Name: _____ **Class:** _____ **Date:** _____ **ID: A**

Subiecte algebra licenta informatica 4 ani

Multiple Choice
Identify the letter of the choice that best completes the statement or answers the question.

B 1. Fie functia $f : A \to B$ cu proprietatea:
$\forall (x_1, x_2) \in A \times A,\ x_1 \neq x_2 \Rightarrow f(x_1) \neq f(x_2)$.
Care din următoarele afirmatii este adevărată?

a. f este surjectivă
b. f este injectivă
c. f este bijectivă

C 2. Fie $f : \mathbb{Z} \to \mathbb{Z},\ f(x) = 2x + 1$. Care din afirmatiile următoare este adevărată?

a. f este bijectivă
b. f este surjectivă
c. f este injectivă

A 3. Fie $f : \mathbb{Q} \to \mathbb{Q},\ f(x) = 2x + 1$. Care din afirmatiile următoare este adevărată?

a. f este bijectivă
b. f nu este bijectivă

A 4. Fie $f : A \to B$, si $g : B \to C$ două functii injective. Care din afirmatiile următoare este adevărată?

a. $g \circ f$ este injectivă
b. $g \circ f$ nu este injectivă

A 5. Fie $A = \{0,1,2,3,4\}$. Care din afirmatiile următoare este adevărată?

a. $\forall x \in \mathbb{Z},\ \exists a \in A$ astfel încât $x = a \pmod 5$
b. $\exists x \in \mathbb{Z}$ astfel încât $\forall a \in A,\ x \neq a \pmod 5$

B 6. Constanta $a \in \mathbb{R}$ este astfel încât legea de compozitie '*' definită prin
$$\forall (x, y) \in \mathbb{R}^2 : x * y = xy + ax + ay$$
este asociativă. Care din afirmatiile următoare este adevărată?

a. $a \in \{2,5\}$
b. $a \in \{0,1\}$
c. $a = 3$

A 7. Fie grupul simetric $(S_3, \circ)$ (grupul permutarilor de ordinul 3). Atunci numărul subgrupurilor lui S_3 este:

a. 6
b. 4
c. 3

B 8. Fie grupul simetric $\left(S_3,\circ\right)$ (grupul permutarilor de ordinul 3). Atunci numărul subgrupurilor normale ale lui S_3 este:

a. 1
b. 3
c. 4

B 9. Fie permutarea $\sigma \in S_6$,

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 1 & 5 & 2 & 6 & 4 \end{pmatrix}$$

Atunci numărul inversiunilor permutării σ este:

a. 7
b. 5
c. 3

C 10. Fie permutarea $\sigma \in S_6$,

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 2 & 4 & 1 & 6 & 5 \end{pmatrix}$$

Atunci ordinul lui σ în S_6 este:

a. 3
b. 5
c. 6

B 11. Fie $f:\mathbb{Z}\to\mathbb{C}^*$, $f(k)=\cos\frac{2k\pi}{n}+i\sin\frac{2k\pi}{n}$, unde $n\in\mathbb{N}^*$. Atunci $\forall\left(h,k\right)\in\mathbb{Z}^2$:

a. $f\left(h+k\right)=f\left(h\right)+f\left(k\right)$
b. $f\left(h+k\right)=f\left(h\right)f\left(k\right)$
c. $f\left(hk\right)=f\left(h\right)f\left(k\right)$

C 12. Fie morfismul de grupuri $f:\mathbb{Z}\to\mathbb{C}^*$, $f(k)=\cos\frac{2k\pi}{5}+i\sin\frac{2k\pi}{5}$. Atunci:

a. $1+i\in \mathrm{Im}\left(f\right)$
b. $card\left(Im\left(f\right)\right)=6$
c. $Ker\left(f\right)=5\mathbb{Z}=\left\{5q\middle|q\in\mathbb{Z}\right\}$

A 13. Fie $\mathbb{Q}\left(\sqrt{2}\right)=\left\{a+b\sqrt{2}\middle|a,b\in\mathbb{Q}\right\}$. Atunci $\left(\mathbb{Q}\left(\sqrt{2}\right),+,\bullet\right)$ este:

a. corp comutativ
b. inel comutativ cu divizori ai lui zero

C 14. Fie K un subcorp al corpului $\mathbb{R}$. Atunci:

a. $\mathbb{Q} \neq K$ si $\mathbb{Q} \not\subset K$
b. $\mathbb{Q} \bigcap K = \mathbb{Z}$
c. $\mathbb{Q} \subseteq K$

C 15. Fie $f = \hat{3} + \hat{2}X \in \mathbb{Z}_4[X]$. Atunci:

a. $\forall g(X) \in \mathbb{Z}_4[X],\ f(X)g(X) \neq \hat{1}$
b. $\exists g(X) \in \mathbb{Z}_4[X],\ g(X) \neq \hat{0}$ astfel încât $f(X)g(X) = \hat{0}$
c. $\exists g(X) \in \mathbb{Z}_4[X]$ astfel încât $f(X)g(X) = \hat{1}$

B 16. Fie $A, B \in \mathbf{M}_2(\mathbb{R}),\ A = \begin{pmatrix} \cos\frac{2\pi}{n} & -\sin\frac{2\pi}{n} \\ \sin\frac{2\pi}{n} & \cos\frac{2\pi}{n} \end{pmatrix},\ B = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix},\ n \in \mathbb{N}^*$. Atunci:

a. $AB = BA$
b. $AB = BA^{n-1}$
c. $A^{n-1} = I_2$

A 17. Una din afirmatiile următoare este adevărată:

a. $\forall \hat{a}, \hat{b} \in \mathbb{Z}_5,\ (\hat{a} + \hat{b})^5 = \hat{a}^5 + \hat{b}^5$
b. $\exists \hat{a}, \hat{b} \in \mathbb{Z}_5$ astfel încât $(\hat{a} + \hat{b})^5 \neq \hat{a}^5 + \hat{b}^5$
c. $\exists f(X), g(X) \in \mathbb{Z}_5[X]$ astfel încât $(f(X) + g(X))^5 \neq f^5(X) + g^5(X)$

B 18. Fie $G = \left\{ \begin{pmatrix} \hat{1} & \hat{a} & \hat{b} \\ \hat{0} & \hat{1} & \hat{c} \\ \hat{0} & \hat{0} & \hat{1} \end{pmatrix} \middle| \hat{a}, \hat{b}, \hat{c} \in \mathbb{Z}_3 \right\}$. Atunci $\forall A \in G$:

a. $A^3 = A$
b. $A^3 = I_3$
c. $A^3 = A^2$

B 19. Fie $\sigma \in S_n,\ n = 3$, cu proprietatea $\forall \pi \in S_n:\ \sigma \circ \pi = \pi \circ \sigma$. Atunci:

a. $\sigma = (1,2)$
b. $\sigma = e$=permutarea identică
c. $\sigma = (1,2,3)$

B 20. Fie G un grup cu proprietatea $\forall x \in G:\ x^2 = e$. Atunci grupul G este:

a. izomorf cu $(\mathbb{Z}_6, +)$
b. Comutativ
c. izomorf cu $(S_3, \circ)$

A 21. Fie $K = \left\{ \begin{pmatrix} \hat{a} & \hat{b} \\ -\hat{b} & \hat{a} \end{pmatrix} \middle| \hat{a}, \hat{b} \in \mathbb{Z}_3 \right\}$. Atunci $(K, +, \bullet)$ este:

a. corp comutativ cu 9 elemente
b. inel cu divizori ai lui zero
c. corp necomutativ cu 9 elemente

B 22. Fie $d = \begin{vmatrix} x & y & z \\ y+z & x+z & x+y \\ y^2+z^2 & x^2+z^2 & x^2+y^2 \end{vmatrix}$, unde $x, y, z \in R$. Avem

a. $d = (z-x)(z-y)(y-x)(x-y-z)$
b. $d = (z-x)(z-y)(y-x)(x+y+z)$
c. $d = (z-x)(z-y)(y-x)(x-y+z)$

C 23. Fie matricea $A \in M_n(R)$, $A = (a_{ij})$, unde $a_{ij} = \begin{cases} -1 & daca\ i \le j \\ 1 & daca\ i > j \end{cases}$. Avem

a. $\det A = 0$
b. $\det A = 2n+1$
c. $\det A = (-1)^n 2^{n-1}$

A 24. Fie matricele A si $\overline{A}$, $A = \begin{pmatrix} 1 & -1 & 1 & \alpha \\ 1 & 1 & \beta & 1 \\ 2 & -1 & 1 & -1 \end{pmatrix}$, $\overline{A} = \begin{pmatrix} 1 & -1 & 1 & \alpha & \gamma \\ 1 & 1 & \beta & 1 & -1 \\ 2 & -1 & 1 & -1 & 1 \end{pmatrix}$, unde $\alpha, \beta, \gamma \in R$.

Daca $\operatorname{rang} A = \operatorname{rang} \overline{A} = 2$, atunci

a. $\alpha = -1,\ \beta = -1,\ \gamma = 1$
b. $\beta = \gamma$
c. $\alpha = -2,\ \beta = 2,\ \gamma = 1$

C 25. Fie sistemul (S),

$$(S)\ \begin{cases} x+y+z=0 \\ (\beta+\gamma)x+(\alpha+\gamma)y+(\alpha+\beta)z=0, \\ \beta\gamma x+\alpha\gamma y+\alpha\beta z=0 \end{cases} \alpha, \beta, \gamma \in R.$$

Daca sistemul (S) are solutie unica, atunci

a. $\alpha = \beta = 1,\ \gamma = 2$
b. $\alpha = \beta = \gamma = 3$
c. $(\alpha-\beta)(\beta-\gamma)(\gamma-\alpha) \neq 0$

C 26. Fie matricea $A=\begin{pmatrix}\hat{2} & \hat{3} & \hat{a}\\ \hat{1} & \hat{b} & \hat{2}\\ \hat{4} & \hat{1} & \hat{2}\end{pmatrix}\in M_3(Z_6)$. Atunci

a. A este inversabila daca $\hat{a}=\hat{2}$ si $\hat{b}=\hat{1}$
b. A este inversabila daca $\hat{a}=\hat{1}$ si $\hat{b}=\hat{2}$
c. A este inversabila daca $\hat{a}=\hat{3}$ si $\hat{b}=\hat{2}$

B 27. Fie sistemul (S) cu coeficienti in corpul Z_5,

$$(S)\begin{cases}\hat{2}x_1+\hat{3}x_2+x_3+\hat{2}x_4=\hat{2}\\ x_1+\hat{4}x_2+\hat{3}x_3+x_4=\hat{1}\\ \hat{3}x_1+\hat{2}x_2+\hat{4}x_3+\hat{3}x_4=\hat{3}\end{cases}.$$

Atunci

a. sistemul (S) are solutie unica
b. sistemul (S) are exact 25 de solutii
c. sistemul (S) are o infinitate de solutii

C 28. Fie matricea $A=\begin{pmatrix}1 & -3 & m & 1\\ m & 1 & -1 & 0\\ 0 & 1 & 2 & m\end{pmatrix}$, unde $m\in C$. Atunci

a. exista $m\in C$ astfel incat $\operatorname{rang} A=2$
b. exista $m\in C$ astfel incat $\operatorname{rang} A=1$
c. $\operatorname{rang} A=3$ oricare ar fi $m\in C$

A 29. Fie $a_0,a_1,...,a_{n-1},\lambda\in R$ si $d=\begin{vmatrix}\lambda & -1 & 0 & \cdots & 0 & 0\\ 0 & \lambda & -1 & \cdots & 0 & 0\\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots\\ 0 & 0 & 0 & \cdots & \lambda & -1\\ a_0 & a_1 & a_2 & \cdots & a_{n-2} & \lambda+a_{n-1}\end{vmatrix}$. Atunci

a. $d=a_0+a_1\lambda+a_2\lambda^2+...+a_{n-1}\lambda^{n-1}+\lambda^n$
b. $d=0$
c. $d=\lambda^n+a_0a_1...a_{n-1}$

B 30. Fie $A\in M_n(R)$, $A=\begin{pmatrix}x+y & y & \cdots & y\\ y & x+y & \cdots & y\\ \vdots & \vdots & \ddots & \vdots\\ y & y & \cdots & x+y\end{pmatrix}$ si $d=\det A$. Atunci

a. $d=(nx+y)^{n-1}$
b. $d=(x+ny)x^{n-1}$
c. $d=x^n+y^n$

B 31. Fie $A=\begin{pmatrix}2 & 0 & -1\\ 1 & 1 & -1\\ 0 & 1 & 0\end{pmatrix}$, $\lambda \in R$ si $x=\begin{pmatrix}x_1\\ x_2\\ x_3\end{pmatrix}\in R^3$, $x\neq\begin{pmatrix}0\\ 0\\ 0\end{pmatrix}$, astfel incat $Ax=\lambda x$. Atunci

a. $\lambda \in \{-1\}$
b. $\lambda \in \{1,-2\}$
c. $\lambda \in \{9,-4,5\}$

C 32. Fie $A=\begin{pmatrix}\alpha & 1 & 2 & 4\\ 1 & \beta & 2 & 3\\ 1 & 2\beta & 2 & 4\end{pmatrix}$ cu $\alpha,\beta\in R$. Daca rang $A=2$, atunci

a. $\alpha=2,\ \beta=-1$
b. $\alpha=0,\ \beta=3$
c. $\alpha=1,\ \beta=\frac{1}{2}$

B 33. Fie $(G,\bullet)$ un grup de ordin 7 si$a\in G, a\neq e$, unde e este elementul neutru. Avem

a. $a^3=a^{23}$
b. $a^3=a^{24}$
c. $a^3=a^{25}$

A 34. Fie $\sigma\in S_5$, $\sigma=\begin{pmatrix}1 & 2 & 3 & 4 & 5\\ 3 & 5 & 4 & 1 & 2\end{pmatrix}$. Avem

a. $\sigma^{632}=\sigma^2$
b. $\sigma^{632}=\sigma^3$
c. $\sigma^{632}=\sigma^4$

A 35. Fie $f(X_1,X_2,X_3)\in\mathbb{R}[X_1,X_2,X_3], f(X_1,X_2,X_3)=(X_1-X_2)^2+(X_2-X_3)^2+(X_3-X_1)^2$. Avem

a. $\forall\sigma\in S_3, f(X_{\sigma(1)},X_{\sigma(2)},X_{\sigma(3)})=$ $=f(X_1,X_2,X_3)$
b. $\exists\sigma\in S_3, f(X_{\sigma(1)},X_{\sigma(2)},X_{\sigma(3)})\neq$ $\neq f(X_1,X_2,X_3)$

