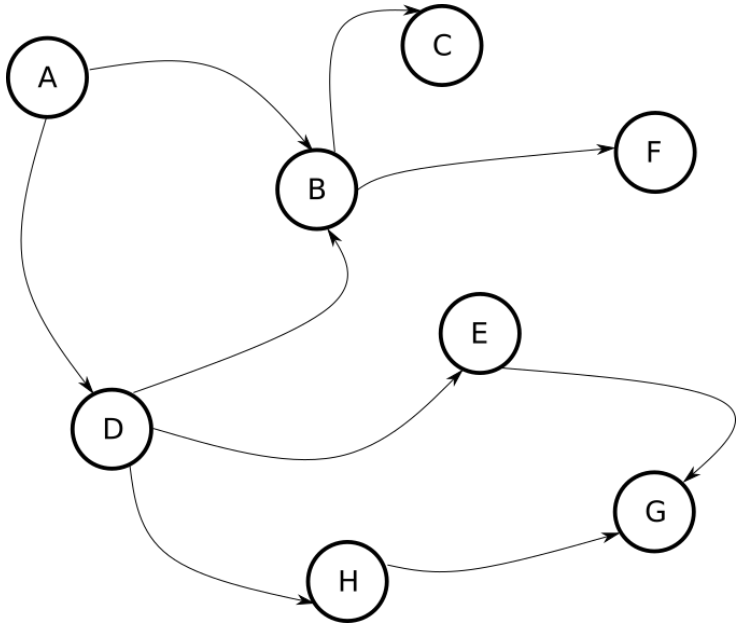
Model de subiect
(Examen)

1. Pentru un arbore minimax de adâncime maximă A generat pentru un joc oarecare, care dintre următoarele afirmații sunt adevărate?
   a. Un nod MIN poate fi rădăcină a arborelui minimax.
   b. Dacă aplicăm alpha-beta, nu putem să tăiem mai mult de jumătate dintre nodurile arborelui.
   c. Alpha-beta niciodată nu va elimina (reteza) un nod care este fiu direct al rădăcinii.
   d. Frunzele sunt întotdeauna noduri MAX.
   e. Frunzele sunt întotdeauna la distanță de exact A muchii față de rădacină.

2. Care dintre următoarele afirmații sunt adevărate?
   a. Algoritmul A* parcurge întotdeauna întreg spațiul stărilor în căutarea unei soluții.
   b. Algoritmul A* întoarce întotdeauna soluția de cost minim indiferent de euristică.
   c. Algoritmul A* este o tehnică de căutare neinformată.
   d. Algoritmul A* admite mai multe stări scop.
   e. Algoritmul A* poate fi implementat folosind o coadă de priorități pentru lista OPEN (lista nodurilor de expandat).

3. Se dă următoarea topologie de rețea Bayesiană:



Care dintre următoarele afirmații sunt adevărate?
   a. Mulțimea {E, H} d-separă mulțimea {A, B} de mulțimea {G}.
   b. Graful dat nu este o topologie corectă pentru o rețea Bayesiană.

c. **Dacă am adăuga arcul F->A graful nu ar mai fi o topologie corectă pentru o rețea Bayesiana.**
d. Drumul de la nodul A la nodul F e blocat conditionat de mulțimea {C,F}
e. Mulțimea {E,G,H} d-separă mulțimea {A} de mulțimea {D}

4. Se consideră următorul joc. Avem un grid cu NRLIN linii și 4 coloane în care în starea inițială primele două rânduri conțin simboluri x și ultimele două rânduri au simboluri 0 ca în configurația din imaginea de mai jos. Regulile de mutare sunt următoarele:

- jucătorul cu simbolul x mută primul
- jucătorii își pot muta simbolurile proprii cu o singura pozitie doar în direcția opusă configurației inițiale (x poate muta doar în jos pe coloană sau diagonală iar 0 poate muta doar în sus pe coloană sau diagonală). Jucătorii sunt obligați să facă o mutare. Dacă un jucător nu poate muta când îi vine rândul, atunci e remiză.
- în momentul în care un jucător face 3 simboluri pe linie, coloană sau diagonală poate muta un simbol al celuilalt jucător cu exact o poziție în spate, pe coloană sau diagonală
- scopul fiecărui jucător e să ajungă cu un simbol pe linia din capătul opus poziției sale de start (x să ajungă cu un simbol pe ultima linie și 0 cu un simbol pe prima linie).

