23K-0004
BAI-4A

AI Assignment # 02

Date:

Q2

Step 1: Population Initialization

C_1 =	1, 3, 1, 2, 3, 2, 1
C_2 =	3, 2, 2, 1, 1, 3, 2
C_3 =	3, 3, 2, 2, 1, 1, 3
C_4 =	1, 1, 2, 3, 1, 2, 2
C_5 =	1, 3, 3, 1, 1, 1, 2
C_6 =	2, 2, 3, 2, 1, 1, 3

Step 2: Evaluate fitness

								Loads		
Chromosome	Task1	Task2	Task3	Task4	Task5	Task6	Task7	F1	F2	F3
C1	5x10=50	8x16=128	4x8=32	7x10=70	6x12=72	3x8=24	9x11=99	5+4+9=18	7+3=10	8+6=14
C2	5x9=45	8x14=112	4x9=36	7x12=84	6x14=84	3x10=30	9x12=108	7+6=13	8+4+9=21	5+3=8
C3	5x9=45	8x16=128	4x9=36	7x10=70	6x14=84	3x9=27	9x13=117	6+3=9	4+7=11	5+8+9=22
C4	5x10=50	8x15=120	4x9=36	7x13=91	6x14=84	3x8=24	9x12=108	5+8+6=19	4+3+9=16	7
C5	5x10=50	8x16=128	4x7=28	7x12=84	6x14=84	3x9=27	9x12=108	5+7+6+3=21	9	8+4=12
C6	5x12=60	8x14=112	4x7=28	7x10=70	6x14=84	3x9=27	9x13=117	6+3=9	5+8+7=20	4+9=13

Fitness:

C1 = 50+128+32+70+72+24+99 => 475

C2 = 45+112+36+84+84+30+108 => 499

C3 = 45+128+36+70+84+27+117 => 507

C4 = 50+120+36+91+84+24+108 => 513

C5 = 50+128+28+84+84+27+108 => 509

C6 = 60+112+28+70+84+27+117 => 498

Fitness = [475, 499, 507, 513, 509, 498]

Step 3: Roulette Wheel Selection

C	Inverse	Probability	
C1	1/475 => 0.002105	P(C1) = 0.002105/0.012003 => 0.1753	
C2	1/499 => 0.002004	P(C2) = 0.002004/0.012003 => 0.1669	C1 and C6 favored
C3	1/507 => 0.001972	P(C3) = 0.001972/0.012003 => 0.1643	because of lower costs
C4	1/513 => 0.001949	P(C4) = 0.001949/0.012003 => 0.1624	
C5	1/509 => 0.001965	P(C5) = 0.001965/0.012003 => 0.1637	
C6	1/498 => 0.002008	P(C6) = 0.002008/0.012003 => 0.1673	

Total = 0.012003

23K-0004
BAI-4A

Date:

Step 4: Crossover (single-point)

Pair1: C_1 = [1,3,1,2,3,2,1]
C_5 = [2,2,3,2,1,1,3]

Pair2: C_1 = [1,3,1,2,3,2,1]
C_2 = [3,2,2,1,1,3,2]

Pair3: C_2 = [3,2,2,1,1,3,2]
C_6 = [2,2,3,2,1,1,3]

Cross-over at '3':
O1 = [1,3,1,2,1,1,3]
O2 = [2,2,3,2,3,2,1]

Cross-over at '2':
O3 = [1,3,2,1,1,3,2]
O4 = [3,2,1,2,3,2,1]

Crossover at '4':
O5 = [3,2,2,2,1,1,3]
O6 = [2,2,3,2,1,3,2]

Step 5: Mutation (20% chance)

O1 = [1,3,1,2,1,1,3] → swapping position ~~8~~ 1 and 5 => [1,~~8~~1,1,2,1,3,3]
O2 = [2,2,3,2,3,2,1]
O3 = [1,3,2,1,1,3,2]
O4 = [3,2,1,2,3,2,1] → swapping position 4 and 5 => [3,2,1,2,2,3,1]
O5 = [3,2,2,1,1,1,3]
O6 = [2,2,3,2,1,3,2]

Step 6: Fitness of new population.