C 36. Fie $A\in M_2(\mathbb{R})$ astfel incat $\det A=1$. Atunci:

a. $\det A^{-1}=-1$
b. $\det A^{-1}=\frac{1}{2}$
c. $\det A^{-1}=1$

B 37. Fie $A\in M_2(\mathbb{R})$ astfel incat $\det A=-2$. Atunci:

a. $\det A^{-1}=2$
b. $\det A^{-1}=-\frac{1}{2}$
c. $\det A^{-1}=\frac{1}{2}$

C 38. Fie $A,B\in M_2(\mathbb{R})$ astfel incat $\det A=1$ si $\det B\neq 0$. Atunci:

a. $\det(BAB^{-1})=\det B$
b. $\det(BAB^{-1})=-1$
c. $\det(BAB^{-1})=1$

C 39. Fie p un numar prim si n numarul de subgrupurilor grupului $(Z_p,+), p>2$. Atunci

a. $n=p$
b. $n=p^2$
c. $n=2$

C 40. Fie n numarul de subgrupurilor grupului $\left(Z_8,+\right)$. Atunci

a. $n=3$
c. $n=4$
b. $n=2$

B 41. Fie G un grup, $a\in G$ si aplicatia $\varphi{:}G\to G,\ \varphi(x)=axa^{-1}$. Atunci:

a. $\exists b\in G$ astfel incat $\varphi(x)\neq b,\forall x\in G$
c. $\exists x_1,x_2\in G, x_1\neq x_2$ astfel incat $\varphi\left(x_1\right)=\varphi\left(x_2\right)$
b. $\varphi\left(xy\right)=\varphi(x)\varphi\left(y\right),\forall x,y\in G$

B 42. Fie $I=\left\{\begin{pmatrix}3a & 3b\\ 3c & 3d\end{pmatrix}\middle|a,b,c,d\in Z\right\}\subset M_2(Z)$. Avem

a. I nu este ideal la stanga al inelului $M_2(Z)$
c. I nu este ideal la dreapta al inelului $M_2(Z)$
b. I este ideal bilateral al inelului $M_2(Z)$

C 43. Fie polinomul $f(X)=X^3+\hat{2}X+\hat{2}\in Z_3[X]$. Atunci:

a. $\exists\hat{a}\in Z_3$ astfel incat $f(\hat{a})=\hat{1}$
c. $f(\hat{c})=\hat{2},\forall\hat{c}\in Z_3$
b. $\exists\hat{b}\in Z_3$ astfel incat $f\left(\hat{b}\right)=\hat{0}$

A 44. Fie $A=\begin{pmatrix}a & b\\ c & d\end{pmatrix}\in M_2(\mathbb{R})$. Atunci:

a. $A^2-(a+d)A+(ad-bc)=O_2$
c. $A^2-(a+d)A+(ad-bc)=3I_2$
b. $A^2-(a+d)A+(ad-bc)=2I_2$

B 45. Fie ecuatia $\sigma\circ x=\pi$, unde $\sigma,\pi\in S_5$, $\sigma=\begin{pmatrix}1&2&3&4&5\\3&4&2&1&5\end{pmatrix}$, $\pi=\begin{pmatrix}1&2&3&4&5\\1&5&4&2&3\end{pmatrix}$. Atunci:

a. $x=\begin{pmatrix}1&2&3&4&5\\2&3&4&5&1\end{pmatrix}$
c. $x=\begin{pmatrix}1&2&3&4&5\\3&4&5&1&2\end{pmatrix}$
b. $x=\begin{pmatrix}1&2&3&4&5\\4&5&2&3&1\end{pmatrix}$

A 46. Fie ecuatia $AX=B$, unde $A,B\in M_2\left(Z_3\right)$, $A=\begin{pmatrix}\hat{2}&\hat{2}\\ \hat{1}&\hat{2}\end{pmatrix}$, $B=\begin{pmatrix}\hat{0}&\hat{2}\\ \hat{1}&\hat{0}\end{pmatrix}$. Atunci:

a. $X=\begin{pmatrix}\hat{2}&\hat{2}\\ \hat{1}&\hat{2}\end{pmatrix}$
c. $X=\begin{pmatrix}\hat{2}&\hat{0}\\ \hat{0}&\hat{2}\end{pmatrix}$
b. $X=\begin{pmatrix}\hat{1}&\hat{1}\\ \hat{2}&\hat{1}\end{pmatrix}$

C 47. Fie U multimea elementelor inversabile ale inelului Z_{12}. Avem:

a. $U = \left\{\hat{5}, \hat{9}, 1\hat{1}\right\}$

b. $U = \left\{\hat{3}, \hat{7}, 1\hat{1}\right\}$

c. $U = \left\{\hat{1}, \hat{5}, \hat{7}, 1\hat{1}\right\}$

C 48. Fie $f\left(X_1, X_2, X_3\right) \in \mathbb{R}\left[X_1, X_2, X_3\right], f\left(X_1, X_2, X_3\right) = X_1X_2X_3 + X_1X_2 + X_2X_3 + X_1X_3 + \lambda\left(X_1 + X_2\right)$, cu $\lambda \in \mathbb{R}$. Daca $f\left(X_{\sigma(1)}, X_{\sigma(2)}, X_{\sigma(3)}\right) = f\left(X_1, X_2, X_3\right), \forall \sigma \in S_3$, avem

a. $\lambda = 1$

b. $\lambda = -1$

c. $\lambda = 0$

A 49.

Sa se afle valorile lui *a*, pentru care sistemul urmator are solutii nenule

$$\begin{cases} x + 4y + z - 2t = 0 \\ 2x - 5y - 4z + 2t = 0 \\ 5x + 3y - 3z + 4t = 0 \\ 2x - ay - 2z = 0 \end{cases}$$

a. $\frac{2}{3}$

b. 1

c. $\frac{1}{3}$

d. 2

A 50. Sa se rezolve ecuatia matriciala $X \cdot \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 3 & 4 & 1 \end{pmatrix} = \begin{pmatrix} 6 & 9 & 8 \\ 0 & 1 & 6 \end{pmatrix}$

a. $\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \end{pmatrix}$

b. $\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$

c. $\begin{pmatrix} 1 & 1 & -1 \\ 1 & 1 & -1 \end{pmatrix}$

d. $\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix}$

Name: _____ **Class:** _____ **Date:** _____ **ID: A**

Subiecte algoritmica si programarea calculatoarelor licenta informatica 4 ani

Multiple Choice
Identify the letter of the choice that best completes the statement or answers the question.

C 1. O problemă pentru care există un algoritm de rezolvare se numeste
   a. rezolvabilă
   b. solutionabilă
   c. decidabilă

B 2. Un proces de calcul este un algoritm dacă
   a. este determinat si eficace
   b. este finit si furnizează cel putin o iesire
   c. este finit si nu furnizează iesiri

C 3. O schemă logică este structurată dacă
   a. este formată din structuri logice
   b. are cel putin o intrare
   c. este formată din structuri fundamentale

B 4. Structurile alternative asigură
   a. înlănțuirea operatiilor de prelucrare
   b. ramificarea prelucrărilor
   c. repetarea unor secvente de prelucrare

C 5. Este posibilă utilizarea unei structuri FOR:
   a. totdeauna când este necesară repetarea unei prelucrări
   b. la repetarea prelucrări pe baza variatiei unui indice care se modifică în prelucrare printr-o functie oarecare
   c. la repetarea prelucrări pe baza variatiei unui indice care se modifică unitar doar în finalul prelucrării

C 6. Spunem că un algoritm este cu timpul de executie T(n) este polinomial dacă există $P \in \mathbb{N}[X]$ astfel încât
   a. T(n) este de forma $\log_a P$
   b. T(n) este de forma a^P
   c. $T(n) \le P(n)$ pentru orice n dat

A 7. Spunem că o problemă este NP dacă
   a. problema are un algoritm de rezolvare care nu este polinomial
   b. problema este ne prelucrabilă
   c. problema este ne procedurală

B 8. Ordinul algoritmului O(a(n)) reprezintă
   a. timpul de executie pentru n pasi de aplicare
   b. partea cea mai semnificativă a functiei care dă timpul de executie este a(n)
   c. o exprimare prin a(n) a dimensiunii datelor intermediare pentru n date de intrare

C 9. Într-o instructiune de atribuire toate datele
   a. toate datele se transformă la formatul variabilei receptor
   b. se reduc la formatul întreg
   c. trebuie sa fie de tipuri compatibile

A 10. Care din următoarele afirmatii nu este valabilă în limbajul PASCAL:
   a. Operatorul OR are operanzi CHAR
   b. Operatorul OR are ca operanzi expresii întregi
   c. Operatorul OR are ca operanzi expresii logice

A 11. Sirurile de cifre prezente ca atare într-un program sunt
   a. constante întregi
   b. variabile de tip întreg
   c. cuvinte cheie

A 12. În limbajul PASCAL cuvântul RECORD defineste
   a. un articol definit utilizator
   b. înregistrarea de informatii pe un suport extern
   c. o constantă predefinită

C 13. În limbajul PASCAL, o instructiune compusă este
   a. o instructiune IF
   b. o instructiune WHILE
   c. o secventă de instructiuni delimitată de BEGIN si END

B 14. În limbajul PASCAL, care din următoarele instructiuni nu implică folosirea instructiunii compuse
   a. FOR
   b. REPEAT
   c. CASE

C 15. În limbajul PASCAL, subprogramele FUNCTION
   a. contin doar parametrii de intrare
   b. returnează toate rezultatele prin numele functiei
   c. nu respectă nici una din conditiile a si b

C 16. Pentru un subprogram PASCAL este adevărată afirmatia:
   a. Toti parametrii au tipul predefinit
   b. Toti parametrii de intrare nu contin definirea tipului
   c. Unii parametrii definiti sub VAR pot avea tipul absent

C 17. În limbajul C:
   a. orice declarare impune alocarea de memorie
   b. nici o declarare nu impune alocarea de memorie
   c. nici una din afirmatiile a si b nu este adevărată

A 18. În limbajul C, declaratia REGISTER :
   a. induce memorarea datelor INT si CHAR într-un registru procesor
   b. induce alocarea de memorie interna pentru salvarea unui registru
   c. induce alocarea unui registru suplimentar în memoria internă

C 19. În imbajul C:
   a. există variabile care se alocă automat
   b. nu există variabile predefinite
   c. orice utilizare de variabilă trebuie să urmeze după o declaratie a variabilei

B 20. În limbajul C, cuvântul cheie UNION:
   a. declară un operator pentru reuniunea elementelor a două multimi
   b. se foloseste pentru a suprapune mai multe variabile în acelasi spatiu de memorie
   c. produce concatenarea a două siruri de caractere

C 21. În limbajul C, nu este adevărată afirmatia
   a. operatorii aritmetici sunt: +, -, *, / si %
   b. operatorii bit sunt &, |, <<, >> si ~
   c. operatorii bit sunt AND, OR, NOT, XOR

A 22. În limbajul C, instructiunea BREAK produce în contextul instructiunilor de ciclare:
   a. întreruperea completă a ciclării
   b. abandonarea programului
   c. abandonarea iteratiei curente

C 23. Nu este o modalitate de descriere a algoritmilor:
   a. masina Turing
   b. sistemele post
   c. sistemele logice
   d. functiile recursive

D 24. Nu este strategie pentru elaborarea algoritmilor
   a. greedy
   b. divide et impera
   c. backtraking
   d. statistica

A 25. Nu este strategie pentru elaborarea algoritmilor
   a. programare liniara
   b. programare dinamica
   c. euristica

D 26. Următoarea structură nu este repetitivă
   a. WHILE(p;a)
   b. REPEAT(p;a)
   c. FOR(p;a,b,c)
   d. CASE(p_1,p_2,…,p_k;a_1,a_2,…,a_k)

Completion
Complete each sentence or statement.

27. Descrierea algoritmilor prin limbaj conventional este numită limbaj pseudocod
28. Daca G=(V,E) este un graf, elementele lui E se numesc muchii
29. Daca G=(V,E) este un graf orientat, elementele lui E se numesc arce
30. Daca G=(V,E) este un graf, F este submultime a lui E, atunci H=(V,F) se numeste graf partial
31. Daca G=(V,E) este un graf orientat, un drum orientat în G în care extremitătile coincid se numeste circuit
32. Daca G=(V,E) este un graf, un lant în G în care extremitătile coincid se numeste ciclu
33. Într-un graf un lant în care toate virfurile, cu exceptia extremitătilor, sunt distincte se numeste ciclu elementar
34. Într-un graf, un lant elementar care contine toate cârfurile grafului se numeste lant hamiltonian
35. Un graf în care orice două vârfuri sunt conectate se numeste graf complet

Name: _____ **Class:** _____ **Date:** _____ **ID: A**

Subiecte baze de date licenta informatica 4 ani

True/False
Indicate whether the sentence or statement is true or false.

T 1. Modelarea oricarui sistem din *lumea reala* porneste de la realitate si se exprima printr-o entitate.

F 2. Elementele principale ale unei BDR sunt: clasa, obiectul, atributul, metoda, etc.
Elementele principale ale unei BDOO sunt: tabelul, campurile si inregistrarile.

F 3. Se numeste atribut o colectie persistenta, neredundanta, coerenta logic de date corelate.

F 4. Se numeste inregistrare o unitate elementara de date ce poseda un nume

T 5. Etapele realizarii diagramei E/R:

1. Se identifica entitatile
2. Se identifica relatiile dintre entitati (legaturile)
3. Se stabilesc cardinalitatile
4. Se identifica atributele pentru fiecare entitate
5. Se stabilesc cheile (atributele de identificare)

T 6. Restrictii ale modelului ierahic sunt:
- La inserare nu se pot introduce noi realizari ale unei inregistrari subordonate daca nu sunt cunoscuti superiorii;
- Daca se sterge o realizare radacina a unei inregistrari, atunci se sterg automat toate inregistrarile subordonate (tot subarborele).

F 7. Restrictii ale modelului ierahic
- La inserare se pot introduce noi realizari ale unei inregistrari subordonate chiar daca nu sunt cunoscuti superiorii;
- Daca se sterge o realizare radacina a unei inregistrari, atunci se sterg automat toate inregistrarile subordonate (tot subarborele).

T 8. Modelul retea:
– Aranjeaza articolele intr-o lista cu legaturi de tip *graf orientat*, un articol putand avea mai multi parinti.
– Deosebirea fata de modelul ierarhic este ca intre un nod inferior si un nod superior exista legatura de tip 1:n.

F 9. Modelul ierarhic:
– Aranjeaza articolele intr-o lista cu legaturi de tip *graf orientat*, un articol putand avea mai multi parinti.
– Deosebirea fata de modelul retea este ca intre un nod inferior si un nod superior exista legatura de tip 1:n.

T 10. Relatia virtuala este numita si vizualizare, relatie derivata, filtru, tabel view, vedere – ea cuprinde definitia vizualizarii. Este un tabel virtual al datelor, compus din campuri provenite din doua sau mai multe tabele sau/si campuri din alte vizualizari in care nu se pot face modificari, stergeri, deci are avantajul pastrarii securitatii tabelului initial de date.
Vizualizarile pot fi:
– Vizualizari de date (tabele);,
– Vizualizari de validare (tabele de validare);
– Vizualizari agregate (informatii selectate din mai multe tabele).