Tabla inițială de joc arată așa pentru M=8

X	X	X	X
	X	X	
	0	0	
0	0	0	0

Care dintre variantele de mai jos oferă o funcție de evaluare care să arate în mod corect cât de favorabilă este starea jocului pentru calculator (MAX), cu alte cuvinte să aibă o valoare mai mare pentru stări mai favorabile și mai mică pentru stări mai nefavorabile?

a. Numărul de simboluri ale lui MAX din care scădem numărul de simboluri ale lui MIN.

b. Câte configurații de 3 simboluri are MAX din care scădem câte configurații de 3 simboluri are MIN.
c. Numerotarea liniilor și coloanelor începe de la 0 din colțul stânga sus. Calculatorul joaca cu X. Presupunem piesa P cu coordonatele(linie și coloană) LM și CM ca fiind cea mai apropiată piesă a jucătorului MAX de capătul în care trebuie să ajungă pentru a câștiga. Idem, avem piesa p a jucătorului MIN (cea mai apropiată de capatul câștigător al lui MIN) cu coordonatele Lm și Cm. O evaluare ar fi NRLIN-LM+Lm
d. Numerotarea liniilor și coloanelor începe de la 0 din colțul stânga sus. Calculatorul joaca cu X. Presupunem piesa P cu coordonatele(linie și coloană) LM și CM ca fiind cea mai apropiată piesă a jucătorului MAX de capătul în care trebuie să ajungă pentru a câștiga. Idem, avem piesa p a jucătorului MIN (cea mai apropiată de capatul câștigător al lui MIN) cu coordonatele Lm și Cm. O evaluare ar fi LM-Lm
e. O evaluare e reprezentată de câte configurații incomplete (adică de 2 simboluri vecine) are MAX din care scădem numărul de configurații incomplete ale lui MIN

5. Se consideră regulile de mai jos:
Dacă **miaună**, **e acoperit de blană** și are **culoare_închisa** atunci **animalul e pisică** cu certitudine 80.
Dacă **miaună** și **e acoperit de pene** atunci **animalul e păun** cu certitudine 50.
Dacă **latră** atunci **animalul e cățel** cu certitudine 90.
Dacă **are urechi lungi** e iepure cu certitudine 80.
Dacă **are culoare neagră** atunci **are culoare închisă** cu certitudine 100.
Dacă **are culoare maro** atunci **are culoare închisă** cu certitudine 100.
Dacă **are culoare albă** atunci **nu are culoare închisă** cu certitudine 100.

Presupunem că un utilizator observă un animal și dorește să consulte sistemul expert cu referire la acel animal. Care dintre următoarele afirmații este adevărată?

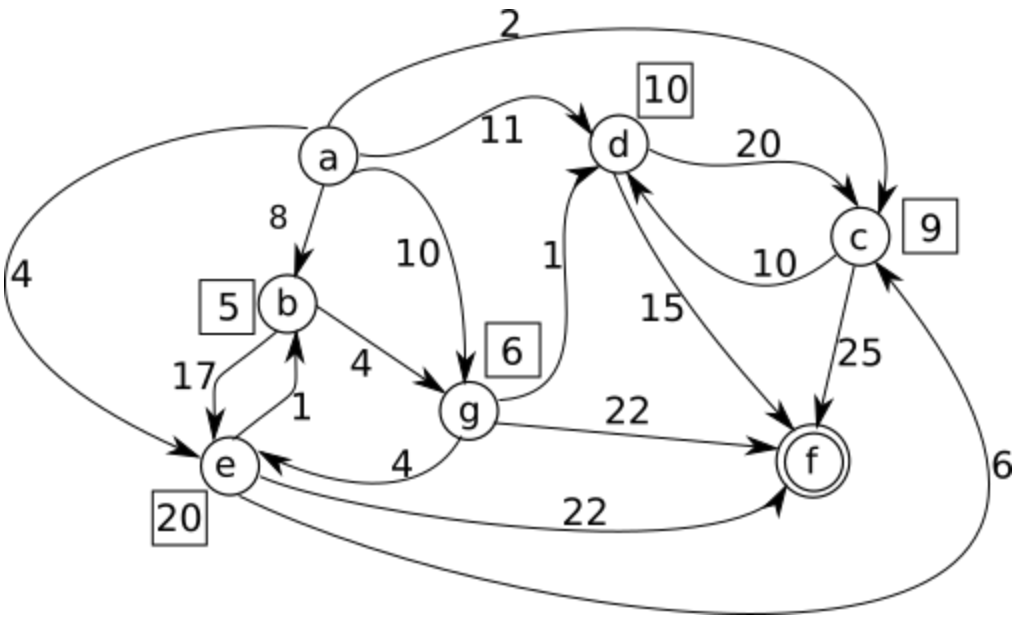
a. Presupunem că sistemul expert dorește să verifice că animalul este pisică și pune întrebări utilizatorului (precum "miauna?", "are blană?", respectiv "ce culoare are?" pentru a determina dacă are sau nu culoare închisă). In acest caz, sistemul expert realizează o căutare de tip înlănțuire înapoi.
b. Dacă animalul observat miaună dar nu are culoare inchisă, sistemul expert nu poate oferi o soluție.
c. Dacă sistemul expert cere întâi introducerea tuturor informațiilor despre animalul observat și pe baza lor trage concluzii intermediare, precum faptul că are sau nu o culoare închisă, iar prin deducții înlănțuite ajunge la concluzia că animalul este pisică, atunci sistemul expert realizează o căutare de tip înlănțuire înapoi.

