maximum prize when walking from $(1, 1)$ to (m, n)

(d) Algorithm:

create $C[1 \dots m, 1 \dots n]$, all of elements are 0.

// initialization

$$C[1, 1] \leftarrow P[1, 1]$$

// when walking on $(\infty, 1)$ column

① for $i \leftarrow 2$ to m :

$$C[i, 1] \leftarrow C[i-1, 1] + P[i, 1]$$

// when walking on $(1, \infty)$ row.

② for $j \leftarrow 2$ to n :

$$C[1, j] \leftarrow C[1, j-1] + P[1, j]$$

// when walking on $(1, \infty)$ row.

③ for $i \leftarrow 2$ to m :

// find maximum at (i, j)

for $j \leftarrow 2$ to n :

$$C[i, j] \leftarrow \max\{(C[i-1, j] + P[i, j]), (C[i, j-1] + P[i, j]), (C[i-1, j-1] + \frac{3}{2}P[i, j])\}$$

(e) O(1) for initialization.

loop D use $O(m+1)$, loop E use $O(n-1)$, loop B use $O((m-1)(n-1))$

$$\begin{aligned}\therefore \text{in total: } & O(m+1) + O(n-1) + O((m-1)(n-1)) \\ &= O(m) + O(n) + O(mn - n - m + 1) \\ &= O(mn)\end{aligned}$$

\therefore my algorithm runs in $O(mn)$.

Problem 3 :

(a) if n is even:

(i) edges: $((i, 0), (i+1, 0))$, $i \in [1, \dots, n-2]$

$((n-1, i-1), (n-1, i))$, $i \in [1, \dots, n-1]$

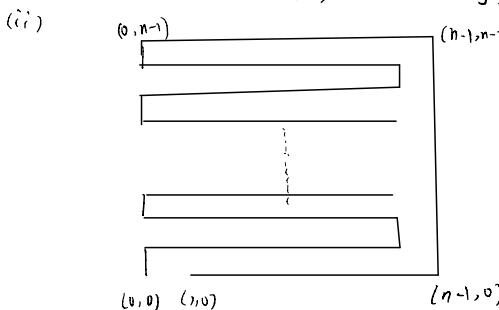
$((i, n-1), (i-1, n-1))$, $i \in [1, \dots, n-1]$

$((0, i), (0, i-1))$, $i \in [1, 3, 5, 7, \dots, n-1]$

$((n-2, i), (n-2, i-1))$, $i \in [2, 4, 6, \dots, n-2]$

$((i-1, j), (i, j))$, $i \in [1, \dots, n-2]$, $j \in [2, 4, 6, \dots, n-2]$

$((i, j), (i-1, j))$, $i \in [1, \dots, n-2]$, $j \in [1, 3, 5, \dots, n-3]$



if n is odd:

(i) edges: $((0, 1), (0, 0))$

$((i, 0), (i+1, 0))$, $i \in [1, \dots, n-2]$

$((n-1, i-1), (n-1, i))$, $i \in [1, \dots, n-1]$

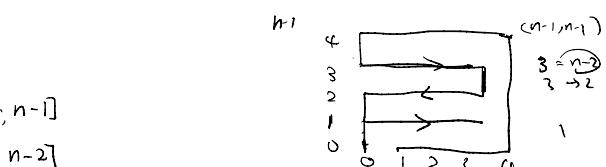
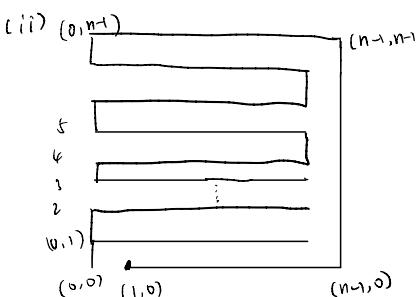
$((i, n-1), (i-1, n-1))$, $i \in [1, \dots, n-1]$

$((0, i), (0, i-1))$, $i \in [2, 4, 6, 8, \dots, n-1]$

$((n-2, i), (n-2, i-1))$, $i \in [3, 5, \dots, n-2]$

$((i-1, j), (i, j))$, $i \in [1, \dots, n-2]$, $j \in [1, 3, 5, \dots, n-2]$

$((i, j), (i-1, j))$, $i \in [1, \dots, n-2]$, $j \in [2, 4, 6, \dots, n-3]$



$0 \rightarrow 1$

$n-2 \rightarrow n-3$

$1 \rightarrow 0$

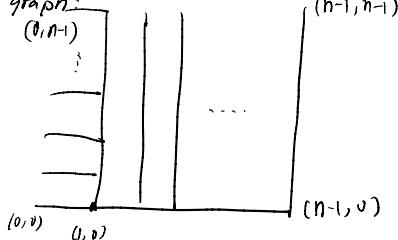
(b) edges:

$$((i,i), (0,i)) , \quad i \in [0 \dots n-1]$$

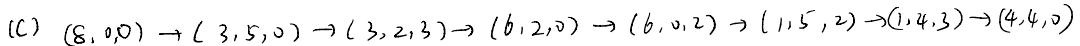
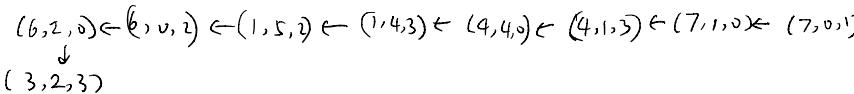
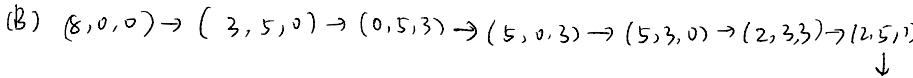
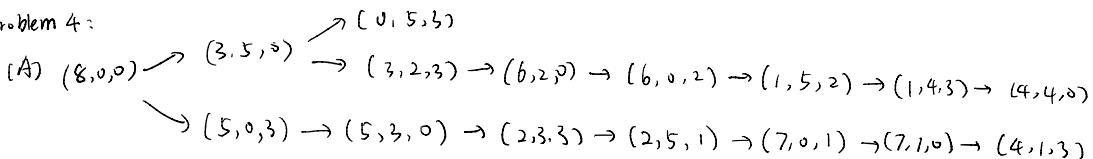
$$((i,0), (i+1,0)) , \quad i \in [0 \dots n-2]$$

$$((i,j), (i,j+1)) , \quad i \in [0 \dots n-1], \quad j \in [0 \dots n-2]$$

Gi) graph:



Problem 4:



Method : Do the BFS at v_0 , count the steps until we reach $(4, 4, 0)$

Once reach $(4, 4, 5)$, finish it, and we get the least number of pouring steps

(D) correctness: first, we know $(4, 4, 0)$ is reachable from $(8, 0, 0)$, from $(A, 0, 0)$.

Do the BFS at v_0 , assume there is an S , the set of nodes we have already visited

we always reach the adjacency of nodes in S in a round, put them into S .
 ∵ depth increases by one when visiting the adjacency of S . we count the depth

So once we reach $(4,4,0)$, we can get the depth of $(4,4,0)$, whose root is $(8,0,0)$

\therefore We get the shortest path from $(8, 0, 0)$ to $(4, 4, 5)$

∴ the Method of Using BFS to find least number of pouring steps is correct

(F) there is no sequence from $\{8, 00\}$ to $\{6, 1, 1\}$

(F) From (A) or (B), we know there is no path from $(8, 0, 0)$ to $(6, 1, 1)$, i.e. $(6, 1, 1)$ is not reachable from $(8, 0, 0)$, so we can't get 6 Liters in first jug and 1 liter in second and third jug.

(G) Proof: $\forall a, b, c, a \in [0, 8], b \in [0, 5], c \in [0, 3], ab + bc = 8$, exist a path such that

$(a,b,c) \rightarrow (a+c, b, 0) \rightarrow (a+b+c, 0, 0)$, so every node v in $V - \{w\}$ can reach $v_0(8,0,0)$

\therefore for all $v \in V - \{v_0\}$, if v is reachable from v_0 then v_0 is reachable from v .

(H) the statement is wrong:

from proof in (g) we know there exists a path for $(6,1,1)$ to $(8,0,0)$

but $(6, 1, 1)$ is not reachable from $(8, 0, 0)$

which is a contradictory for the statement

Problem 5:

i	1	2	3	4	5	6	7	8
a_i	A	B	C	D	E	F	G	H
$f(a_i)$	5	15	5	5	10	10	20	5

a) Fill in the two tables below. As in the example powerpoint only the entries with $i \leq j$ need to be filled in. We have started you off by filling in the $[i, i]$ entries.

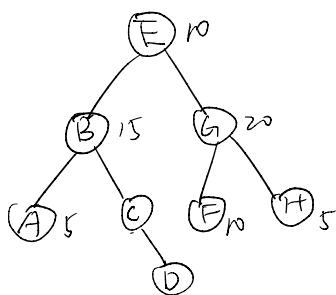
i/j	1	2	3	4	5	6	7	8
1	5	25	35	50	80	110	160	175
2	0	15	25	40	70	95	145	160
3	0	0	5	15	35	55	105	115
4	0	0	0	5	20	40	85	95
5	0	0	0	0	10	30	70	80
6	0	0	0	0	0	10	40	50
7	0	0	0	0	0	0	20	30
8	0	0	0	0	0	0	0	5

Table 1: Left matrix is $e[i, j]$.

i/j	1	2	3	4	5	6	7	8
1	1	2	2	2	2	2	5	5
2	0	2	2	2	2	5	5	5
3	0	0	3	3	4	5	5	7
4	0	0	0	4	5	5	6	7
5	0	0	0	0	5	5	6	7
6	0	0	0	0	0	6	7	7
7	0	0	0	0	0	0	7	7
8	0	0	0	0	0	0	0	8

Right matrix is $root[i, j]$.

(b)



$$\begin{aligned}
 \therefore \text{Cost} &= 10 \times 1 + (15+20) \times 2 + (5+5+10+5) \times 3 \\
 &\quad + 5 \times 4 \\
 &= 175
 \end{aligned}$$