F 11. Relatia virtuala este numita si vizualizare, relatie derivata, filtru, tabel view, vedere – ea cuprinde definitia vizualizarii. Este un tabel virtual al datelor, compus din campuri provenite din doua sau mai multe tabele sau/si campuri din alte vizualizari in care se pot face modificari, stergeri, deci are avantajul pastrarii securitatii tabelului initial de date.
Vizualizarile pot fi:
- Vizualizari de date (tabele);,
- Vizualizari de validare (tabele de validare);
- Vizualizari agregate (informatii selectate din mai multe tabele).

T 12. Una din etapele ce trebuie parcurse pentru realizarea schemei conceptuale este urmatoarea:
Atributele singulare devin coloane.

F 13. Una din etapele ce trebuie parcurse pentru realizarea schemei conceptuale este urmatoarea:
Atributele singulare devin linii;

T 14. SGBD-urile sunt construite modular. Exemple de astfel de module sunt:
Module ce contin programele de gestiune a bazei:
Module pentru LDD
Module pentru LMD
Module utilitare
Module pentru LCD

T 15. Comenzile SQL se incheie cu ; (punct si virgula).

F 16. Crearea unei tabele cu SQL in Access sa face cu ajutorul clauzei ALTER TABLE.

T 17. Modificarea structurii unei tabele cu SQL in ACCESS se poate face folosind clauza ALTER TABLE.

F 18. Cu ajutorul sintaxei :
ALTER TABLE nume_tabela ADD nume_camp tip_data;
se adauga un camp tabelei TABLE

F 19. Crearea unei noi tabele cu SQL in ACCESS se face folosind clauza DROP TABLE.

F 20. In ACCESS, cu clauza
```
SELECT *
FROM TABELA1;
```
se selecteaza numai primul camp din TABELA1.

T 21. In ACCESS selectarea si redenumirea unor campuri se poate face cu clauza:
```
SELECT camp1 AS nume1
FROM nume_tabela1;
```

T 22. In ACCESS, pentru date de tip text, campurile dintr-un tabel pot fi combinate (concatenate) astfel incat mai multe campuri sa formeze un singur camp in rezultatul interogarii astfel:
```
SELECT camp1 + " " + camp2 + " " + camp3 AS campcompus,
FROM nume_tabela1;
```

F 23. Cu clauza DROP TABLE se pot redenumi campurile unei tabele in Access.

T 24. Stergerea unei tabele folosind SQL in ACCESS se face cu clauza DROP TABLE.

F 25. Crearea unei noi tabele cu SQL in ACCESS se face cu clauza UPDATE.

T 26. Cu clauza SELECT se pot extrage informatii din baza de date.

T 27. Deschiderea tabelului TABEL_CARTI pentru a privi datele este echivalenta cu activarea clauzei SQL:
SELECT *
From TABEL_CARTI;

F 28. Deschiderea tabelului TABEL_CARTI pentru a privi datele este echivalenta cu activarea clauzei SQL:
SELECT *
From TABEL_CARTI!

F 29. Pentru a selecta unul din campurile tabelei TABEL_STUDENTI, se foloseste clauza:
SELECT *
From TABEL_STUDENTI;

T 30. Pentru ca baza de date distribuita sa fie usor prelucrabila, prin sistemul distribuit se pun la dispozitia acesteia o serie de independente.
Una dintre acestea este independenta fragmentarii. Fragmentarea poate fi: orizontala (fragmentele au structura identica cu cea a multimii de date, dar difera prin continutul datelor), verticala (fragmentele contin doar o parte din structura relatiei), mixta (fragmentarea orizontala a unui fragment vertical sau fragmentare verticala a unui fragment orizontal).

F 31. Pentru ca baza de date distribuita sa fie usor prelucrabila, prin sistemul distribuit se pun la dispozitia acesteia o serie de independente.
Una dintre acestea este independenta fragmentarii. Fragmentarea poate fi: orizontala (fragmentele contin doar o parte din structura relatiei) , verticala (fragmentele au structura identica cu cea a multimii de date, dar difera prin continutul datelor) , mixta (fragmentarea orizontala a unui fragment vertical sau fragmentare verticala a unui fragment orizontal).

T 32. Pentru ca baza de date distribuita sa fie usor prelucrabila, prin sistemul distribuit se pun la dispozitia acesteia o serie de independente.
Autonomia statiilor - permite fiecarei statii sa-si controleze si sa-si manipuleze datele locale, independent de alte statii. Administrarea unei BDD este complet descentralizata, bazele locale fiind controlate independent de un administrator local.

F 33. In organizarea „ideala” a unei BDD se disting doua nivele de date:
- Nivelul global – aici fiecare baza locala din BDD este tratata ca o baza centralizata
- Nivelul local - aici se realizeaza integrarea bazelor de date locale intr-o baza de date globala

F 34. In cazul SGBDD, pentru a satisface cererile in ordinea emiterii se utilizeaza marcile de timp astfel:
- fiecare cerere primeste automat la emitere o marca de timp (identificatorul nodului si timpul ceasului local).
- toate articolele din BDD au o marca de timp, care ramane neschimbata la fiecare actualizare a cererii.
- cererile se executa in ordinea emiterii marcilor

T 35. Intr-o BDD, pentru a satisface cererile in ordinea emiterii se utilizeaza inelul virtual :
- nodurile retelei sunt inlantuite logic intr-un inel virtual pe care se deplaseaza un token.
- daca un nod detine token-ul el poate transmite.
- token-ul trece din nod in nod pana la nodul caruia ii este adresat.

cand token-ul ajunge la nodul din care a plecat, acesta devine liber, iar token-ul se deplaseaza spre nodul urmator.

F 36. Principalele concepte care stau la baza unui MDOO sunt: obiectul, clasa, fragmentarea, incapsularea, persistenta, mostenirea, polimorfismul si colectia.

F 37. Intr-un MDOO, orice entitate din lumea reala este un obiect si reciproc, orice obiect reprezinta o abstractizare a unei entitati a lumii reale. Un obiect este un grup de date structurate, identificate printr-o referinta unica.

T 38. Componentele de baza ale unui SGBDOO sunt: utilitarele, limbajele si gestiunea obiectelor.

F 39. Integritatea semantica a unui SGBDOO - se realizeaza prin autentificari si accesul controlat la date.

T 40. Integritatea semantica a unui SGBDOO – se realizeaza prin diferite tipuri de constrangeri (de tiparire, ale valorilor domeniului, de unicitate), care pot fi activate la executie, la compilare, la trimiterea unui mesaj, etc.

Multiple Choice
Identify the letter of the choice that best completes the statement or answers the question.

A 41. Se numeste .....o unitate elementara de date ce poseda un nume.
a. Articol
b. Entitate
c. Inregistrare
d. SGBD

A 42. ..... planifica si realizeaza designul bazei.
a. Analistul pentru baze de date
b. Administratorul bazei de date
c. Programatorul de aplicatii
d. Utlizatorul

B 43. …….. se ocupa cu modul de intrare a datelor in baza si cu buna functionare a bazei de date; defineste schemele: conceptuala, interna si externa, raspunzand de toate modificarile ce se fac asupra bazei; da drepturi de acces utilizatorilor ; defineste procedurile de restaurare si de salvare, etc.
a. Analistul pentru baze de date
b. Administratorul bazei de date
c. Programatorul de aplicatii
d. Utilizatorul

C 44. ..... intelege activitatea firmei sau a aplicatiei pe care urmeaza sa o implementeze; dezvolta programe in timp (in diferite limbaje de programare: C, COBOL, PASCAL, etc.), gaseste noi informatii, realizeaza noi rapoarte.
a. Analistul pentru baze de date
b. Administratorul bazei de date
c. Programatorul de aplicatii
d. Utilizatorul

A 45. Bazele de date folosesc mai multe tipuri de limbaje. Limbajele ..... definesc:
- Tipurile de date;
- Relatiile dintre date;
- Atributele asociate relatiilor, structura lor, domeniul lor de definitie (ex: numele, forma de memorare, lungimea atributelor unei entitati);
- Modul de accesare a datelor;
- Criteriile de validare automata a datelor.

a. LDD
b. LMD
c. LCD
d. Limbajele de programare C si C++

B 46. Bazele de date folosesc mai multe tipuri de limbaje. Limbajele ....., actioneaza prin comenzi cu o anumita structura, cu ajutorul lor utilizatorii autorizati au acces la operatiile de inserare, actualizare, stergere a datelor; se mai numesc si limbaje de interogare.
a. LDD
b. LMD
c. LCD
d. Limbajele de programare C si C++

C 47. Bazele de date folosesc mai multe tipuri de limbaje. Limbajele ..... raspund de: integritatea datelor, confidentialitatea datelor, performantele bazei de date.
   a. LDD
   b. LMD
   c. LCD
   d. Limbajele de programare C si C++

B 48. Se numeste .....o colectie de programe care permite crearea si intretinerea unei baze de date.
   a. Dictionarul bazei de date
   b. SGBD
   c. LMD
   d. Normalizare

C 49. ..... - urile sunt o interfata intre utilizatori si sistemul de operare. Ele ajuta la construirea unor baze de date, la introducerea informatiilor in bazele de date si dezvoltarea de aplicatii privind bazele de date; dau acces utilizatorilor la date prin intermediul unui limbaj apropiat de modul obisnuit de exprimare, facand abstractie de algoritmi, aplicatii si de modul de memorare a datelor.
   a. LMD
   b. LCD
   c. SGBD
   d. LDD

A 50. Diagrama entitate-relatie a fost introdusa pentru prima data de .....in 1976 si este un model neformalizat de reprezentare a fenomenelor din lumea reala.
   a. Chen
   b. Codd
   c. Gardarin
   d. ANSI-X3/SPARK

B 51. Modelul care aranjeaza articolele intr-o lista cu legaturi de tip *graf orientat*, un articol putand avea mai multi parinti si in care intre un nod inferior si un nod superior exista legatura de tip 1:n este:
   a. Modelul ierarhic
   b. Modelul retea
   c. Modelul liniar
   d. Nu exista un asemenea model

D 52. Modelul ..... a fost introdus de E.F. Codd in 1970 si este descris cu ajutorul teoriei matematice a relatiilor. Este un model orientat spre multimi, este simplu si riguros matematic.
   a. Ierarhic
   b. Retea
   c. Orientat obiect
   d. Relational

B 53. Multimea tuturor schemelor relationale corespunzatoare unei aplicatii se numeste ..... bazei de date relationale
   a. dictionarul
   b. schema
   c. SGBD-ul

B 54. Multimea tuturor schemelor relationale corespunzatoare unei aplicatii se numeste schema bazei de date relationale, iar continutul curent al relatiilor la un moment dat se numeste baza de date .....
   a. Orientata obiect
   b. Relationala
   c. distribuita

A 55. Modelul relational, are la baza cele 13 reguli de fidelitate ale lui Codd in raport cu care un SGBD poate fi analizat cat este de relational. Aceste reguli au fost completate in timp, numarul lor fiind in jur de 100. Una din cele 13 reguli date de Codd este:
Un SGBD relational trebuie sa-si gestioneze singur baza de date (nici un SGBD nu contine numai caracteristici relationale.). Se numeste .....
a. regula gestionarii datelor
b. regula reprezentarii informatiei
c. regula accesului garantat la date
d. regula reprezentarii informatiei necunoscute
e. regula dictionarelor de date

B 56. Modelul relational, are la baza cele 13 reguli de fidelitate ale lui Codd in raport cu care un SGBD poate fi analizat cat este de relational. Aceste reguli au fost completate in timp, numarul lor ajungand la 100. Una din cele 13 reguli date de Codd este:
La nivel logic informatia trebuie sa fie reprezentata explicit prin valori in tabele numite relatii (regula ce nu poate fi incalcata intr-o baza de date relationala.). Se numeste .....
a. regula gestionarii datelor
b. regula reprezentarii informatiei
c. regula accesului garantat la date
d. regula reprezentarii informatiei necunoscute
e. regula dictionarelor de date

C 57. Modelul relational, are la baza cele 13 reguli de fidelitate ale lui Codd in raport cu care un SGBD poate fi analizat cat este de relational. Aceste reguli au fost completate in timp, numarul lor ajungand la 100. Una din cele 13 reguli date de Codd este:
Orice element de date (valoare atomica) din baza se poate accesa utilizand o combinatie intre numele relatiei, cheia primara, si numele atributului(coloanei). Se numeste.....
a. regula gestionarii datelor
b. regula reprezentarii informatiei
c. regula accesului garantat la date
d. regula reprezentarii informatiei necunoscute
e. egula dictionarelor de date

D 58. Modelul relational, are la baza cele 13 reguli de fidelitate ale lui Codd in raport cu care un SGBD poate fi analizat cat este de relational. Aceste reguli au fost completate in timp, numarul lor ajungand la 100. Una din cele 13 reguli date de Codd este:
Informatiile necunoscute trebuie sa se poata defini printr-un tip de date numit NULL, diferit de spatiul necompletat sau de un sir de caractere blanc (valoarea zero, un sir vid de caractere sau o valoare necunoscuta sunt notiuni complet diferite intr-un acelasi camp de date si trebuie ca SGBD-ul sa permita diferentierea lor.). Valorile nule reprezinta varianta *NU STIU*. Se numeste.....
a. regula gestionarii datelor
b. regula reprezentarii informatiei
c. regula accesului garantat la date
d. regula reprezentarii informatiei necunoscute
e. regula dictionarelor de date

E 59. Modelul relational, are la baza cele 13 reguli de fidelitate ale lui Codd in raport cu care un SGBD poate fi analizat cat este de relational. Aceste reguli au fost completate in timp, numarul lor ajungand la 100. Una din cele 13 reguli date de Codd este:
Asupra descrierii bazei de date (tabelelor de descriere) trebuie sa se aplice aceleasi operatii ca si asupra tabelelor de date. Se numeste .....
a. regula gestionarii datelor
b. regula reprezentarii informatiei
c. regula accesului garantat la date
d. regula reprezentarii informatiei necunoscute
e. regula dictionarelor de date

A 60. Modelul relational, are la baza cele 13 reguli de fidelitate ale lui Codd in raport cu care un SGBD poate fi analizat cat este de relational. Aceste reguli au fost completate in timp, numarul lor ajungand la 100. Una din cele 13 reguli date de Codd este:
Trebuie sa existe cel putin un limbaj de interogare pentru manipularea bazei de date (in general acesta este SQL.). Limbajul tre-buie sa permita: definirea datelor, definirea vizualizarilor, manipularea datelor, autorizari, restrictii de integritate. Se numeste.....
a. regula limbajului de interogare
b. regula de actualizare a vizualizarii.
c. regula limbajului de nivel inalt
d. regula independentei fizice a datelor