d. **Dacă utilizatorul răspunde "da" la întrebarea "latră", atunci o soluție a sistemului expert este cățel, însă nu e obligatoriu să fie singura.**

6. Considerăm următoarea problemă asemănătoare cu problema blocurilor. Avem un număr N de stive pe care avem așezate blocuri ce conțin **litere**. Atunci când mutăm un bloc b1 peste un bloc b2, dacă sub blocul b2 e un bloc b3 care are aceeași literă ca și b1, atunci blocul b2 dispare și blocul b1 ajunge direct plasat peste b3 (atenție, dispariția se întâmplă doar când există exact un bloc între două blocuri cu aceeași literă). Se consideră o stare finală (scop) o stare în care în care au rămas doar blocuri cu aceeași literă (nu există în configurație două blocuri cu litere diferite). Costul mutării unui bloc este egal cu numărul de blocuri care au aceeași literă precum blocul de mutat, din configurație (de exemplu dacă în configurație sunt 5 blocuri cu litera "a" costul mutării oricărui bloc cu litera "a" este 5).
Care dintre următoarele moduri de calculare a estimației $\hat{h}$ duce la o estimație admisibilă?

a. **Inițializăm h=0. Luăm pe rând fiecare stivă i și vedem care este litera cu număr maxim de apariții pe acea stivă (de exemplu, pentru stiva: a,a,b,c,a, b, litera ar fi "a"). Numărăm câte blocuri sunt pe stiva i cu literă diferită față de acea literă și adunăm numărul lor la h.**
b. **Căutăm care este litera (notată cu Lit) cu număr <u>maxim</u> de apariții relativ la toată configurația. Numărăm câte blocuri sunt în configurație cu litera diferită de Lit și considerăm estimația egală cu acest număr.**
c. **Căutăm care este litera (notată cu Lit) cu număr <u>minim</u> de apariții relativ la toată configurația. Numărăm câte blocuri sunt în configurație cu litera diferită de Lit și considerăm estimația egală cu acest număr.**
d. **Considerăm estimația ca fiind numărul de stive cu litere diferite pe ele.**
e. **Pentru o stare scop, considerăm estimația 0, iar pentru orice altă stare care nu e scop considerăm estimația ca fiind 2.**

7. Pentru graful de mai jos (cu nodul de start ***a*** și nodul final ***f***), care este al treilea nod care e extins de A*? (prima extindere, numerotată cu 1, se consideră a fi cea a rădăcinii).

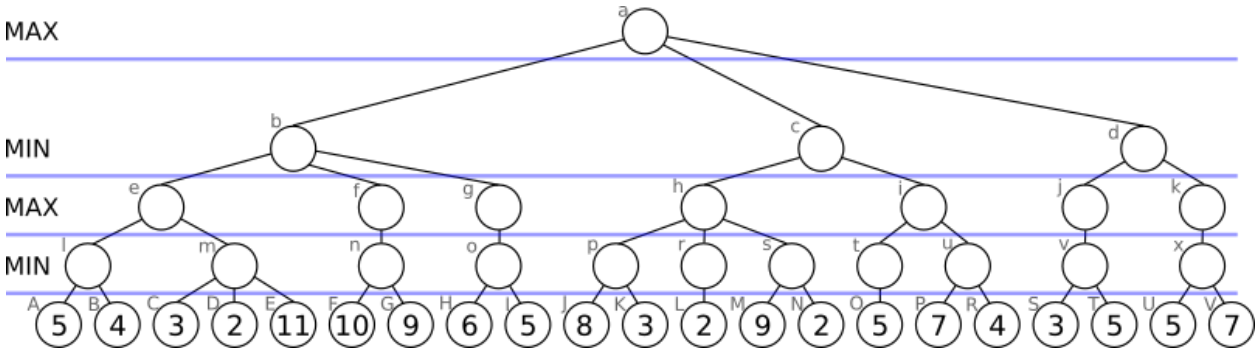


a. a
b. b
c. c
d. d
e. e
f. f
g. g

8. În care dintre următoarele situații coada OPEN a algoritmului A* ajunge vidă înainte de returnarea unei soluții?

