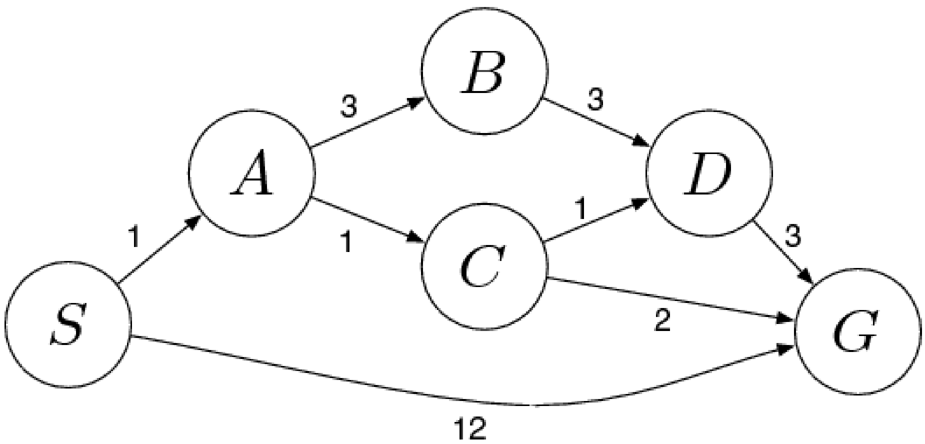
Ques 1. Define AI and explain in detail all FOUR approaches of AI with diagrams/pictorial representations? [5]

Systems that think like humans: 1. "The exciting new effort to make computers think. *Machines with minds*, in the full and literal sense." (Haugeland, 1985) 2. The automation of activities that we associate with human thinking tasks such as decision-making, learning, problem solving, etc. (Bellman 1978)	Systems that think rationally: 1. The study of mental faculties (power) through the use of computational models. (Charmaik and McDermott, 1985) 2. The study of computations that make it possible to perceive, reason, and act." (Winston, 1992)
Systems that act like humans **1. "The art of creating machines that perform functions that require intelligence when performed by people." (Kurzwell, 1990).** **2. "The study of how to make computers do things at which, at the moment, people do better." (Rich and Knight, 1991)**	Systems that act rationally: 1. "Computational Intelligence is the study of design of intelligent agents." (Poole et al., 1998) 2. AI is concerned with intelligent behaviour in the artifacts (articles /products created by human) (Nilsson, 1998)

1 Mark for AI Definition and 1 Mark for each of the FOUR approaches

Q.2. Consider the state space search as shown in Fig.1, with initial state S and goal state G. Using UCS (Uniform cost search) algorithm, find the optimal path from S to G. Disclose the OPEN and CLOSED list maintained by the UCS algorithm at each level. Maintain each node in OPEN and CLOSED as (node, parent node, g(n)). Also, provide the time and space complexity of UCS algorithm. [5]

(Iteration 1 and 2: 0.5 each + iteration 3 and 4: 1 marks each + iteration 5: 0.5+ path and path cost: 0.5 , Time and Space complexity: 0.5 each)



Initially g(S)=0

Add (S,f,0) to OPEN and CLOSED is empty

Iteration	Open	Close
0	{(S, Null, 0)}	{}

Iteration I

Remove head node (S,f,0) from OPEN and add to CLOSED. Since S is not goal node, therefore successors of S i.e.A and G are produced (case 1: both are new not in OPEN and CLOSED)

Successors of S:

A: g(A)=g(S)+cost(S,A) = 0+1 =1

G : g(G)=g(S)+cost(S,G) = 0+12 =12

Iteration	Open	Close
0	{(S,f,0)	{}
1	{(A, S, 1), (G, S, 12)}	{(S,f,0)

Iteration II

Remove head node (A,S,1) from OPEN and add to CLOSED. Since A is not goal node, therefore successors of A i.e. B and C are produced (case 1: both are new not in OPEN and CLOSED)

Successors of A:

B: g(B)=g(A)+cost(A,B) = 1+3 =4

C : g(C)=g(A)+cost(A,C) = 1+1 =2

Iteration	Open	Close
0	{(S,f,0)	{}
1	{(A, S, 1), (G, S, 12)}	{(S,f,0)
2	{(C, A, 2), (B, A, 4), (G, S, 12)}	{(S,f,0, (A, S, 1))

Iteration III

Remove head node (C,A,2) from OPEN and add to CLOSED. Since C is not goal node, therefore successors of C i.e. D and G are produced (case 1: D is new not in OPEN and CLOSED) (Case 2: G is already in OPEN)

Successors of C:

D: g(D)=g(C)+cost (C, D) = 2+1 =3

G: newg(G)=g(C)+cost(C,G) = 2+2 =4 since newg(G) is less than g(G), so update (G, C, 4)