B 61. Modelul relational, are la baza cele 13 reguli de fidelitate ale lui Codd in raport cu care un SGBD poate fi analizat cat este de relational. Aceste reguli au fost completate in timp, numarul lor ajungand la 100. Una din cele 13 reguli date de Codd este:
Un SGBD trebuie sa poata determina daca o vizualizare poate fi actualizata sau nu si sa stocheze rezultatul interogarii intr-un dictionar de tipul unui catalog de sistem. Se numeste .....
a. regula limbajului de interogare
b. regula de actualizare a vizualizarii
c. regula limbajului de nivel inalt.
d. regula independentei fizice a datelor

C 62. Modelul relational, are la baza cele 13 reguli de fidelitate ale lui Codd in raport cu care un SGBD poate fi analizat cat este de relational. Aceste reguli au fost completate in timp, numarul lor ajungand la 100. Una din cele 13 reguli date de Codd este:
Regulile de manipulare asupra unei relatii luata ca intreg se aplica si operatiilor de regasire, inserare, actualizare sau stergere a datelor (limbajele de nivel scazut actioneaza asupra unei singure inregistrari, iar limbajele de nivel inalt actioneaza asupra mai multor inregistrari in acelasi timp. Codd spune ca indiferent de nivel, limbajele trebuie sa respecte aceleasi reguli). Se numeste .....
a. regula limbajului de interogare
b. regula de actualizare a vizualizarii
c. regula limbajului de nivel inalt
d. regula independentei fizice a datelor

D 63. Modelul relational, are la baza cele 13 reguli de fidelitate ale lui Codd in raport cu care un SGBD poate fi analizat cat este de relational. Aceste reguli au fost completate in timp, numarul lor ajungand la 100. Una din cele 13 reguli date de Codd este:
Modul de depunere a datelor sau de acces la ele nu influenteaza programele de aplicatii sau activitatile utilizatorilor (utilizatorul nu trebuie sa stie daca datele au fost stocate pe Unix sau pe Windows 2000 Server, el trebuie sa cunoasca numai numele serverului). Se numeste .....
a. regula limbajului de interogare
b. regula de actualizare a vizualizarii
c. regula limbajului de nivel inalt
d. regula independentei fizice a datelor

A 64. Modelul relational, are la baza cele 13 reguli de fidelitate ale lui Codd in raport cu care un SGBD poate fi analizat cat este de relational. Aceste reguli au fost completate in timp, numarul lor ajungand la 100. Una din cele 13 reguli date de Codd este:
Programele de aplicatie nu trebuie sa afecteze manipularea datelor.Se numeste .....
a. regula independentei logice a datelor
b. regula independentei datelor din punct de vedere al integritatii
c. regula versiunii procedurale a SGBD-ului
d. regula independentei datelor din punct de vedere al distribuirii

A 65. Se numeste ..... a doua relatii R_1,R_2 apartin $R_n(A_1,\ldots, A_n)$, relatia R care are aceeasi schema (structura) ca R_1(implicit R_2) si care are multimea tuplurilor formata din tuplurile celor doua relatii luate o singura data.

a. Reuniunea
b. Diferenta
c. Produsul cartezian
d. Intersectia

B 66. Se numeste ..... a doua relatii R_1,R_2 apartin $R_n(A_1,\ldots, A_n)$, relatia R care are aceeasi schema (structura) ca R_1 (implicit R_2) si care are multimea tuplurilor formata din tuplurile relatiei R_1 ce nu se gasesc printre tuplurile relatiei R_2.

a. Reuniunea
b. Diferenta
c. Produsul cartezian
d. Intersectia

C 67. Se numeste ..... a doua relatii R_1 apartine $R_n(A_1,\ldots,A_n)$ de aritate n si R_2 apartine $R_m(B_1,\ldots,B_m)$ de aritate m, cu $A_1,\ldots,A_n$, $B_1,\ldots,B_m$ distincti, relatia R cu schema obtinuta prin concatenarea schemei relatiei R_1 cu schema relatiei R_2 si care are multimea tuplurilor formata din toate perechile de tupluri de aritate n+m astfel incat primele n componente formeaza un tuplu in R_1 iar urmatoarele m un tuplu in R_2.

a. Reuniunea
b. Diferenta
c. Produsul cartezian
d. Intersectia

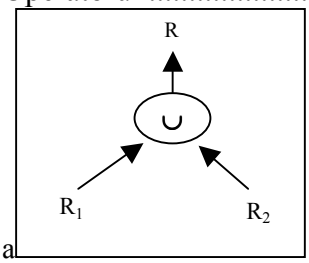
B 68. Operatorul ..... are notatiile: R_1-R_2, sau REMOVE(R_1,R_2), sau ..... (R_1,R_2), sau MINUS(R_1,R_2),.

a. UNION
b. DIFFERENCE
c. PRODUCT
d. INTERSECT

C 69. Operatorul ..... . are notatiile: R_1xR_2, ...(R_1,R_2), TIMES(R_1,R_2).

a. UNION
b. DIFFERENCE
c. PRODUCT
d. INTERSECT

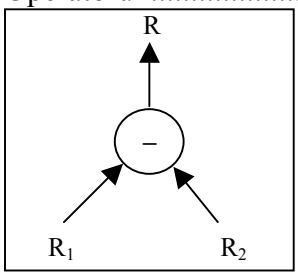
A 70. Operatorul ..... are reprezentarea



a

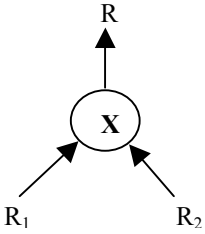
a. UNION
b. DIFFERENCE
c. PRODUCT
d. INTERSECT

B 71. Operatorul ..... are reprezentarea



a. UNION
b. DIFFERENCE
c. PRODUCT
d. INTERSECT

C 72. Operatorul ..... . are reprezentarea



a. UNION
b. DIFFERENCE
c. PRODUCT
d. INTERSECT

B 73. Se numeste ..... a relatiei R_1 apartine $R_n(A_1,\ldots,A_n)$ printr-o conditie cond, relatia unara R cu aceeasi schema ca R_1 si cu multimea tuplurilor formata din tuplurile relatiei R ce satisfac conditia cond.
a. proiectia
b. selectia
c. intersectia
d. diviziunea

C 74. Se numeste ..... a doua relatii, relatia binara R cu aceeasi schema ca R_1(implicit R_2) si cu multimea tuplurilor formata din tuplurile care apartin ambelor relatii in acelasi timp.
a. proiectia
b. selectia
c. intersectia
d. diviziunea

B 75. Se numeste ..... (compunere) operatia algebrei relationale care construieste o noua relatie R prin concatenarea (combinarea) unor tupluri din R_1 apartine $R_n(A_1,\ldots,A_n)$ cu tupluri din R_2 apartine $R_m(B_1,\ldots,B_m)$, respectand anumite conditii puse tuplurilor.

Operatorul combina produsul cartezian, selectia si proiectia.
a. intersectie
b. jonctiune
c. diviziune

B 76. Se numeste ..... a relatiilor R_1 si R_2 relatia R cu schema formata din reuniunea atributelor relatiilor R_1 si R_2 (cele comune se iau o singura data) si cu multimea tuplurilor formata din tuplurile R_1 concatenate cu tuplurile din R_2 pentru care valorile atributelor comune au valori identice.

a. θ-jonctiune
b. jonctiunea naturala
c. semi-jonctiune

C 77. Se numeste ..... a relatiei R_1 cu relatia R_2 prin conditia cond, relatia R cu aceeasi schema ca si R_1 si multimea tuplurilor formata *numai din tuplurile relatiei* R_1 care concatenate cu tupluri din R_2 verifica conditia cond.

a. θ-jonctiune
b. jonctiunea naturala
c. semi-jonctiune

A 78. Se numeste **.....** procesul de organizare si determinare a coloanelor unui tabel, astfel incat redundanta sa fie minima.

a. Normalizare
b. Selectie
c. Proiectie

A 79. Spunem ca o relatie este ....., daca si numai daca orice atribut al sau este atomic (indivizibil) si un tuplu nu contine atribute sau grupuri de atribute repetitive.

a. 1-normalizata
b. 2-normalizata
c. 3-normalizata

B 80. Spunem ca R este ..... daca si numai daca relatia este 1FN si atributele noncheie nu depind numai de o parte a cheii primare.

a. 1-normalizata
b. 2-normalizata
c. 3-normalizata

C 81. Spunem ca R este ..... daca si numai daca este 2FN si orice atribut noncheie nu depinde tranzitiv de cheia primara a lui R

a. 1-normalizata
b. 2-normalizata
c. 3-normalizata

D 82. Spunem ca R este ....-normalizata daca izoleaza relatiile independente multiple.

a. 1
b. 2
c. 3
d. 4

B 83. ..... FN presupune divizarea tabelelor aduse la a patra forma normala in scopul reducerii numarului de inregistrari (tuple) care trebuie introduse, modificate sau sterse la diferitele operatii de actualizare.

a. 2
b. 5
c. 4
d. 3

A 84. Algoritm pentru aducerea unei relatii in ..... FN:

1. Se inlocuiesc in relatie atributele compuse cu componentele lor.
2. Se creeaza cate o noua relatie pentru fiecare din grupurile repetitive.
3. Pentru fiecare din relatiile create la pasul 2 se introduce in schema cheia primara a relatiei din care a fost extras atributul repetitiv.
4. Pentru fiecare din relatiile create la pasul 2 se stabileste cheia primara care va fi formata din cheia introdusa la pasul 3, precum si din alte atribute ale acestei noi relatii.
5. Daca in noile relatii mai sunt inca atribute repetitive, se reia algoritmul. Daca nu, STOP.

a. 1
b. 2
c. 3
d. 4
e. 5

C 85. Algoritm pentru aducerea unei relatii in .... FN prin eliminarea dependentelor functionale tranzitive

1. Pentru fiecare dependenta functionala tranzitiva (atribute ce nu depind direct de cheia primara a relatiei R, A_0, A_1, … , A_p in care A_0 este cheie primara a lui R si pentru orice i=1,…,p, A_i depinde direct de A_{i-1}) se creeaza o noua relatie R' care contine atributele A_1,…,A_p si care are pe A_1 drept cheie primara.
2. Se elimina din R atributele A_2, A_3, … , A_p obtinand relatia R''
3. In noile relatii se repeta pasii 1 si 2 cat timp contin dependente tranzitive.

a. 1
b. 2
c. 3
d. 4
e. 5

D 86. Care din pachetele software enumerate nu este un sistem de prelucrare al bazelor de date?

a. Microsoft SQL Server
b. ACCESS
c. ORACLE
d. MICROSOFT POWERPOINT
e. INFORMIX

E 87. Specificati care varianta este incorecta

Componentele software ale sistemului de baze de date distribuite sunt:

a. SGBDL (Sistemul de gestiune al bazei de date locale) - sistem standard de gestiune a datelor care cuprinde propriul dictionar pentru datele locale
b. CC (Componenta de comunicatie) – responsabila cu legaturile in retea, cuprinde descreierea completa a nodurilor si a legaturilor retelei
c. DDG (Dictionarul de date globale) – detine informatii despre localizarea, disponibilitatea si modul de utilizare a datelor in BDD
d. SGBDD (Sistemul de gestiune al bazei de date distribuita) - interfata intre baza de date distribuita si utilizatori .
e. ASDD administrator de soft al datelor distribuite

A 88. Bazele de date ..... sunt multimi de baze de date autonome, slab corelate, manipulate de utilizator printr-un limbaj specific, care:
- Permit slabirea legaturii dintre bazele de date locale
- Furnizeaza un limbaj prin care:
  - se pot defini relatiile dintre diferite baze
  - se pot manipula mai multe baze concurent.

a. federale
b. distribuite mogen
c. paralele
d. distribuite eterogen

B 89. Pentru ca BDD sa fie usor prelucrabila, prin sistemul distribuit se pun la dispozitia acesteia o serie de independente. Locul unde sunt stocate datelele unei BDD nu-i este cunoscut utilizatorului, aceste informatii sunt pastrate in dictionarul datelor si sunt accesate de SGBDD pentru a stabili localizarea relatiilor ce apar in cererile utilizatorilor. Aceasta poarta numele de:
a. Independenta fragmentarii
b. Independenta localizarii
c. Independenta SGBD
d. Autonomia statiilor

C 90. Descrierea globala si unificata a tuturor datelor dintr-o BDD, independent de orice baza globala se numeste
a. schema externa globala
b. schema de alocare
c. schema globala .
d. schema conceptuala globala

B 91. Care varianta de raspuns nu este corecta?
Dictionarul datelor unei baze de date distribuite contine si informatii despre controlul semantic al datelor. Controlului semantic al datelor are o serie de functii:
a. functia de gestiune a vizualizarilor
b. functia de definire a datelor
c. functia de control a accesului autorizat
d. functia de control a integritatii semantice a datelor

D 92. In sistemul distribuit, evaluarea cererilor se realizeaza in patru faze. Una din fazele urmatoare nu este corecta. Specificati care:
a. faza de descompunere,
b. faza de localizare (transformarea unei cereri distribuite intr-o cerere echivalenta asupra fragmentelor)
c. faza de inregistrare
d. faza de executie

C 93. Care din variantele de mai jos nu face parte din gestiunea tranzactiilor distribuite?
a. Controlul concurentei
b. Gestiunea fiabilitatii
c. Evaluarea cererilor
d. Validarea tranzactiilor

C 94. *Controlul concurentei* impiedica producerea tranzactiilor distribuite neserializabile. El poate fi abordat din punct de vedere al stampilarii sau al blocarii. Care din afirmatiile de mai jos nu este corecta?
a. *Ştampilarea* - ordoneaza tranzactiile la lansarea lor in executie
b. *Ştampilarea* - are grija ca operatiile de acces la date sa se execute intr-o ordine predefinita.
c. In cadrul *stampilarii* - fiecare tranzactie are asociat un numar de ordine unic numit *stampila sau inel virtual*
d. *Blocarea* opreste tranzactiile care executa operatii conflicuale pe acelasi articol.
e. Accesul la articole prin protocolul *blocarii* se realizeaza cu primitivele: LOCK si UNLOCK.

