M16

Q 1, 11, 12, 13

Q 9, 4, 2, 8

Q1
10 persons

A(1)\B(9)	Y(>=1 report) 1-(1-q)^9	N (1-q)^9
Y (p)	5-3	5-3
N (1-p)	5	0

(p=q)

Expected reward of A: p*2+(1-p)*5* (1-(1-q)^9)

dA/dp = 2-5(1-(1-q)^9) = -3 +5*(1-q)^9 = 0 =>. (1-q)^9 = 3/5

(1-q)^10 = (3/5)^(10/9)

Q11

-7. (-4, -3, -2, 8, 10)

$x>-1$ (-4+3)

Q12

+ (+2)	- (+1)	+(+2)
- (-1)	- (-4)	- (+1)
- (-2)	- (-1)	+ (+2)

Q13

- 19 R F

N firm to R

From N firm who pollute the river: 15*N → 10*(N-1)+60

From 19-N firm who build filter: 10*N+60 → 15*(N+1)

15*N <= 10*(N-1)+60.   5N <= 50

10*N+60 <= 15*(N+1).   45 <= 5N

N = 9 or N = 10

Q9

A \ B	I	II	III	IV
I	3, -10	11, -8	7, -7	6, -5
II	7, -3	13, -3	13, -6	8, -2
III	0, -9	7, -4	5, -11	3, -8
IV	8, -1	11, -6	12, -9	5, 2

For player A, action 2 is strictly better than action 1
For player B, action 4 is strictly better than action 3

Q4

Row \ Col	L	R
U (p)	9,5	9,0
D (1-p)	9,0	0,10

If the C player choose L, Row do not care about p

If the C player choose R → Row will have p = 1 → C player choose L

When will C player choose L?

Expect reward for C(C choose L with 100%) = p*5

Expect reward for C(C choose R with 100%) = (1-p)*10

R(C choose L)>= R(C choose R)

P*5 >= (1-P)*10

P>=2/3

P<=1

Q2

- 266
- Long (266-N): 1.   1 → (N+1)/19
- Direct (N): n/19.   N/19 → 1

- 1<= (N+1)/19   18<=N
- N/19 <= 1.   M<=19

- N =18 or N=19
- 266 - N

Q8

Romeo \ Juliet	Bach (q)	Stravinsky (1-q)
Bach (p)	6, 3 (pq)	0, 0 (p(1-q))
Stravinsky (1-p)	0, 0. ((1-p)q)	3, 6 ((1-p)(1-q))

First P
Second Q

First cannot find better P conditioned on Q
Second cannot find better Q conditioned on P

Expected reward for R player: p*q*6 + (1-p)*(1-q)*3
Expected reward for C player: p*q*3 + (1-p)*(1-q)*6

dR’s Reward(p)/dp = q*6-(1-q)*3 = 0. q=1/3
dC’s Reward(q)/dq = p*3-(1-p)*6 = 0. p=2/3

2/9*6 + 2/9*3 = 2

w*x+b = [w,b]*[x,1]. Who wio

X→ x*w1+b1 =[x,1] * wih → sigmoid → hidden (28)

hidden (28) → hidden*w2+b2 = [hiddent,1]*who → value [-inf,+inf]

(value - => 0; value + =>1) →sigmoid→ pre\in [0,1] y{0,1} (value<0.5 =>0, value>0.5 =>1)

Loss = Loss(pre, y) = crossentropy

Gradient d loss/ d w1

D loss/ d b1

Sigmoid(x) = return 1/(1+e^x)

Sigmoid ->function return [0,1] -> 0,1

[-inf, +inf]