								Loads		
Offspring	Task1	Task2	Task3	Task4	Task5	Task6	Task7	F1	F2	F3
O1	5x10=50	8x15=120	4x8=32	7x10=70	6x14=84	3x10=30	9x13=117	5+8+4+6=23	7	3+9=12
O2	5x12=60	8x14=112	4x7=28	7x10=70	6x12=72	3x8=24	9x11=99	9	5+8+7+3=23	4+6=10
O3	5x10=50	8x16=128	4x9=36	7x12=84	6x14=84	3x10=30	9x12=108	5+7+6=18	4+9=13	8+3=11
O4	5x9=45	8x14=112	4x8=32	7x10=70	6x13=78	3x10=30	9x11=99	4+9=13	8+7+6=21	5+3=8
O5	5x9=45	8x14=112	4x9=36	7x12=84	6x14=84	3x9=27	9x13=117	7+6+3=16	8+4=12	5+9=14
O6	5x12=60	8x14=112	4x7=28	7x10=70	6x14=84	3x10=30	9x12=108	6	5+8+7+9=29	4+3=7

Fitness:

O1 = 50+120+32+70+84+30+117 => 503

O2 = 60+112+28+70+72+24+99 => 465

O3 = 50+128+36+84+84+30+108 => 520

O4 = 45+112+32+70+78+30+99 => 466

O5 = 45+112+36+84+84+27+117 => 505

O6 = 60+112+28+70+84+30+108 => 492

23K-0004
BAI-9A

Date:

Q3.

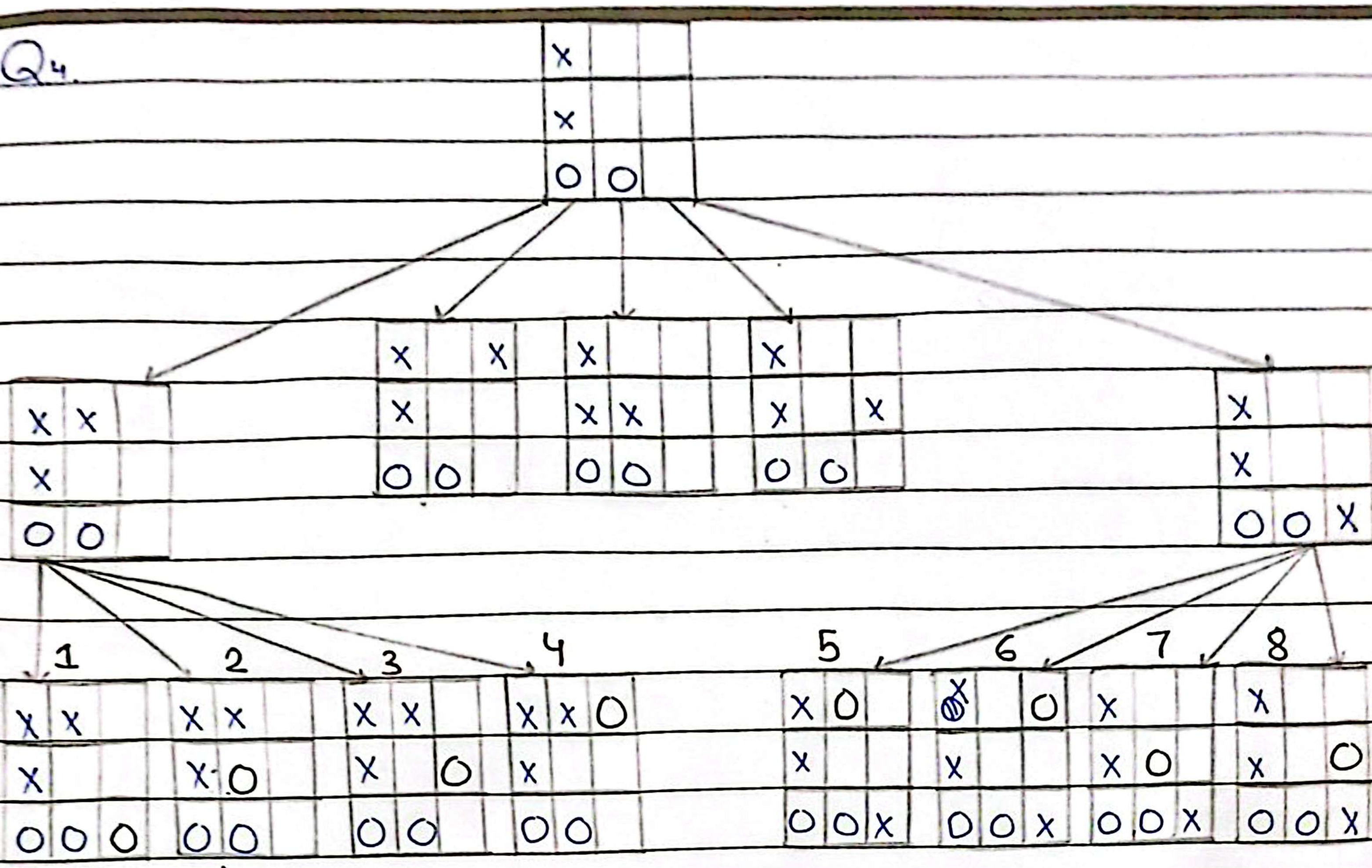
My version of sudoku solver is different from Google OR tools solver because it uses pure python with no external libraries to ensure optimization. Moreover, it lacks heuristics like MRV unlike chatGPT and OR-Tools.