B 95. Una din regulile de integritate ale MDOO nu este adevarata. Specificati care:
a. toate obiectele respecta protocolul specificat de definirile lor de clasa
b. obiectele nu sunt incapsulate
c. identificatorul obiectului asigura integritatea referirii la un obiect

B 96. Una din caracteristicile fundamentale obligatorii ale unui SGBDOO este gresita. Care anume?
a. trebuie sa fie un sistem orientat pe obiecte
b. trebuie sa indeplineasca conditiile unui SGBDD

C 97. Care varianta este gresita?
Una din componentele de baza ale unui SGBDOO este gestiunea obiectelor. Aceasta se realizeaza cu ajutorul:
a. administratorului de obiecte,
b. stocului rezident de obiecte
c. utilitarelor
d. serverului de obiecte

B 98. Care din urmatoarele trei variante este corecta?
In cadrul gestiunii obiectelor dintr-un SGBDOO, administratorul de obiecte (AO) asigura interfata dintre .
a. procesele interne si SGBDO
b. procesele externe si SGBDO
c. procesele externe si procesele interne

B 99. Care din urmatoarele trei variante este corecta?
Serverul de obiecte, asigura realizarea serviciilor de baza cum ar fi:
a. gestionarea tranzactiilor si gestionarea translatorului de cereri
b. gestionarea tranzactiilor si gestionarea stocului de obiecte
c. gestionarea stocului de obiecte si gestionarea translatorului de cereri

A 100. Prin ..... fragmentarii utilizatorul nu vede ca datele sunt fragmentate. Informatiile despre fragmentare sunt stocate in dictionarul datelor si utilizate de SGBDD pentru a traduce automat cererile referitoare la relatii in cereri referitoare la fragmente.

a. independenta
b. inelul
c. marca
d. arhitectura

B 101. O baza de date distribuita ..... este o multime de baze de date locale situate pe site-uri diferite, administrate de SGBD-uri identice.

a. eterogen
b. omogen

B 102. O baza de date distribuita ..... se obtine prin integrarea bazelor existente, administrate de SGBD-uri diferite si cu modele diferite, intr-o singura baza de date.
a. omogen
b. eterogen

D 103. Diferentele dintre un ..... si un SGBDD:
- Nu poate administra un dictionar global care contine informatii despre bazele de date distribuite;
- Suporta un limbaj pentru definirea dependentelor dintre diferite baze de date;
- Suporta un limbaj pentru definirea si manipularea bazelor de date din federatie

a. SGBDL
b. BDDE
c. BDOO
d. SGBD federal

C 104. Arhitectura unui ..... cuprinde:
- Un sistem global de gestiune a datelor;
- O interfata cu baza locala, care asigura:
  - Translatarea cererilor in limbajul de manipulare al datelor specific sistemului local;
  - Executia cererilor;

a. SGBDL
b. BDEE
c. SGBDF
d. BDOO

C 105. Bazele de date **.....** sunt BDDO in care statiile sunt nodurile unui calculator paralel.
Statiile comunica intre ele prin mesaje.
Programele sunt executate pe calculatorul gazda sau pe statii de lucru care comunica cu calculatorul paralel printr-o interfata specifica.

a. omogene
b. eterogene
c. paralele
d. federale

C 106. Independenta **.....** - pentru a asigura fiabilitatea, disponibilitatea si accesul performant la date, BDD-urile au copii ale informatiei, astfel daca o statie nu poate fi accesata (este neoperationala) la un moment dat exista o copie a fragmentului cautat.

a. localizarii
b. fragmentarii
c. dublurii
d. statiilor

A 107. Cele douasprezece reguli (sau obiective) ale lui Date (in 1990) pentru sistemele SGBDD au la baza ideea ca un sistem SGBD distribuit trebuie sa apara utilizatorului ca un sistem SGBD nedistribuit. Aceste reguli sunt inrudite cu cele douasprezece reguli ale lui Codd pentru sistemele relationale.
Principiul ..... Pentru utilizator, un sistem distribuit trebuie sa arate exact ca unul nedistribuit.

a. fundamental
b. autonomiei locale
c. independentei de locatie
d. operarii continue

B 108. Cele douasprezece reguli (sau obiective) ale lui Date (in 1990) pentru sistemele SGBDD au la baza ideea ca un sistem SGBD distribuit trebuie sa apara utilizatorului ca un sistem SGBD nedistribuit. Aceste reguli sunt inrudite cu cele douasprezece reguli ale lui Codd pentru sistemele relationale.
Regula ..... Site-urile dintr-un sistem distribuit trebuie sa fie autonome. In acest context, autonomia inseamna ca:

- Datele locale sunt detinute si gestionate local;
- Operatiile locale raman pur locale;
- Toate operatiile dintr-un anumit site sunt controlate de catre site-ul respectiv.

a. fundamentala
b. autonomiei locale
c. operarii continue
d. independentei de locatie

C 109. Cele douasprezece reguli (sau obiective) ale lui Date (in 1990) pentru sistemele SGBDD au la baza ideea ca un sistem SGBD distribuit trebuie sa apara utilizatorului ca un sistem SGBD nedistribuit. Aceste reguli sunt inrudite cu cele douasprezece reguli ale lui Codd pentru sistemele relationale.
Regula **.....**. Ideal este ca niciodata sa nu fie nevoie de o oprire planificata a sistemului pentru operatii cum ar fi:

- Adaugarea sau eliminarea unui site din sistem;
- Crearea si stergerea dinamica a fragmentelor dintr-unul sau mai multe site-uri.

a. fundamentala
b. autonomiei locale
c. operarii continue
d. independentei de locatie

D 110. Cele douasprezece reguli (sau obiective) ale lui Date (in 1990) pentru sistemele SGBDD au la baza ideea ca un sistem SGBD distribuit trebuie sa apara utilizatorului ca un sistem SGBD nedistribuit. Aceste reguli sunt inrudite cu cele douasprezece reguli ale lui Codd pentru sistemele relationale.
Regula ..... Utilizatorul trebuie sa aiba posibilitatea de a accesa datele, indiferent de modul in care sunt fragmentate.

a. fundamentala
b. autonomiei locale
c. operarii continue
d. independentei de fragmentare

D 111. Cele douasprezece reguli (sau obiective) ale lui Date (in 1990) pentru sistemele SGBDD au la baza ideea ca un sistem SGBD distribuit trebuie sa apara utilizatorului ca un sistem SGBD nedistribuit si sunt inrudite cu cele douasprezece reguli ale lui Codd pentru sistemele relationale.
Una din regulile ideale este **.....**. Trebuie sa fie posibil ca sistemul SGBDD sa poata fi rulat pe o diversitate de platforme hardware.

a. independentei de fragmentare
b. independentei de reproducere
c. independentei de retea
d. independentei de hardware

D 112. Cele douasprezece reguli (sau obiective) ale lui Date (in 1990) pentru sistemele SGBDD au la baza ideea ca un sistem SGBD distribuit trebuie sa apara utilizatorului ca un sistem SGBD nedistribuit si sunt inrudite cu cele douasprezece reguli ale lui Codd pentru sistemele relationale.
Una din regulile ideale este regula **.....** care afirma ca trebuie sa fie posibil sa se ruleze sistemul SGBDD pe o diversitate de sisteme de operare.

a. independentei de retea
b. independentei de fragmentare
c. independentei de hardware
d. independentei de sistemul de operare

D 113. Cele douasprezece reguli (sau obiective) ale lui Date (in 1990) pentru sistemele SGBDD au la baza ideea ca un sistem SGBD distribuit trebuie sa apara utilizatorului ca un sistem SGBD nedistribuit si sunt inrudite cu cele douasprezece reguli ale lui Codd pentru sistemele relationale.
Una din regulile ideale este regula ..... care spune ca trebuie sa fie posibil sa se ruleze sistemul SGBDD pe o diversitate de retele de comunicatie separate.

a. independentei de hardware
b. independentei de fragmentare
c. independentei de sistemul de operare
d. independentei de retea

Name: _____ **Class:** _____ **Date:** _____ **ID: A**

Subiecte modele si limbaje de simulare licenta informatica 4 ani

True/False
Indicate whether the sentence or statement is true or false.

T 1. Pentru validarea generatorilor prin testul hi patrat intervine histograma frecventelor absolute?

F 2. Testul hi patrat este singura modalitate de a valida generatori de variabile aleatoare?

F 3. Fie $X \sim\!\!\rightarrow H(N,p,n)$ atunci $P(X=a)=\dfrac{C_A^a C_B^{n-a}}{C_N^n}$ unde $0<p<1$, $A=\{Np\}$, ($\{x\}$, $x\in R$ este cel mai apropiat intreg de x), $n<N$, $B=N-A$, $0\le a\le n$. Este adevarat ca $E(X)=(n-1)p$, $Var(X)=np(1-p)\dfrac{N-n}{N-1}$?

T 4. Daca $X \sim\!\!\rightarrow Geom(p)$ este adevarat ca $E(X)=\dfrac{1-p}{p}$ si $Var(X)=\dfrac{1-p}{p^2}$?

T 5. Daca $X \sim\!\!\rightarrow Pascal(k,p)$ atunci $P(X=\alpha)=C_{\alpha+k-1}^{k-1}p^k(1-p)^\alpha$, $\alpha=0,1,...$ Este adevarat ca $E(X)=\dfrac{k(1-p)}{p}$, $Var(X)=\dfrac{k(1-p)}{p^2}$?

F 6. Daca $X \sim\!\!\rightarrow Binom(n,p)$, $0<p<1$, $n\in N^+$ atunci $P(X=\alpha)=C_n^\alpha p^\alpha(1-p)^{n-\alpha}$, $\alpha\in 0,1,\ldots,n$. Este adevarat ca $E(X)=np$ si $Var(X)=np^2(1-p)$?

F 7. Daca $X \sim\!\!\rightarrow Bernoulli(p)$, $0<p<1$, este adevarat ca functia de repartitie a lui X are pentru $x\in[0,1)$ valoarea $F(x)=p$?

F 8. Daca $X \sim\!\!\rightarrow Bernoulli(p)$, $0<p<1$, este adevarat ca $E(X)=1-p$ si $Var(X)=p(1-p)$

F 9. In definitia repartitiei $Beta(a,b)$ intervine functia lui Euler de speta I-a $B(a,b)$ data prin $B(a,b)=\int_0^1 x^{a-1}(1-x)^{b-1}dx$. Este adevarat ca $B(a,b)=\dfrac{\Gamma(a)\Gamma(b)}{\Gamma(a+b-1)}$, unde $\Gamma(a)=\int_0^\infty x^{a-1}e^{-x}dx$

T 10. Este stabila familia de variabile aleatoare repartizate Pascal ?

F 11. Daca infasuram $X \sim\!\!\rightarrow f(x) \sim\!\!\rightarrow Gamma(0,1,\upsilon)$, $\upsilon>1$ cu densitatea $h(x) \sim\!\!\rightarrow Exp\left(\dfrac{1}{\upsilon}\right)$ gasim un algoritm de generare a variabilei X care are probabilitatea de acceptare $p_a\approx\sqrt{\dfrac{2\pi}{e^2(y-1)}}$. Stiind ca $p_a=\dfrac{1}{\alpha}$, unde $\alpha=\dfrac{\upsilon^\upsilon e^{1-\upsilon}}{\Gamma(\upsilon)}$ si ca pentru $\Gamma(\upsilon)$ cu $\upsilon\to\infty$ s-a luat aproximarea lui Stirling $\Gamma(\upsilon)\approx(\upsilon-1)^{\upsilon-1}e^{-(\upsilon-1)}\sqrt{2\pi(\upsilon-1)}$, este adevarata relatia obtinuta pentru p_a ?

T 12. Este adevarat ca variabila hi patrat centrata cu f grade de libertate este de tip $Gamma(0,\dfrac{1}{2},\dfrac{f}{2})$

T 13. Este stabila familia variabilelor aleatoare Gamma standard?

T 14. Este stabila familia variabilelor aleatoare normale standard?

F 15. Se poate simula repartitia maximului folosind o transformare de tipul $X = \min(U_1, \ldots, U_n)$, unde U_i este variabila uniforma 0-1 pentru orice $i = 1, \ldots, n$?

F 16. Daca generatorul G se obtine prin amestecarea generatorilor G_1 si G_2 care au perioadele $T(G_1)$ si respectiv $T(G_2)$, este adevarat ca $T(G) = \text{cmmdc}(T(G_1), T(G_2))$?

T 17. Se bazeaza generatorul Fibonacci decalat $(k: x_0, x_1, \ldots, x_k, j, m)$ pe relatia: $X_n = (X_{n-j} + X_{n-k}) \bmod m$?

T 18. Un stoc este o resursa de orice fel care are o valoare economica, caracterizata prin ***intrari*** si ***iesiri***

T 19. De regula, intrarea in stoc se realizeaza in cantitati mari (numite ***comenzi***) care se introduc in stoc la intervale de timp numite ***cicluri de reaprovizionare***.

F 20. De regula, intrarea in stoc se realizeaza in cantitati mari (numite ***comenzi***) care se introduc in stoc la intervale de timp numite ***inventare***.

T 21. *Modelul clasic al lotului economic* este cunoscut si sub numele de ***modelul lui Wilson***

F 22. *Modelul clasic al lotului economic* este cunoscut si sub numele de ***modelul clasic al lipsei de stoc.***

T 23. *Procesul stochastic discret $N(t)$, cu cresteri independente, se numeste proces de nastere si moarte, daca satisface urmatoarele proprietati:*

$$P\big([N(t+\Delta t) = n+1][N(t) = n]\big) = \lambda_n \Delta t + o(\Delta t)$$
$$P\big([N(t+\Delta t) = n-1][N(t) = n]\big) = \mu_n \Delta t + o(\Delta t)$$
$$P\big([N(t+\Delta t) = n \pm i][N(t) = n]\big) = o(\Delta t),\ i > 1$$

unde $P(A|B)$ inseamna probabilitatea lui A conditionata de B, constantele $\{\lambda_n, n \geq 0\}$, $\{\mu_n, n \geq 1\}$ sunt siruri de numere pozitive date, iar $o(\Delta t)$ este un element al unei clase de functii ce satisfac conditiile

$$\lim_{\Delta t \to 0} o(\Delta t) = 0,\ \lim_{\Delta t \to 0} \frac{o(\Delta t)}{\Delta t} = 0$$

F 24. *Procesul stochastic discret $N(t)$, cu cresteri independente, se numeste proces Marcov simplu, daca satisface urmatoarele proprietati:*

$$P\big([N(t+\Delta t) = n+1][N(t) = n]\big) = \lambda_n \Delta t + o(\Delta t)$$
$$P\big([N(t+\Delta t) = n-1][N(t) = n]\big) = \mu_n \Delta t + o(\Delta t)$$
$$P\big([N(t+\Delta t) = n \pm i][N(t) = n]\big) = o(\Delta t),\ i > 1$$

unde $P(A|B)$ inseamna probabilitatea lui A conditionata de B, constantele $\{\lambda_n, n \geq 0\}$, $\{\mu_n, n \geq 1\}$ sunt siruri de numere pozitive date, iar $o(\Delta t)$ este un element al unei clase de functii ce satisfac conditiile

$$\lim_{\Delta t \to 0} o(\Delta t) = 0,\ \lim_{\Delta t \to 0} \frac{o(\Delta t)}{\Delta t} = 0$$

T 25. Procesul de nastere si moarte este ***stationar*** daca $P_n(t) = p_n = const.$ adica repartitia sa nu depinde de timp.

F 26. Procesul de nastere si moarte este ***nestationar*** daca $P_n(t) = p_n = const.$ adica repartitia sa nu depinde de timp.

T 27. In modelul cu ceas variabil, ceasul variabil al simularii nu apare explicit (el este ***implicit***), in sensul ca de fapt pentru alegerea evenimentului ce trebuie prelucrat se foloseste ***regula primului eveniment urmator***, care este o consecinta a tehnicii bazate pe ceasul cu crestere variabila.