a. Nodurile scop sunt pe altă componentă conexă a grafului față de nodul de start.
b. Se poate ajunge la un nod scop doar trecând printr-o muchie cu cost foarte mare în graf.
c. Condiția scop este imposibilă pentru problema dată.
d. Nodul scop este succesor direct al nodului de start.
e. Coada OPEN nu ajunge niciodată vidă înainte de returnarea unei soluții.

9. Considerăm arborele Minimax de mai jos pentru care cunoaștem valorile din frunze:

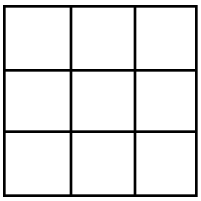


Care dintre următoarele afirmații sunt adevărate?

a. Dacă am aplica alpha-beta pe acest joc atunci nodul *m* nu ar mai fi evaluat.
b. **Dacă am aplica alpha-beta pe acest joc atunci nodurile D și E nu ar mai fi evaluate**
c. Valoare minimax a nodului f este 10.
d. Pentru a calcula valoarea nodului b este suficient să calculăm minimul dintre valorile minimax înscrise în nodurile de la A la I *(i mare)*.
e. Nu există nicio valoare posibilă cu care am putea înlocui valoarea minimax a lui C astfel încât să forțăm variația principală să treacă prin C.

10. Pentru problema X și 0, care dintre următoarele afirmații sunt adevărate?

a. Pentru starea inițială a tablei de joc, reprezentată mai jos, calculând arborele minimax în totalitate, fără a impune o adâncime maximă, numărul de noduri din arbore ar fi egal cu 1!+2!+3!+4!+5!+6!+7!+8!+9! (9 factorial = 1*2*3*4*5*6*7*8*9).



b. O stare finală a jocului este ori una în care a câștigat MAX ori una în care a câștigat MIN.
c. Arborele minimax pentru X și 0 întotdeauna va avea un număr de niveluri mai mic sau egal cu 10.
d. Pentru starea curentă de mai jos (rădăcină a arborelui curent minimax), presupunând că simbolul calculatorului este X, arborele minimax de adâncime maxima 3 (adică numărul de muchii de pe un lanț de la rădăcină la un nod frunză nu poate depăși 3) are exact 10 noduri, incluzând și rădăcina.

X		
0	X	0
X		0

e. Considerând simbolul calculatorului ca fiind 0 (deci la începutul jocului utilizatorul mută primul) și numerotarea nivelurilor începând de la 0 (0 fiind nivelul rădăcinii), atunci pentru arborele minimax de adâncime maximă 4 având ca rădăcină tabla:

0	X	
X		

putem găsi o stare finală a jocului (câștig, pierdere sau remiză) pe nivelul 1 al arborelui

11. Această întrebare este referitoare la algoritmul A* (discutat la laborator sub numele de A* optimizat, cel implementat cu coada OPEN și lista CLOSED). Care dintre următoarele afirmații sunt adevărate?

a. Daca inlocuim costurile pe muchii cu negatul lor (costul c e înlocuit cu -c) putem folosi A* pentru a obține cel mai lung drum-soluție din graf, negând costul soluției oținute cu modificările anterioare.
b. **A* returnează un drum soluție numai când un nod scop se află pe prima poziție în coada OPEN ordonată crescător după valoarea funcției euristice de evaluare, $\hat{f}$ aplicată fiecărui nod din OPEN**.
c. Un nod poate fi in acelasi timp atat in lista open cât și în lista closed.
d. Ordinea nodurilor din lista open este data de ordinea descoperirii lor de către algoritm
e. **Dacă aplicăm A* pe un anumit graf cu un anumit nod de start *a*, orice valoare (chiar și mai mare decât costul oricârui drum de la a la nodul scop am seta pentru estimația $\hat{h}(a)$, dar pentru orice alt nod n, estimația îndeplinește condiția $\hat{h}(n) \leq h(n)$ atunci soluția returnată de A* e in mod cert drumul de cost minim de la a la un nod scop.**