By adding heuristics (MRV), forward checking and reducing arc recalculation, the custom version can improve its speed significantly

23K-0004
BAI-4A

Date: ____________

Q4.

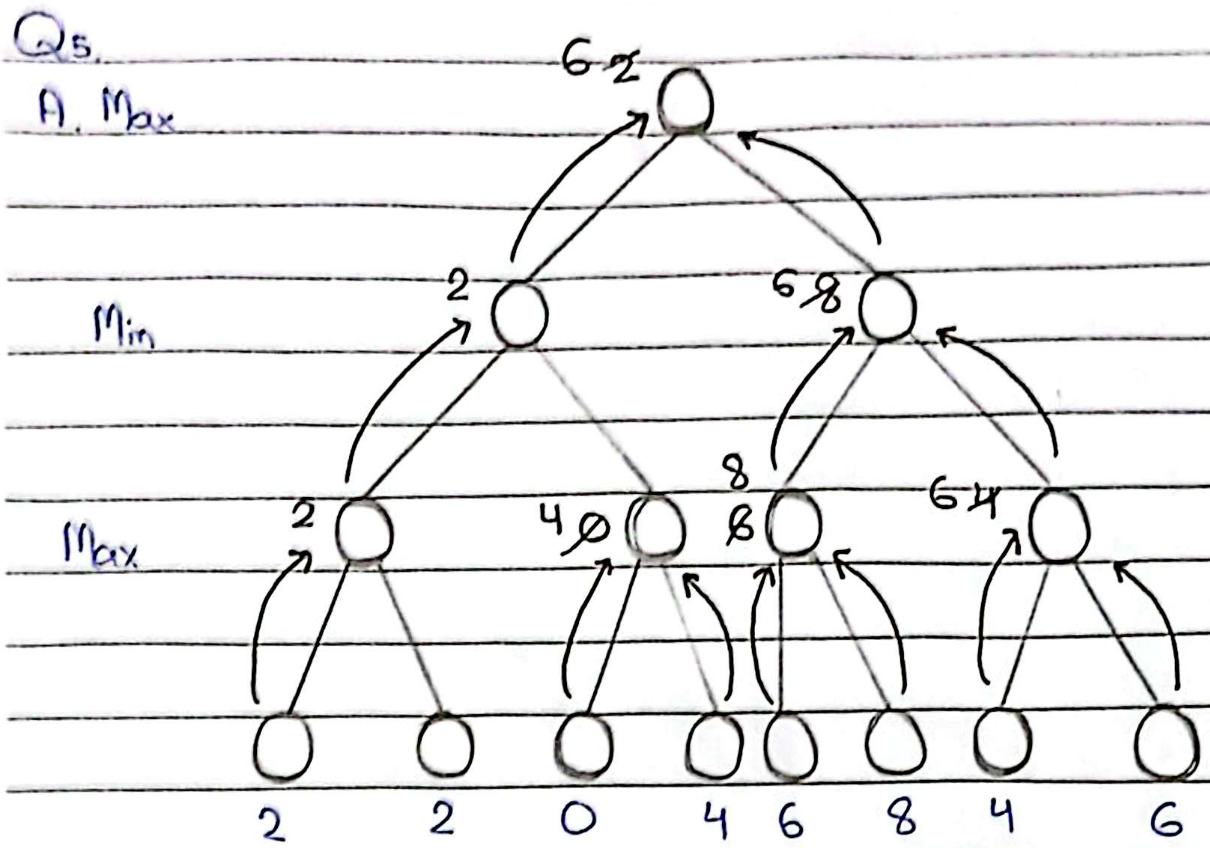


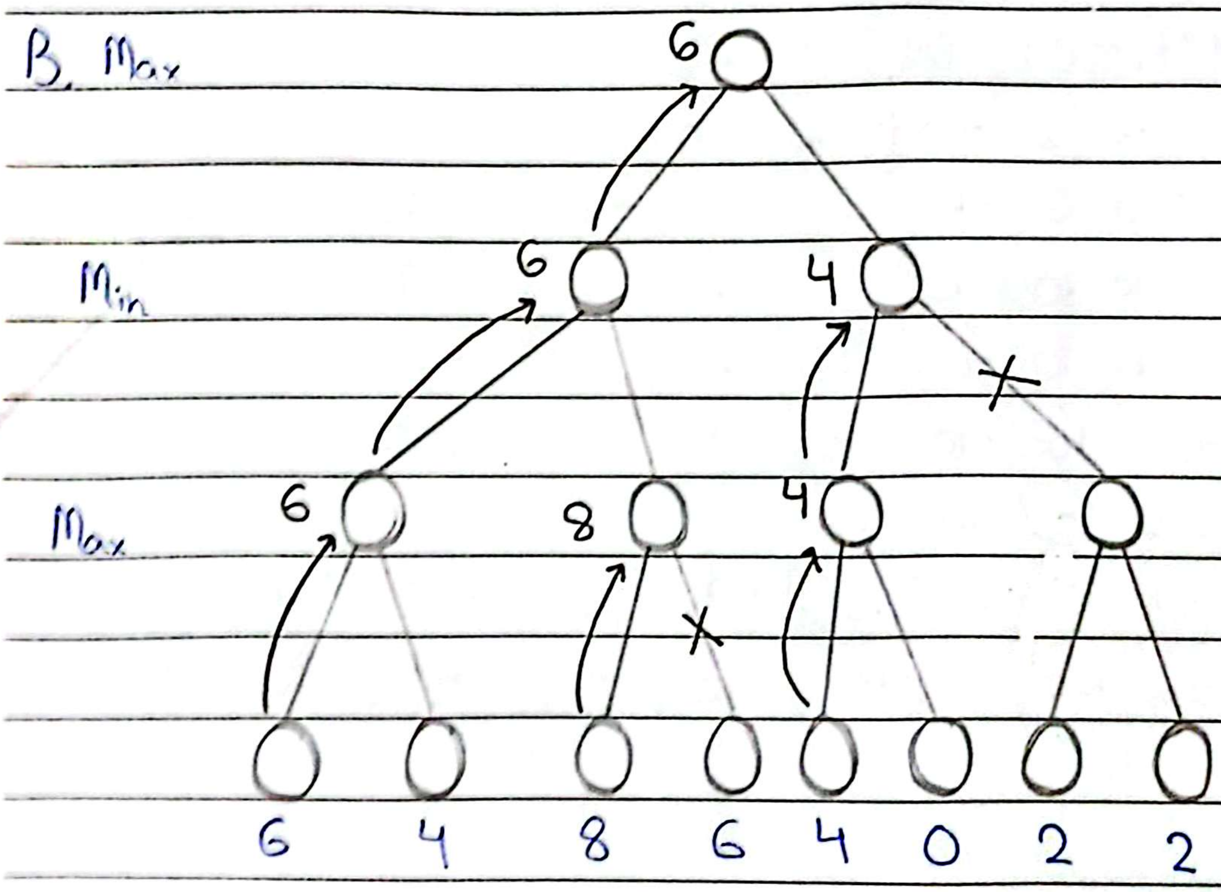
States	R1	R2	R3	C1	C2	C3	D1 TL→BR	D2 TR→BL	Sum R	Sum C	Sum D	V=SUM(R,C,D)
State 1	100	10	-1000	0	0	-10	0	-10	-890	-10	-10	-910
State 2	100	0	-100	0	0	0	0	-100	0	0	-100	-100
State 3	100	0	-100	0	0	-10	10	-10	0	-10	0	-10
State 4	0	10	-100	0	0	-10	10	-100	-90	-10	-90	-190
State 5	0	+10	0	0	-100	10	100	-10	+10	-90	90	+10
State 6	0	10	0	0	-10	0	100	-100	10	-10	0	0
State 7	10	0	0	0	-100	10	0	-100	10	-90	-100	-180
State 8	10	0	0	0	-10	0	100	-10	10	-10	90	90

23K-0004
BAI-4A

Date: ______

Q5.