T 28. Procesele si lanturile Markov sunt acelea care satisfac ***proprietatea lui Markov***.

T 29. Tehnica urmatoare **"** Sa presupunem ca se cunoaste repartitia initiala $\pi = (\pi_1, \pi_2, ..., \pi_m)'$ si matricea probabilitatilor de tranzitie $P = \|p_{ij}\|$. Atunci starea initiala $I = i$ se simuleaza ca variabila discreta" permite simularea unui lant Markov.

$$I : \begin{pmatrix} 1, & 2, & ..., & m \\ \pi_1, & \pi_2, & ..., & \pi_m \end{pmatrix}$$

Daca lantul se afla in starea i atunci starea aleatoare urmatoare in care trece lantul se simuleaza ca variabila discreta

$$J : \begin{pmatrix} 1, & 2, & ..., & m \\ p_{i1}, & p_{i2}, & ..., & p_{im} \end{pmatrix}$$

F 30. Tehnica urmatoare " Sa presupunem ca se cunoaste repartitia initiala $\pi = (\pi_1, \pi_2, ..., \pi_m)'$ si matricea probabilitatilor de tranzitie $P = \|p_{ij}\|$. Atunci starea initiala $I = i$ se simuleaza ca variabila discreta" permite simularea unui proces gausian stationar.

$$I : \begin{pmatrix} 1, & 2, & ..., & m \\ \pi_1, & \pi_2, & ..., & \pi_m \end{pmatrix}$$

Daca lantul se afla in starea i atunci starea aleatoare urmatoare in care trece lantul se simuleaza ca variabila discreta

$$J : \begin{pmatrix} 1, & 2, & ..., & m \\ p_{i1}, & p_{i2}, & ..., & p_{im} \end{pmatrix}$$

T 31. Daca o stare i a unui lant omogen are proprietatea ca exista un j, $j \neq i$ astfel incat $p_{ij} > 0$, atunci starea i este ***stare de tranzitie***.

F 32. Daca o stare i a unui lant omogen are proprietatea ca exista un j, $j \neq i$ astfel incat $p_{ij} > 0$, atunci starea i este ***stare absorbanta***.

T 33. Daca o stare i a unui lant omogen are proprietatea ca exista un j, $j \neq i$ astfel incat $p_{ii} = 1$ atunci starea i este ***stare absorbanta***.

F 34. Daca o stare i a unui lant omogen are proprietatea ca exista un j, $j \neq i$ astfel incat $p_{ij} \in (0,1)$, atunci starea i este ***stare absorbanta***.

T 35. Sub denumirea de ***metode Monte Carlo*** sunt cuprinse o serie de tehnici de rezolvare a diverse probleme utilizand ***numere aleatoare, variabile aleatoare*** sau ***procese stochastice*** simulate cu calculatorul.

T 36. Ecuatia Poisson este un caz particular de ecuatie de tip eliptic.

T 37. În diverse modele de stocare se pot da costurile h, s si/sau d, se da rata cererii r si timpul de avans L si se cer q, P optime. O multime de elemente ce definesc un mecanism de aprovizionare se spune ca determina o ***politica de reaprovizionare***.

T 38. In diverse modele de stocare se pot da costurile h, s si/sau d, se da rata cererii r si timpul de avans L si se cer q, P optime. Cand r, L nu sunt aleatoare, modelul se numeste ***determinist***.

T 39. In diverse modele de stocare se pot da costurile h, s si/sau d, se da rata cererii r si timpul de avans L si se cer q, P optime. Cand cel putin una din variabilele r, L este aleatoare, modelul este ***stochastic***.

F 40. In diverse modele de stocare se pot da costurile h, s si/sau d, se da rata cererii r si timpul de avans L si se cer q, P optime. Cand r, L nu sunt aleatoare, modelul se numeste ***stochastic***.

F 41. In diverse modele de stocare se pot da costurile h, s si/sau d, se da rata cererii r si timpul de avans L si se cer q, P optime. In cazul cand cel putin una din variabilele *r, L* este aleatoare, modelul este ***determinist.***.

T 42. Modelul:

$$Exp(\lambda)/\,Exp(\mu)/1:(\infty, FIFO)$$

Este un exemplu de model matematic de asteptare cu o statie de serviciu.

T 43. Modelul:

$$Exp(\lambda)/\,Exp(\mu)/\,N(paralel):(\infty, FIFO)$$

este un model matematic de asteptare cu N statii paralele presupuse identice (adica au aceeasi repartitie pentru ST-ul fiecareia), coada poate fi infinita, iar disciplina este standard (***primul venit primul servit***).

Multiple Choice

Identify the letter of the choice that best completes the statement or answers the question.

C 44. Dupa natura variabilelor folosite in modelele matematice acestea se clasifica in:
- a. continue/discrete, statice/dinamice, cu o componenta / cu mai multe componente in paralel
- b. deterministe/stochastice, continue/discrete, cu o componenta / cu mai multe componente in serie
- c. statice/dinamice, deterministe/stochastice, continue/discrete

B 45. Dupa structura (topologia componentelor in care se descompun) modelele matematice se clasifica in:
- a. cu o componenta / cu mai multe componente in retea, statice/dinamice, continue/discrete
- b. cu o componenta / cu mai multe componente in paralel, serie, retea
- c. deterministe/stochastice, cu o componenta / cu mai multe componente in paralel

C 46. Conceptele de baza ale unui model de simulare sunt:
- a. modelul matematic si agenda simularii
- b. ceasul simularii si modelul matematic
- c. ceasul simularii si agenda simularii

A 47. In modelele de simulare, ceasul simularii este de “n” tipuri, unde:
- a. n=2
- b. n=3
- c. n=4

A 48. In modelele de simulare, agenda simularii se compune din n parti, unde:
a. n=2
b. n=3
c. n=4

A 49. In modelele de simulare, ceasul simularii poate fi:
a. cu crestere variabila sau cu crestere constanta
b. doar cu crestere variabila
c. doar cu crestere constanta

B 50. Nivelele de reprezentare a sistemelor sunt:
a. comportare, ierarhica, modulara
b. modulara, structura de stare, comportare
c. structura de stare, ierarhica, comportare

C 51. In GPSS exista m tipuri de entitati grupate in n categorii, unde perechea (n,m) este:
a. (3, 16)
b. (4,10)
c. (5,16)
d. (6,15)

A 52. In GPSS exista m tipuri de blocuri, care sunt entitati active, unde m este:
a. 6
b. 5
c. 4
d. 3

D 53. Algoritmul:

for i=1 **to** *n* **do begin**
genereaza $U := \text{random};$
calculeaza $X=F^{-1}(U),$

unde F este functia de repartitie a variabilei aleatoare *X*, care este functie inversabila, simuleaza o selectie de volum *n* asupra lui *X* prin metoda:
a. compunerii discrete
b. compunerii continue
c. congruentiala liniara
d. inversa

B 54. Algoritmul:

intrare: vectorul $T[m]$ cu $T(i)=\sum_{j=1}^{i} p_j, i=1,\ldots,m$ si vectorul $a[m]$

genereaza U uniform $0-1$ cu $U := random;$
$i := 1;$ while $U > F(i)$ do $i := i+1;$
ia $X = a(i),$

unde F este functia de repartitie a variabilei aleatoare discrete $X:\begin{pmatrix} a_1 & a_2 & \ldots & a_m \\ p_1 & p_2 & \ldots & p_m \end{pmatrix}, \sum_{i=1}^{m} p_i = 1,$

simuleaza X prin metoda:
a. congruentiala liniara
b. inversa
c. amestecarii

D 55. Algoritmul:

genereaza indicele j cu repartitia $J:\begin{pmatrix}1 & 2 & \dots & m\\ p_1 & p_2 & \dots & p_m\end{pmatrix}, \sum_{i=1}^{m} p_i = 1;$

genereaza X_j avand functia de repartitie $F_j(x)$;

ia $X := X_j$,

unde X este variabila aleatoare cu functia de repartitie $F(x)$, iar X_j este variabila aleatoare cu functia de repartie $F_j(x)$ pentru orice $j = 1,...,m$ si in plus $F(x) = \sum_{j=1}^{m} p_j F_j(x)$,

simuleaza X prin metoda:

a. compunerii continue
b. inversa
c. congruentiala liniara
d. amestecarii discrete

A 56. Algoritmul:

genereaza Y cu functia de repartitie continua $H(y)$;

genereaza Z_Y cu functia de repartitie $G(x,Y)$;

ia $X := Z_Y$,

unde X este variabila aleatoare cu functia de repartitie $F(x)$, $\{G(x,Y)\}_{Y\in R}$ este o familiile de variabile aleatoare continue in care Y are functia de repartitie $H(y)$ si in plus $F(x) = \int_R G(x,y)dH(y)$ simuleaza X prin metoda:

a. mixturii continue
b. inversa
c. amestecarii discrete

C 57. Algoritmul:

repeat

genereaza $Z_0 \sim\!\!\rightarrow G_0(x)$; ia $Z^* = Z_0$, $K := 1$; genereaza $Z_1 \sim\!\!\rightarrow G(x)$;

while $Z_0 \geq Z_1$ **do begin**

$K = K + 1$; $Z_0 := Z_1$;

genereaza $Z_1 \sim\!\!\rightarrow G(x)$;

end;

until $K \bmod 2 = 1$;

ia $X := Z^*$,

unde $G_0(x)$ si $G(x)$ sunt functii de repartitie, simuleaza variabila aleatoare X prin metoda respingerii, fiind bazat pe:

a. teorema infasuratoarei
b. algoritmul general de respingere
c. teorema sirului descendent

B 58. Repartitia χ^2 centrata cu f grade de libertate se defineste pornid de la f variabile aleatoare repartizate:

a. $N(m_i f)$, $i=1,...,f$
b. $N(0,1)$
c. $Exp(f)$
d. $Gamma(0,1,f)$

C 59. Repartitia χ^2 necentrata cu f grade de libertate si cu parametrul de excentricitate nenul se defineste pornind de la f variabile aleatoare repartizate:

a. $Gamma(0, f, 1)$
b. $N(m_i f)$, $i=1,...,f$
c. $N(m_i, 1)$, $i=1,...,f$
d. $N(0,1)$

C 60. Repartitia Student cu *f* grade de libertate se defineste pornind de la doua variabile aleatoare independente *Z* si *X*, unde perechea (*Z,X*) are repartitia:
a. (*N*(0,*f*), *Exp*(*f*))
b. (*Exp*(*f*), *N*(0,1))
c. (*N*(0,1), hi patrat centrat cu *f* grade de libertate)

C 61. Repartitia Student cu *f* grade de libertate si cu parametrul de excentricitate nenul se defineste pornind de la variabilele aleatoare independente Z si X, in care perechea (*Z,X*) are repartitia:
a. (*N*(0,*f*), hi patrat necentrat)
b. (*Exp*(*f*), hi patrat centrat)
c. (*N*(0,1), hi patrat necentrat cu *f* grade de libertate)

D 62. Repartitia Snedecor centrata cu f_1, f_2 grade de libertate se defineste pornind de la doua variabile aleatoare independente X_1 si X_2 in care perechea (X_1, X_2) are repartitia:
a. (*Exp*(*f1*), *Exp*(f_2))
b. (*N*(0,f_1), *Exp*(f_2))
c. *Exp*(f_1), hi patrat centrat cu f_2 grade de libertate)
d. (hi patrat centrat cu f_1 grade de libertate, hi patrat centrat cu f_2 grade de libertate)

C 63. Algoritmul:

Intrare m, s^2; calculeaza μ, σ;
genereaza $Z \sim\!\!\rightarrow N(0,1)$;
calculeaza $X = \mu + Z\sigma$;
ia $Y = e^X$,

simuleaza repartitia:
a. *N*(0,1)
b. *N*(m, s^2)
c. $LN(\mu,\sigma)$
d. Gamma(0, m, s^2)

B 64. Daca in teorema sirului descendent se considera $Z_0 = U_0$, $Z_i = U_1$, $i \geq 1$ unde U_0, U_1 sunt uniforme $0-1$ si daca notam cu N numarul aleator de subsiruri descendente respinse pana cand se accepta un subsir, atunci $X=N+Z_0$, (Z_0 fiind cel din ultimul sir descendent acceptat) urmeaza repartitia:
a. *N*(0,1)
b. *Exp*(1)
c. *Gamma*(0,1,1)

A 65. Agoritmul:

initializeaza $N := 0$;

repeat

genereaza U_0, U_1 uniforme $0-1$ si independente;

ia $U^* := U_0$; $K := 1$;

while $U_0 \geq U_1$ **do begin**

$K := K + 1$; $U_0 := U_1$;

genereaza U_1 uniform $0-1$;

end;

if $K \bmod 2 = 0$ **then** $N := N + 1$;

until $K \bmod 2 = 1$;

ia $Z := N + U^*$,

simuleaza prin metoda respingerii repartitia:

a. *Exp*(1)
b. *N*(0,1)
c. *LN*(0,1)

D 66. Repartitia *Beta*(*a,b*) cu *a,b*>0 se defineste pornind de la doua variabile X_1 si X_2 independente pentru care perechea (X_1,X_2) are repartitia:

a. (*N*(0,*a*), *Exp*(*b*))
b. (*Gamma*(0,1,*a*), *N*(*a,b*))
c. (*Exp*(*a*), *Exp*(*b*))
d. (*Gamma*(0,1,*a*), *Gamma*(0,1,*b*))

D 67. Algoritmul:

intrare a,b∈(0,1)

repeat

genereaza U_1=random, U_2=random independente;

ia $V = U_1^{\frac{1}{a}}$, $T = U_2^{\frac{1}{b}}$;

until $V + T < 1$;

calculeaza $X = \frac{V}{V+T}$,

simuleaza repartitia:

a. *N*(*a,b*)
b. *LN*(*a,b*)
c. *Gamma*(0,*a,b*)
d. *Beta*(*a,b*)

C 68. Algoritmul:

repeat

genereaza U random;

Genereaza $Y \sim\!\!\rightarrow Exp(1)$;

until $U \leq e^{-\frac{Y^2}{2}+Y-0.5}$;

ia $X_1 := Y$;

genereaza U random;

if $U \leq 0.5$ **then** $s := 1$ **else** $s := -1$;

ia $X = sX_1$,

simuleaza prin compunere-respingere repartitia:

a. *Exp*(1)
b. *Gamma*(0,1,s)
c. *N*(0,1)

D 69. Teorema lui Box si Muller conduce la o metoda de simulare a variabilei $N(0,1)$ numita metoda:
a. infasuraoarei
b. amestecarii
c. compunerii-respingerii
d. polara

C 70. Algoritmul:

genereaza U = random;
if $U > p$ **then** Z=0 **else** Z=1,

simuleaza repartitia:
a. *Geom*(*p*)
b. *Exp*(*p*)
c. *Bernoulli*(*p*)
d. *N*(0,*p*)
e. *Binom*(0,*p*)