[node generated before has higher priority in case of same g values]

Iteration	Open	Close
0	{(S,f,0)	{}
1	{(A, S, 1), (G, S, 12)}	{(S,f,0)

2	{(C, A, 2), (B, A, 4), (G, S, 12)}	{(S,f,0, (A, S, 1))
3	{(D, C, 3)(G, C, 4), B, A, 4)}	{(S,f,0, (A, S, 1), (C, A, 2))

Iteration IV

Remove head node (D,C,3) from OPEN and add to CLOSED. Since D is not goal node, therefore successors of D i.e. G is produced (Case 2: G is already in OPEN)

G: newg(G)=g(D)+cost(D,G) = 3+3 =6 since newg(G) is not less than g(G), so ignore this successor

Iteration	Open	Close
0	{(S,f,0)	{}
1	{(A, S, 1), (G, S, 12)}	{(S,f,0)
2	{(C, A, 2), (B, A, 4), (G, S, 12)}	{(S,f,0, (A, S, 1))
3	((D, C, 3) (G, C, 4), (B, A, 4)}	{(S,f,0, (A, S, 1), (C, A, 2))
4	{(G, C, 4), (B, A, 4)}	{(S,f,0, (A, S, 1), (C, A, 2) (D, C, 3))

Iteration V:

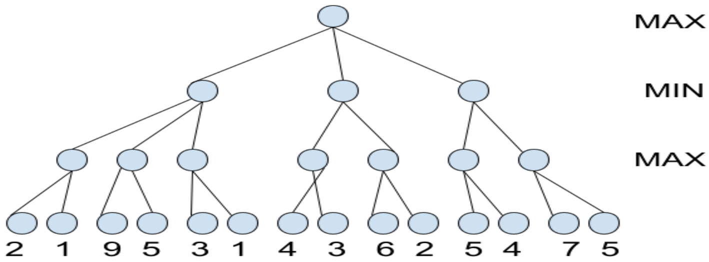
Remove head node (G,C,4) from OPEN and add to CLOSED. Since G is goal node, therefore algorithm will terminate.