23K-0004
BAI-4A

Date:

Q6.

a).

1. Players:

Max(Defender): AI-powered IDS which will defend the network from external attacks.

Min(Attacker): It's goal is to breach the network using various attacks.

2. Decision-Making:

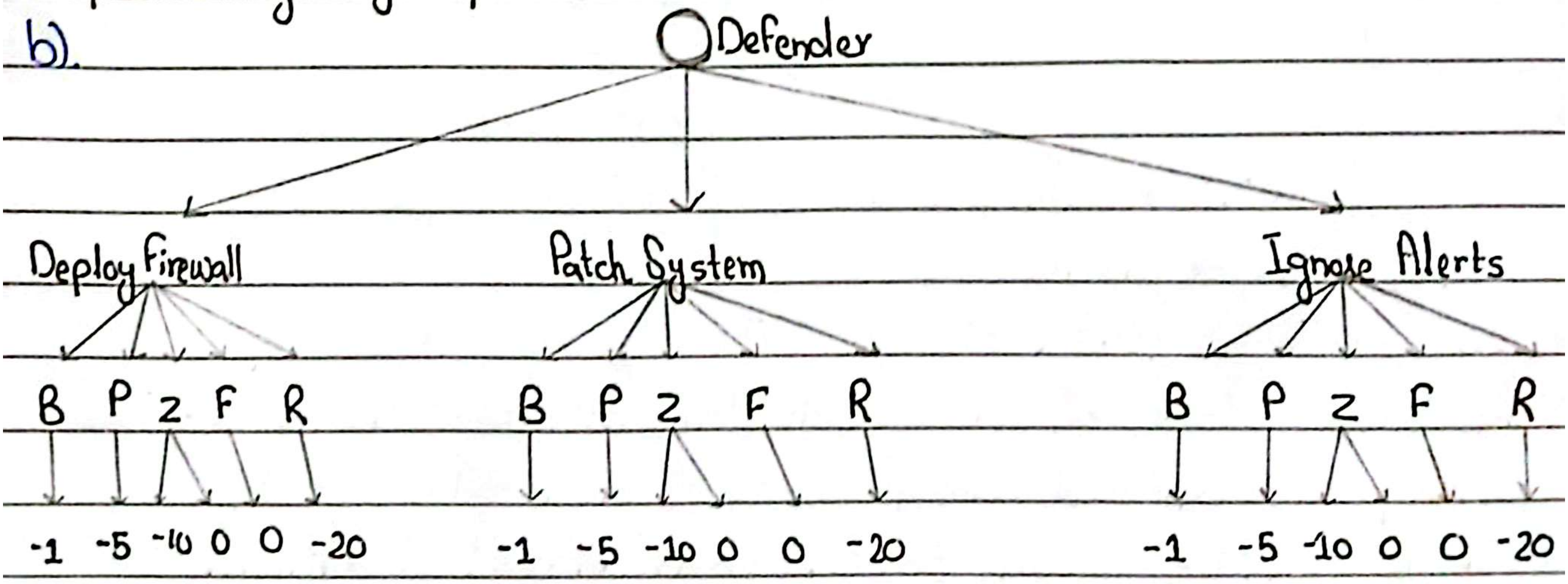
Max(Defender): Uses strategies like deploying firewalls, patching systems or ignoring alerts to minimize the damage caused while maintaining costs.

Min(Attacker): Uses attacks like Brute-force, Phishing, Zero-Day Exploit, fake and real attacks to maximize the damage caused to the network.

3. Stochastic Elements:

Attacks like Zero-Day exploit are probabilistic with 50% success rate. They introduce uncertainty and the defender may need to shift its focus from worst-case to average case based on probability (e.g Expectimax).