D 71. Algoritmul:

citeste *n,p* si calculeaza *q=1-p*
genereaza $W \sim\!\!\rightarrow N(0,1)$;
calculeaza $X := \{np + W\sqrt{npq}\}$,

unde $\{E\}$ este cel mai apropiat intreg de E, simuleaza repartitia
a. *N*(*n,p*)
b. *Weibull*(0,*n,p*)
c. *Pascal*(*n,p*)
d. *Binom*(*n,p*)

C 72. Repartitia binomiala *Binom*(*n,p*) se defineste pornind de la repartitia:
a. *N*(*n,p*)
b. *Pascal*(*n,p*)
c. *Bernoulli*(*p*)
d. *Weibull*(0,*n,p*)

B 73. Algoritmul de generare a repartitiei *Binom*(*n,p*), pentru n mare se bazeaza pe:
a. teorema infasuratoarei
b. teorema limita centrala
c. teorema sirului descendent

A 74. Repartitia *Pascal*(*k,p*) se defineste pornind de la repartitia:
a. *Bernoulli*(*p*)
b. *N*(*k,p*)
c. *Binom*(*k,p*)
d. *Exp*(*p*)

B 75. Repartitia binomiala cu exponent negativ se mai numeste repartitia:
a. Exponentiala
b. Pascal
c. Bernoulli
d. Geometrica

C 76. Algoritmul:

intrare $k \in N^+$ si p cu $0 < p < 1$; ia $X = 0$; $j := 0$;
repeat
genereaza $U :=$ random;
if $U < p$ **then** $j := j + 1$ **else** $X := X + 1$;
until $j = k$,

simuleaza variabila aleatoare X cu repartitia:
a. *Binom*(*k,p*)
b. *N*(*k,p*)
c. *Pascal*(*k,p*)
d. *Geom*(*p*)

C 77. Repartitia geometrica se defineste pornind de la repartitia:
a. *Normala*
b. *Binomiala*
c. *Bernoulli*

C 78. Se mai numeste repartitia evenimentelor rare, repartitia:
a. Geometrica
b. Bernoulli
c. Poisson
d. Pascal

C 79. Algoritmul:

intrare A, B, n, $N = A + B$;

calculeaza $p = \frac{A}{N}$; si ia $j := 0$, $X := 0$;

repeat

genereaza $U :=$ random; si ia $j := j + 1$;

if $U < p$ **then begin**

$X := X + 1$; $S := 1$;

end else $S := 0$**;**

calculeaza $N := N - 1$, A: = A - S, $p := \frac{A}{N}$;

until $j = n$,

simuleaza repartitia:

a. *N*(*n*,*p*)
c. *Hipergeometrica*
b. *Pascal*(*n*,*p*)
d. *Binomiala*

C 80. Repartitia *Erlang*(*k*) se defineste pornind de la repartitia:

a. *Poisson*(*k*)
c. *Exp*(1)
b. *Exp*(*k*)
d. *Geom*(*k*)

D 81. Procesul stochastic $\{X_t\}_{t \in T}$ pentru care exista si sunt finite

$m_t = E[X_t]$, $\sigma_{st} = Cov[X_s, X_t]$, s < t si $m_t = m = constant, \sigma_{st} = \sigma(t - s)$

se numeste :

a. stationar tare;
c. cu timp discret;
b. cu timp continuu;
d. stationar slab

D 82. Numarul metodelor necesare reducerii dispersiei estimatorului secundar pentru calculul integralelor prin metoda Monte Carlo bruta este:

a. 3 b. 4 c. 5 d. 6

A 83. O familie de variabile aleatoare, $\{X_t\}_{t \in T}$, $T \subset R$, deprinzand de parametrul real t (presupus a fi ***timpul***) se numeste **.....**.

a. proces stochastic
c. proces cu stari discrete
b. lant
d. proces cu stari continue

B 84. *Procesul stochastic* $\{X_t\}_{t \in T}$ *se numeste* **.....** *daca* $\forall n, h \in R$, $t_1 < t_2 < ... < t_n$ *avem* $F_{t_1+h, t_2+h, ..., t_n+h}(x_1, x_2, ..., x_n) = F_{t_1, t_2, ..., t_n}(x_1, x_2, ..., x_n)$

a. stationat
c. stationar slab
b. stationar tare

C 85. *Procesul* $\{X_t\}_{t \in T}$ *discret, este un proces ..... daca satisface proprietatea* $P(X_{t_n} = x_n | X_{t_{n-1}} = x_{n-1}, ..., X_{t_1} = x_1) = P(X_{t_n} = x_n | X_{t_{n-1}} = x_{n-1})$

adica procesul ***nu are memorie***

a. stochastic
c. *Markov*
b. slab stationar

C 86. Probabilitatile de tranzitie santisfac ***relatia lui .....***, adica

$$P_{ij}(s,t)=\sum_{k\in S}\sum_{s\le r\le t}P_{ik}(s,r)P_{kj}(rt)$$

$P_{ij}(s,t)$ definesc ***matrice de probabilitati de tranzitie*** care au proprietatea

$$\sum_{j\in S}P_{ij}(s,t)=1$$

a. Markov
b. Dirichlet
c. Chapman-Kolmogorov

A 87. Fie procesul stationar care are ***functia de .....***

$$\phi(t)=Cov(X_s,X_{s+t}),\ t=0,1,2,...,\ \phi(0)=Var(X_t)=\sigma^2$$

si fara a pierde generalitatea consideram $m_t=m=0$. Daca notam $\rho(X_s,X_{t+s})=Corr(X_s,X_{t+s})$ rezulta ca $\rho(X_s,X_{t+s})=\rho(t),\ t=1,2,...$

a. autocovarianta
b. repartitie
c. autocorelatie

B 88.

Fie procesul stationar care are ***functia de autocovarianta***

$$\phi(t)=Cov(X_s,X_{s+t}),\ t=0,1,2,...,\ \phi(0)=Var(X_t)=\sigma^2$$

si fara a pierde generalitatea consideram $m_t=m=0$. Daca notam $\rho(X_s,X_{t+s})=Corr(X_s,X_{t+s})$ rezulta ca $\rho(X_s,X_{t+s})=\rho(t),\ t=1,2,...$ Functia $\rho(t)$ este ***functia de ..... a procesului***.

a. repartitie
b. autocorelatie

B 89.

Procesul gausian cu funtie de autocorelatie .....

$$\rho(t)=\frac{\phi(t)}{\phi(0)}=\theta^t,\ t=0,1,2,...,\ 0<\theta<1$$

si σ^2 dispersia sa, se simuleaza cu formula de recurenta

$$Z_0\sim\to N(0,1),\ Z_{t+1}=\theta Z_t+\sqrt{1-\theta^2}\varepsilon_t,\ t=1,2,...,\ X_{t+1}=\sigma Z_{t+1}$$

unde $\varepsilon_t,\ t=1,2,...$ sunt variabile $N(0,1)$ independente si independente de Z_t.

a. liniara
b. exponentiala

B 90. Procesul ..... $\{X_t\}_{t\in T}$ este descris de urmatoarele relatii de recurenta

$$X_0=\sqrt{\frac{1-\theta}{1+\theta}}V_0,\ X_t=\theta X_{t-1}+(1-\theta)V_t,\ 0<\theta<1$$

unde $V_t,\ t=0,1,2,...$ sunt variable aleatoare independente si identic repartizate avand proprietatile $E[V_t]=0$, $Var(V_t)=\sigma^2$.

a. gausian
b. zgomotului alb
c. gausian stationar

A 91. Procesul ..... este un caz particular de proces de nastere si deces, mai precis este un proces de nastere pur cu intensitatea $\lambda_n = \lambda$, a carui repartitie satisface ecuatiile diferentiale

$$P_0'(t) = -\lambda P_0(t),\ P_n'(t) = -\lambda P_n(t) + \lambda P_{n-1}(t),\ \geq 1.$$

a. Poisson
b. procesul de zgomot alb
c. procesul de zgomot alb pur

A 92. Procesul avand repartitia $P_n(t) = \frac{(\lambda t)^n}{n!} e^{-\lambda t}$ se numeste ***proces Poisson .....***

a. omogen
b. neomogen
c. simplu

A 93. Procesul avand repartitia $P_n(t) = \frac{(\Lambda(t))^n}{n!} e^{-\Lambda(t)}$, $\Lambda(t) = \int_0^t \lambda(u)du$ se numeste ***proces Poisson*** ***.....*** .

a. neomogen
b. omogen
c. simplu

A 94. Sub denumirea de ***metode .....*** sunt cuprinse o serie de tehnici de rezolvare a diverse probleme utilizand ***numere aleatoare, variabile aleatoare*** sau ***procese stochastice*** simulate cu calculatorul.

a. Monte Carlo
b. Dirichlet
c. legea numerelor mari
d. Poisson

A 95.

Metoda urmatoare:
"Sa presupunem ca alegem o functie φ (ca in cazul metodei variabilei de control) astfel incat $\theta = \int_0^1 \varphi(u)du + \int_0^1 (f(u) - \varphi(u))du$ unde integrala $\mu = \int_0^1 \varphi(u)du$ se poate calcula exact si usor. Daca in plus se alege φ astfel incat

$$Var(\varphi(U)) - 2Cov(\varphi(U), f(U)) < 0$$

atunci, de asemenea se poate realiza reducerea dispersiei deoarece

$$\begin{aligned} Var(\varphi^*(U)) &= Var(f(U) - \varphi(U)) = \\ &= Var(f(U)) + Var(\varphi(U)) - 2Cov(f(U) - \varphi(U)) < \\ &< Var(f(U)) \end{aligned}$$

se numeste metoda variabilelor .....

a. corelate
b. antitetice
c. de control

A 96. Metoda urmatoare:

"Fie integrala $\theta = \int_0^1 f(x)dx$ care se scrie $\theta = E[\tau] = E[f(U)]$, sa alegem un alt estimator primar $f^*(U)$, $\theta = E[f^*(U)]$ si sa consideram estimatorul primar $\varphi = f^*(1-U)$, $E[\varphi(U)] = E[f^*(1-U)] = \theta$. Atunci, putem considera estimatorul primar $\psi = \frac{1}{2}(\tau + \varphi)$ pentru care avem $Var(\psi) = \frac{1}{4}(Var(\tau) + Var(\varphi) + 2Cov(\tau + \varphi))$ si $Cov(\tau + \varphi) < 0$. Daca se alege f^* astfel incat $Var(\psi) < Var(\tau)$ (1). Atunci estimatorul ψ realizeaza reducerea dispersiei. Uneori puntem alege simplu $\varphi = f(1-U)$ care satisface (1)." Se numeste metoda variabilelor .....

a. antitetice b. corelate c. de control

A 97. Fie ecuatia cu derivate partiale de ordinul doi

$$\beta_{11}\frac{\partial^2 V}{\partial x^2} + 2\beta_{12}\frac{\partial^2 V}{\partial x \partial y} + \beta_{22}\frac{\partial^2 V}{\partial y} + 2\alpha_1\frac{\partial V}{\partial x} + 2\alpha_2\frac{\partial V}{\partial y} = F(x,y) \qquad (1)$$

cu conditia

$$V(x,y) = \varphi(x,y),\ (x,y) \in C \qquad (2)$$

unde $V(x,y)$ este o functie necunoscuta pe domeniul marginit $D \subset R^2$, coeficientii β_{ij}, α_i, $i, j = 1,2$ sunt functii reale cunoscute definite pe D, functia $\varphi(x,y)$ este de asemenea cunoscuta, iar $C = \partial D$ este frontiera lui D. Rezolvarea ecuatiei (1) cu conditia ***pe contur*** (2) este cunoscuta sub numele de ***problema .....***.

a. Dirichlet
b. Poisson
c. Monte Carlo

A 98. Un caz particular de ecuatie de tip eliptic este ***ecuatia .....***, adica $\nabla V(P) = F(P)$, sau $\frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2} = F(x,y)$ cu conditia pe contur $V(P) = \varphi(P)$, $P \in C$, $C = D \subset R^2$.

a. Poisson
b. Dirichlet
c. Monte Carlo

Name: _____ **Class:** _____ **Date:** _____ **ID: A**

Subiecte proiectare si programare orientata obiect licenta informatica 4 ani

Multiple Choice
Identify the letter of the choice that best completes the statement or answers the question.

B 1. Fie următorul program C++:

```
#include <iostream.h>
class A{
public:
      void m(){cout<<"A:m()"<<endl;};
      virtual void v(){cout<<"A:v()"<<endl;};
};
class B: public A{
private:
      void m(){cout<<"B:m()"<<endl;};
      virtual void v(){cout<<"B:v()"<<endl;};
};
void main(){
      A a,*p;
      B b;
      a.m();
      p=&b;
     p->v();
}
```

Care din afirmatiile următoare sunt adevărate:

a. Expresia p->v()este incorectă deoarece metoda v este inaccesibilă
b. Programul afiseaza mesajul:
   A:m()
   B:(v)
c. Programul afisează mesajul: A:v()
d. Expresia a.m() este incorectă deoarece metoda m este inaccesibilă

D 2. Fie următorul program C++:

```
#include <iostream.h>
class A{
public:
      void s(){cout<<"void A::s()"<<endl;}
      void s(int i){i++;cout<<"void A::s(int i)"<<endl; }
      virtual void v(){cout<<"virtual void A::v()"<<endl;}
      virtual void v(int i){
     i++;cout<<"virtual void A::v()"<<endl;
   }

};
class B:public A{
public:
      // 1.
      int s(){
     cout<<"int A::s()"<<endl;
     return 1;
   }

      // 2.
      void s(int i){
     i++;cout<<"void A::s(int i)"<<endl;
   }

      // 3.
      virtual void v(int i){
     i++;cout<<"virtual void A::v()"<<endl;
   }

      // 4.
      virtual int v(){
     cout<<"int A::v()"<<endl; return 1;
   }
};
```

Care din afirmatiile următoare sunt adevărate:

a. //1. este incorectă
b. //2. este incorectă
c. //3. este incorectă
d. //4. este incorectă

B

3. Fie următorul program C++, în care operatorul de insertie << este supraîncărcat. Operatorul de atribuire = este cel predefinit, cu semantica prin referinta.

```
#include <iostream.h>
class C{
public:
      C(int n, int v[]);
            void set(int i, int val){pi[i]=val;}

friend     ostream& operator<<(ostream &o, const C&);

private:
      int dim;
      int *pi;
};

C::C(int n, int v[])  {
      dim=n;
      pi= new int[dim];
      for(int j=0; j<dim; j++){pi[j]=v[j];}
}

ostream& operator<<(ostream &o, const C &m){
      for(int j=0; j<m.dim; j++)o<< m.pi[j]<<" ";
      return o;
}

void main(){
      int a[]={1,2,3}, b[]={10,20};
      C x(3,a),y(2, b);
      cout<<(x=y)<<endl;
      y.set(0,1000);
      cout<<x<<endl;
}
```