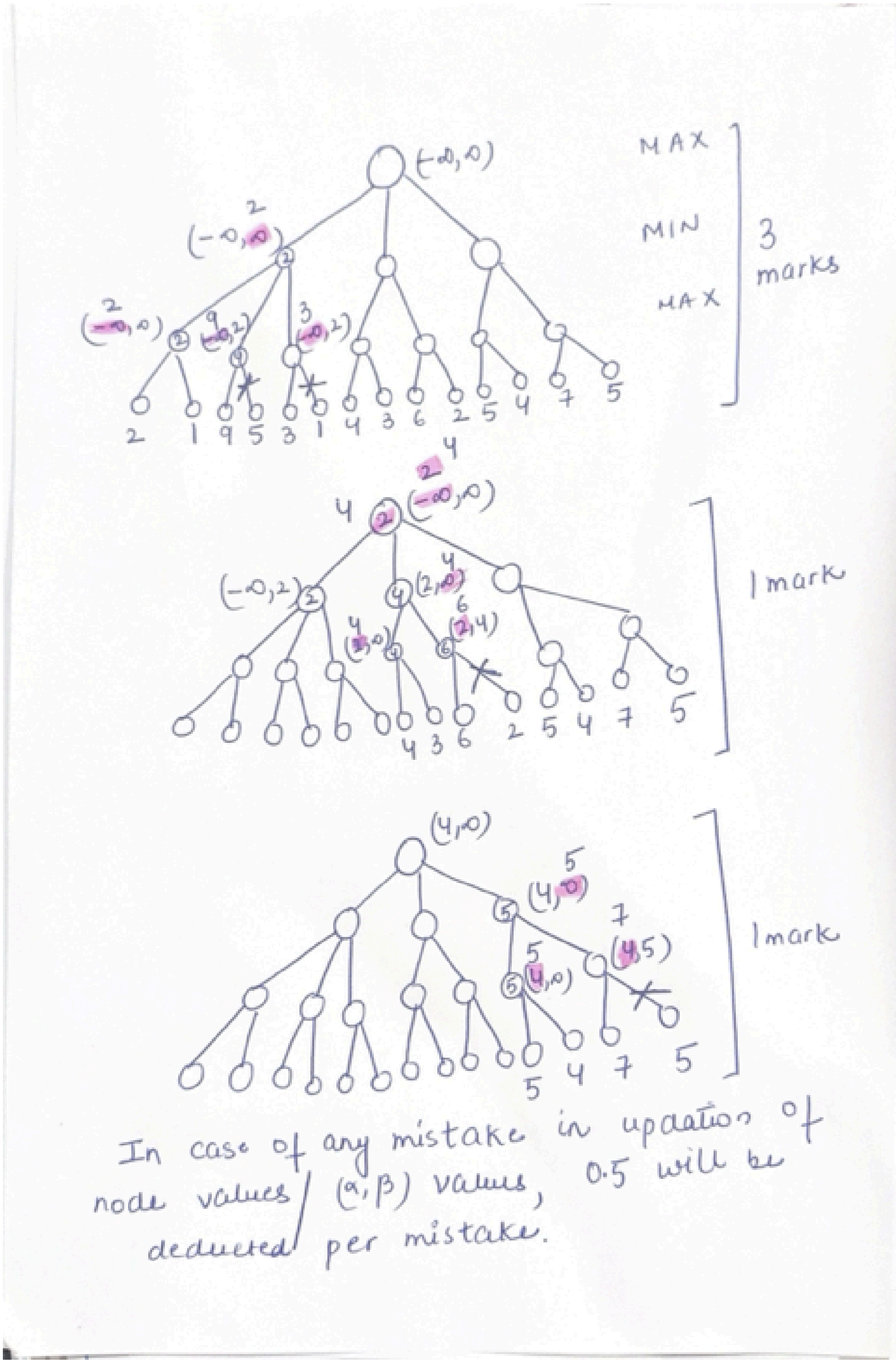
Path isS-> A-> C-> G with path cost 4

Iteration	**Open**	**Close**
0	**{(S,f,0)**	**{}**
1	**{(A, S, 1), (G, S, 12)}**	**{(S,f,0)**
2	**{(C, A, 2), (B, A, 4), (G, S, 12)}**	**{(S,f,0, (A, S, 1))**
3	**((D, C, 3) (G, C, 4), (B, A, 4)}**	**{(S,f,0, (A, S, 1), (C, A, 2))**
4	**{(G, C, 4), (B, A, 4)}**	**{(S,f,0, (A, S, 1), (C, A, 2) (D, C, 3))**
5	**{(B, A, 4)}**	**{(S,f,0, (A, S, 1), (C, A, 2) (D, C, 3), (G, C, 4))**

Time Complexity: The time complexity of UCS is of the order of O(b c /m) where c is the cost of the optimal path and m is the minimum edge length.
Space Complexity: The space complexity of UCS is same as that of BFS i.e. O(b^(d+1))

Q3. Which nodes in the following state space search graph will never be explored in the minimax algorithm after the application of alpha-beta pruning? [5]


MAX
MIN
MAX
2 1 9 5 3 1 4 3 6 2 5 4 7 5



Q4. (a). Specify task environment for Refinery Controller agent using PEAS parameters. [2]

(b). How Goal-Based agent is different from Utility-Based agent? In which agent does the problem generator is present? [3]

Ans 4(a):

1. Performance measure: Purity, Yield, Safety

2. Environment: Refinery, Operators
3. Actuators: Valves, Pumps, Heaters, Displays
4. Sensors: Temperature, Pressure, Chemical Sensors

Ans 4(b):

Goal-Based Agents	Utility-Based Agents
Goal-based agents may perform in a way that produces an unexpected outcome because their search space is limited	Utility-based agents are more reliable because they can learn from their environment and perform most efficiently
Makes decisions based on the goal and the available information	Makes decisions based on the utility and general information
Goal-based agents are easier to program	Implementing utility-based agents can be a complex task
Considers a set of possible actions before deciding whether the goal is achieved or not	Maps each state to an actual number to check how efficiently each step achieves its goals
Utilized in computer vision, robotics, and NLP	Used in GPS and tracking systems

The problem generator is present in **learning agent**.

4a - 0.5 mark for each of four PEAS parameters

4b - 1 mark for each of TWO differences + 1 mark for writing correct answer of second subpart (i.e learning agent)

Q5. Let a function f(x) = x-(x2/16) be defined on the initial population as [8, 10, 12, 13] while taking 5-bit binary representation. Apply a genetic algorithm to determine the maximum number of given functions while considering the population size of 4. Suggest an evaluation order for the next iteration and populate the next iteration with only the two best chromosomes from the first iteration. 1st iteration:

Further, perform the following operations from the

results of the 1st iteration of chromosomes: (1mark)

a) Single-point crossover after the 3rd gene for two fittest chromosomes.(1mark)

b) Mutation of the last gene for the worst two chromosomes.(1mark)

c) Calculate the fitness of the new population generated using (a) and (b) and find the best fit chromosome.