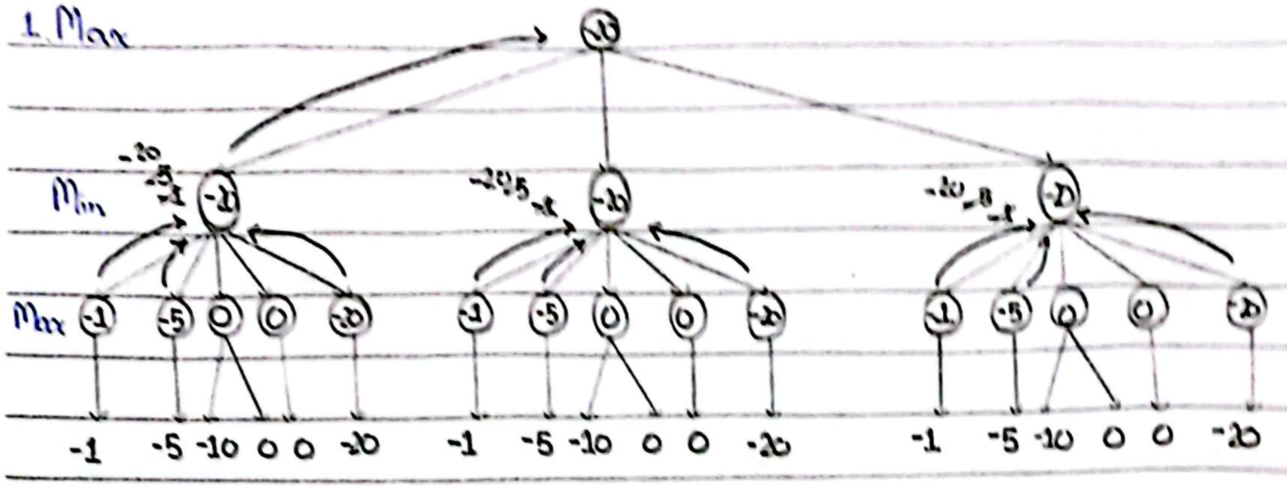
b).

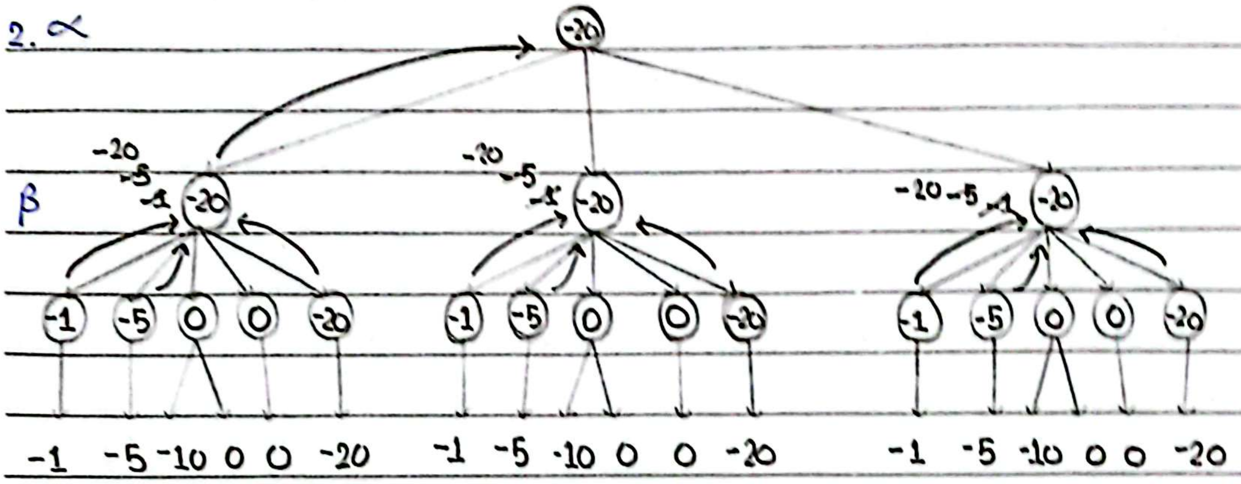


23K-0004
BAI-4A

Date:

c).





d).

1. success (50%) → damage = -10
   fail (50%) → damage = 0
   Expected Value = $[0.5 \times (-10)] + [0.5 \times (0)] \Rightarrow -5$

This means that, on average, if the attacker chooses zero-day exploit, the expected damage to the system will be -5.

2. If the defender switches to expectimax instead of minimax, it doesn't always assume the worst case, ~~instead~~ instead it also takes into account the probabilities of attack successes therefore, it may choose defenses that have lower expected damage, ~~an~~ even if the worst case deals high damage.