Care din afirmatiile următoare sunt adevărate:

a. Programul afisează:
   10 20
   10 20

b. Programul afisează:
   10 20
   1000 20

c. Programul afisează:
   1000 20
   10 20

d. Programul afisează:
   1000 20
   1000 20

A 4. Fie următorul program C++, în care operatorul de insertie << si operatorul de atribuire = sunt supraîncărcati.
Prin supraîncărcare, operatorul de atribuire are semantica prin valoare (el înlocuieste pe cel predefinit, care are semantica prin referintă). Rersursele obiectului destinatie d din expresia d=s sunt eliberate chiar si în cazul de autoatribure d=d.

```
#include <iostream.h>
class C{
public:
      C(int n, int v[]);
      void set(int i, int val){pi[i]=val;}
      C& operator=(C&);

friend      ostream& operator<<(ostream &o, const C&);

private:
      int dim;
      int *pi;
};

C::C(int n, int v[])    {
      dim=n;
      pi= new int[dim];
      for(int j=0; j<dim; j++){pi[j]=v[j];}
}
// supraincarcare cu semantica prin valoare
C& C::operator=(C& x){

   // incepe prin dezalocare resursa *pi
   // fara verificarea cazului de autoatribuire d=d

      delete[] pi;
      dim=x.dim;
      pi=new int[dim];
      for(int j=0; j<dim; j++){pi[j]=x.pi[j];}
      return *this;
}

ostream& operator<<(ostream &o, const C &m){
      for(int j=0; j<m.dim; j++)o<< m.pi[j]<<" ";
      return o;
}

void main(){
      int a[]={1,2,3}, b[]={10,20};
      C x(3,a),y(2, b);
      cout<<(x=y)<<endl;
      y.set(0,1000);
      cout<<x<<endl;
      cout<<(x=x)<<endl;
}
```

În afirmatiile de mai jos, xx si yy sunt valori arbitrare, rezultate prin referirea prin pointerul pi la o zona de memorie *pi neinitializată
Care din afirmatiile următoare sunt adevărate:

a. Programul afisează:
10 20
10 20
xx yy

b. a) Programul afisează:
10 20
1000 20
xx yy

c. Programul afisează:
1000 20
10 20
xx yy

d. Programul afisează:
10 20
10 20
10 20

D 5. Fie următorul program C++,
în care operatorul de insertie << si operatorul de atribuire = sunt supraîncărcati. Prin supraîncărcare, operatorul de atribuire are semantica prin valoare (el înlocuieste pe cel predefinit, care are semantica prin referintă). Rersursele obiectului destinatie d din expresia d=s sunt eliberate numai dacă d?s.

```
#include <iostream.h>
class C{
public:
      C(int n, int v[]);
      void set(int i, int val){pi[i]=val;}
      C& operator=(C&);

friend      ostream& operator<<(ostream &o, const C&);

private:
      int dim;
      int *pi;
};

C::C(int n, int v[])    {
      dim=n;
      pi= new int[dim];
      for(int j=0; j<dim; j++){pi[j]=v[j];}
}

   C& C::operator=(C& x){
      // incepe prin dezalocare resursa *pi
      // numai daca  destinatia difera de sursa
    if(this!=&x){
                  dim=x.dim;
                  delete[] pi;
                  pi=new int[dim];
                  for(int j=0; j<dim; j++){pi[j]=x.pi[j];}
    }
    return *this;
}

ostream& operator<<(ostream &o, const C &m){
      for(int j=0; j<m.dim; j++)o<< m.pi[j]<<" ";
      return o;
}

void main(){
      int a[]={1,2,3}, b[]={10,20};
      C x(3,a),y(2, b);
      cout<<(x=y)<<endl;
      y.set(0,1000);
      cout<<x<<endl;
      cout<<(x=x)<<endl;
}
```

În afirmatiile de mai jos, xx si yy sunt valori arbitrare, rezultate prin referirea prin pointerul pi la o zona de memorie *pi neinitializată
Care din afirmatiile următoare sunt adevărate:

a. Programul afisează:
10 20
10 20
xx yy

b. Programul afisează:
10 20
1000 20
xx yy

c. Programul afisează:
1000 20
10 20
xx yy

d. Programul afisează:
10 20
10 20
10 20

A 6. Fie următorul program C++, unde functia abs calculează valoarea absolută a unui număr real:

```
#include <iostream.h>
void abs(float &x){if (x<0)x=-x;}
void main(){
      float a=-1;
      abs(a);
      cout<<"a="<<a<<endl;
      int i=-1;
      abs(i); //temporary used for parameter x
      cout<<"i="<<i<<endl; // i=-1

}
```

La compilare a fost emis următorul mesaj de avertizare:
Warning: temporary used for parameter 'x' in call to 'abs(float &)
Care din afirmatiile următoare sunt adevărate:

a. Programul afisează:
a =1
i = -1

b. Programul afisează:
a =1
i = 1

c. Programul afisează:
a = -1
i = -1

d. Programul afisează:
a = -1
i = 1

D 7.

Fie următorul program C++:

```
#include <iostream.h>
class Punct{
public:
      Punct(int=0, int=0); //constructor
protected:
      int x,y;
friend ostream& operator<<(ostream&, const Punct&);
};
class Cerc: public Punct{
public:
      Cerc(double r=0.0, int x=0, int y=0);// constructor
protected:
      float raza;
friend ostream& operator<<(ostream&, const Cerc&);
};

//implementare Punct

Punct::Punct(int a, int b):x(a),y(b){} // constructor
ostream& operator<<(ostream& output, const Punct& p){
      output<<"Punct"<<'['<<p.x<<", "<<p.y<< ']';
      return output;
}

//implementare Cerc

Cerc::Cerc(double r, int a, int b):Punct(a,b), raza(r){}
ostream& operator<<(ostream& output, const Cerc& c){
      output      <<"Centru= "<< (Punct)(c)
            <<"; Raza= "<<c.raza;
      return output;
};

void main(){
      Punct *punctPtr=0, p(30,50);
      Cerc *cercPtr=0, c(2.7,120,89);
      cout<<p<<endl;
      cout<<c<<endl;

      //  Cerc tratat ca  Punct (prin pointer la clasa de baza):
      punctPtr=&c;
      cout<< *punctPtr << endl;

/* Cerc tratat ca  Cerc (prin pointer la clasa de baza, dar
       cu conversie explicita de la Punct la  clasa derivata Cerc
*/
      cercPtr= (Cerc *)(punctPtr);
      cout<< *cercPtr<<endl;

/*Punct tratat ca Cerc: programatorul isi asuma responsabilitatea
 unor erori: anumite atribute sunt nedefinite
*/
      punctPtr= &p; // punctPtr  refera un Punct
      //Urmeaza conversie asumata de programator,
      //de la clasa de baza la clasa derivata
      cercPtr=(Cerc *)(punctPtr);
      // cercPtr refera p ca pe un cerc
      //dar acest asa-zis cerc are raza  nedefinita
      cout<< *cercPtr <<endl;
}
```

Prin xx este desemnată o valoare nedefinită.
Care din afirmatiile următoare sunt adevărate:

a. Programul afisează:
   Punct[30,50]
   Centru=Punct[30,50]; Raza=2.7
   Punct[120,89]
   Centru=Punct[120,89]; Raza=2.7
   Centru=Punct[30,50]; Raza=xx

b. Programul afisează:
   Punct[30,50]
   Centru=Punct[120,89]; Raza=2.7
   Punct[120,89]
   Centru=Punct[120,89];
   Centru=Punct[30,50]; Raza=xx

c. Programul afisează:
   Punct[30,50]
   Centru=Punct[30,50];
   Punct[120,89]
   Centru=Punct[120,89]; Raza=2.7
   Centru=Punct[30,50]; Raza=xx

d. Programul afisează:
   Punct[30,50]
   Centru=Punct[120,89]; Raza=2.7
   Punct[120,89]
   Centru=Punct[120,89]; Raza=2.7
   Centru=Punct[30,50]; Raza=xx

A 8. Fie următorul program C++:

```
// constructor de copiere in clasa de baza;
// dar absent in clasa derivata
#include <iostream.h>
class B{
public:
      B(){cout<<"B();";}
      B(B &b){cout<<"B(B &b);";}
};
class D: public B{
public:
      D(){cout<<"D();";}
};
void main(){
      B b;
      B b1(b);
      D d;
      D d1(d);
      B bd(d);
}
```

Care din afirmatiile următoare sunt adevărate:

a. Programul afisează:
   B() B(B &b) B() D() B(B &b) B(B &b)

b. Programul afisează:
   B() B() B() D() B(B &b) B(B &b)

c. Programul afisează:
   B() B() B(B &b) D() B(B &b) B(B &b)

d. programul afisează:
   B() B(B &b) B() D() B() B()

B 9. Fie următorul program C++:

```
// destructor static
#include <iostream.h>
#include <conio.h>
class B{
public:
      B(){cout<<"B()"<<endl;}
      ~B(){cout<<"~B()"<<endl;}
};
class D: public B{
public:
      D(){cout<<"D()"<<endl;}
      ~D(){cout<<"~D()"<<endl;}
};
void main(){
      clrscr();
      B *b=new B();     // apel B()
      delete b;
      b=new D();        // apel B();D()
      delete b;
}
```

Care din afirmatiile următoare sunt adevărate:

a. Programul afisează:
   B() ~B() B() D() ~D()
b. Programul afisează:
   B() ~B() B() D() ~B()
c. Programul afisează:
   B() ~B() B() ~B()
d. Programul afisează:
   B() ~B() D() ~B()

A 10. Fie următorul program C++:

```
// destructor virtual
#include <iostream.h>
class B{
public:
      B(){cout<<"B()"<<endl;}
      virtual ~B(){cout<<"~B()"<<endl;}
};
class D: public B{
public:
      D(){cout<<"D()"<<endl;}
      virtual ~D(){cout<<"~D()"<<endl;}
};
void main(){
      clrscr();
      B *b=new B();       // apel B()
      delete b;
      b=new D();       // apel B();D();
                      // destructorii, in ordine inversa
      delete b;
}
```

Care din afirmatiile următoare sunt adevărate:

a. Programul afisează:
   B() ~B() B() D() ~D() ~B()
b. Programul afisează:
   B() ~B() B() D() ~B() ~B()
c. Programul afisează:
   B() ~B() B() D() ~B() ~D()
d. Programul afisează:
   B() ~B() B() D() ~B()

C 11. Fie următorul program C++:

```
#include <string.h>
#include <iostream.h>
class Person{
public:
       Person(char *p){
       name=new char[strlen(p)+1];
       strcpy(name,p);
       }
       ~Person(){delete[] name;}
private:
      char *name;
};

void f(){
   // obiect nou, alocare dinamica in memoria heap
   Person *p =new Person("Balanescu");
   delete p;
}
void main(){
      while(1)f();
}
```

Care din afirmatiile următoare sunt adevărate:

a. Programul se termină, deoarece instructiunea delete p eliberează memoria alocată pentru obiectul referit de p
b. Programul se termină, deoarece la iesirea din functia f se eliberează memoria alocată pe stivă pentru obiectul referit de p
c. Programul nu se termină, deoarece conditia de continuare din instructiunea while este mereu îndeplinită
d. Programul se blochează la epuizarea memoriei heap

D 12. Fie următorul program C++:

```
#include <iostream.h>
class Persoana{
public:
      Persoana(char * n){nume=n;}
      void afisare(){cout<<nume<<endl;}
      char *nume;

};
Persoana q="Balanescu";
void fvalue(Persoana p){p.afisare();}
void faddress(Persoana *p){p->afisare();}
void freference(Persoana &p){p.afisare();}
void main(){
      fvalue(q);
      faddress(&q);
      freference(q);
      fvalue("Tudor");
      freference("Tudor");// Warning: temporary used for p
}
```

Care din afirmatiile următoare sunt adevărate:

a. Expresia faddress(&q); este eronată;
b. Instructiunea de declarare Persoana q="Balanescu"; este eronată;
c. Expresia p->afisare() este eronată;
d. Programul afisează:
   Balanescu
   Balanescu
   Balanescu
   Tudor
   Tudor

C 13. Fie următorul program C++:

```
#include <iostream.h>
class B{
public:
      B(int i):i(i){}
protected:
      int i;
};
class D1: public B{
public:
      D1(int i):B(i){}
      void inc(){i++;}
};
class D2: public B{
public:
      D2(int i):B(i){}
      void afisare(){cout<<"i="<<i<<endl;}
};
class  D: public D1, public D2{
public:
      D(int i):D1(i),D2(i){}
};
void main(){
      D d(0);
      d.inc();
      d.afisare();
}
```

Care din afirmatiile următoare sunt adevărate:

a. Programul afisează i=-1
b. Programul afisează i=2
c. Programul afisează i=0
d. Programul afisează i=1

B 14. Fie următorul program C++:

```
#include <iostream.h>
class B{
public:
      B(int i=-1):valB(i){}
      B& operator=(const B &b){
            valB=b.valB;
            cout<<"B::op= ";
            return *this;
      }
private:
      int valB;
};

class D:public B{
public:
      D(int i, int j):B(i),valD(j){}
      D& operator=(const D &d){
            valD=d.valD;
            cout<<"D::op= ";
            return *this;
      }
private:
      int valD;
};
void main(){
      B b(0), b1(1);
        B *pB;
      D d(0,0), d1(1,1);
      b=b1;
      d=d1;
      b=d;
      pB=&d;
      *pB=d;
   pB->operator=(d);
}
```

Care din afirmatiile următoare sunt adevărate:

a. Programul afisează :
   B::op= D::op= D::op= B::op= D::op=
b. Programul afisează:
   B::op= D::op= B::op= B::op= B::op=
c. Programul afisează:
   B::op= D::op= B::op= D::op= D::op=
d. Programul afisează:
   B::op= D::op= D::op= D::op= B::op=

B 15. Fie următorul program C++:

```
class A{
      int x;
public:
      A(int x):x(x){};
};

class B{
protected:
      int x;
public:
      B(int x):x(x){};
};
class D:public B{
      A a;
      B b;              // 1
      B b(1); // 2
public:
  void m(){
            x=1;        // 3
            //b.x=1; //x nu este accesibil obiectelor
  }
};
void main(){
      B b(1); // 4
}
```

Care din următoarele instructiuni sunt eronate:

a. B b; // 1
b. B b(1); // 2 (din clasa D)
c. x=1; // 3
d. B b(1); // 4 (din functia main)

C 16.
```
class A{
      int x;
public:
      A(int x):x(x){};
};

class B{
protected:
      int x;
public:
      B(int x):x(x){};
};
class D:public B{
      A a;
      B b;              // 1
public:
  void m(){
            x=1;        // 2
            b.x=1; //3
  }
};
void main(){
      B b(1); // 4
}
```

Care din următoarele instructiuni sunt eronate:

a. B b; // 1
b. x=1 // 2
c. b.x=1; // 3
d. B b(1); // 4 (din functia main)