(2 mark) [2nd iteration and it's fitness(1) and best chromosome value(1)]

String No	Initial Pop^n	X	f(x)
1.	8	01000	4
2.	10	01010	3.75
3.	12	01100	3
4	13	01101	2.43

Evaluation Order [8, 10, 12, 13]

a) 010|00 ⇒ 01010
010|10 ⇒ 01000

b) 01100 ⇒ 01101
01101 ⇒ 01100

c)

String No.	Initial Pop^n	f(x)
1.	01010 (10)	3.75
2.	01000 (8)	4
3.	01101 (13)	2.43
4.	01100 (12)	3

Best chromsome ⇒ 8 ⇒ 01000

Q6. Suppose you want to solve Sudoku game with the help of Simulated Annealing Algorithm. Below is some random state given to you. [5]

	Rows (starting from index 1)								
Columns (starting from index 1)	5	2	4	4	3	7	6	8	9
	6	7	9	1	5	9	2	7	3
	8	1	3	6	8	2	4	1	5
	4	3	1	6	7	5	4	8	1
	5	8	6	1	8	9	9	3	2
	7	9	2	2	3	4	7	6	5
	2	8	9	7	1	5	8	4	2
	6	4	5	2	9	3	9	1	3
	3	1	7	8	6	6	7	5	6

Objective function of any state (n) is O(n) defined by sum of number of duplicates in each row and column. (Minimization problem)

Example: Value of

4	4	2
5	4	2

Objective function of this example is 3

Initial Temperature (T) = 10; Change in Temperature after each iteration (ΔT) = 2

Random value for iteration 1 is 0.3; Random value for Iteration 2 is 0.4

A Random new state for 1st iteration is created by swapping ([Row][Column]) [2][2] with [2][3] (i.e., 7 with 9); A Random new state for 2nd iteration is created by swapping [3][9] with [5][7]. With the information given above solve first two iterations using simulated annealing algorithm and answer the following (Show all intermediate steps):

a) Objective function value for initial state and random states (for both iterations)
b) Probability values for each iteration
c) Whether new states at each iteration will be accepted or rejected

Solution No marks for accepted/rejected state if probability value is not correct.

	Rows (starting from index 1) →									
Columns (starting from index 1) ↓	5	2	4	4	3	7	6	8	9	1
	6	7	9	1	5	9	2	7	3	1+1
	8	1	3	6	8	2	4	1	5	1+1
	4	3	1	6	7	5	4	8	1	1+1
	5	8	6	1	8	9	9	3	2	1+1
	7	9	2	2	3	4	7	6	5	1+1
	2	8	9	7	1	5	8	4	2	1+1
	6	4	5	2	9	3	9	1	3	1+1
	3	1	7	8	6	6	7	5	6	1+1
	1+1	1+1	1	1+1+1	1+1	1+1 +1	1+1 +1	1+ 1+	1+1 +1	

$O(n) = 37$

1 Mark

$O(n) = 38$

$\Delta E = 38 - 37 = 1$

1st iteration

	Rows (starting from index 1) →									
Columns (starting from index 1) ↓	5	2	4	4	3	7	6	8	9	1
	6	9	7	1	5	9	2	7	3	1+1
	8	1	3	6	8	2	4	1	5	1+1
	4	3	1	6	7	5	4	8	1	1+1
	5	8	6	1	8	9	9	3	2	1+1
	7	9	2	2	3	4	7	6	5	1+1
	2	8	9	7	1	5	8	4	2	1+1
	6	4	5	2	9	3	9	1	3	1+1
	3	1	7	8	6	6	7	5	6	1+1
	1+1	1+1 +1	10	1+1+ 1	1+1	1+1	1+1 +1	1+1	1+1 +1	

$P = \frac{1}{1 + e^{\frac{\Delta E}{T}}}$

$= \frac{1}{1 + e^{\frac{1}{10}}}$

$= 0.47750$

1

P is greater than Random, so state accepted

0.5

4

New state

2nd iteration

	Rows (starting from index 1) →									
Columns (starting from index 1) ↓	5	2	4	4	3	7	6	8	9	1
	6	7	9	1	5	9	2	7	3	1+1
	8	1	3	6	8	2	4	1	9	1+1
	4	3	1	6	7	5	4	8	1	1+1
	5	8	6	1	8	9	5	3	2	1+1
	7	9	2	2	3	4	7	6	5	1+1
	2	8	9	7	1	5	8	4	2	1+1
	6	4	5	2	9	3	9	1	3	1+1
	3	1	7	8	6	6	7	5	6	1+1
	1+1	1+1 +1	1	1+1+ 1	1+1	1+1	1+1	1+1	1+1 +1	

$O(n) = 37$

$\Delta E = 37 - 38 = -1$

1

$P = \frac{1}{1 + e^{\frac{\Delta E}{T}}}$

$\boxed{\text{new } T = T - \Delta T = 10 - 2 = 8}$

$\frac{1}{1 + e^{-\frac{1}{8}}}$

$\frac{1}{1 + e^{-0.125}}$

1

$0.5312 = \cancel{0.5249}$

P is greater than Random so current state is accepted.

0.5

No marks for accepted/rejected states if the probability value is not correct