Chaque question peut avoir un nombre quelconque de bonnes réponses.

Structures de données

1. La suite de bits 11101100 peut représenter ...

   (a) Un entier naturel plus grand que 1000.
   (b) Un entier négatif.
   (c) Un rationnel.
   (d) Un caractère.

2. Une pile est ...

   (a) Une structure FILO (First In, Last Out).
   (b) Une structure FIFO (First In, First Out).
   (c) Une structure disponible nativement en OCaml.
   (d) Utile pour implémenter un parcours en profondeur.
   (e) Utile pour implémenter un parcours en largeur.

3. Une file peut ...

   (a) Être implémentée par deux piles avec des opérations élémentaires en temps amorti constant.
   (b) Être implémentée par une seule pile.
   (c) Utile pour implémenter un parcours en profondeur.
   (d) Utile pour implémenter un parcours en largeur.

4. Une fonction de hachage peut être utilisée ...

   (a) En sécurité informatique.
   (b) Pour la reconnaissance de motif dans un texte.
   (c) Pour la mémoïsation.
   (d) Pour accélérer l'algorithme des k plus proches voisins.

5. Quelles structures sont persistantes ?

   (a) Liste OCaml.
   (b) Tableau.
   (c) Arbre binaire OCaml.
   (d) Tas binaire.
   (e) Chaîne de caractères en C.
   (f) Chaîne de caractères en OCaml.

6. Si on considère un arbre binaire strict et non vide à n noeuds internes, f feuilles et de hauteur h, quelles relations sont vérifiées parmi les suivantes ?

   (a) $f \leq n$.
   (b) $f + n < 2^{h+1} - 1$.
   (c) $n \geq h$.
   (d) $f = n + 1$.
   (e) $f + n + h \neq 0$.

7. En ignorant la valeur des étiquettes des noeuds, combien y a-t-il d'arbres binaires de hauteur exactement 2 ?

   (a) 3.
   (b) 14.
   (c) 21.
   (d) 35.

8. Le parcours en largeur d'un arbre est ...

   (a) Linéaire en le nombre de noeuds.
   (b) Linéaire en le nombre d'arêtes.
   (c) Linéaire en la hauteur.
   (d) Jamais utilisé en pratique.
   (e) L'ordre inverse du parcours en profondeur.

9. On note $r(a)$ la racine d'un arbre a. Soit a un arbre binaire non vide et de sous-arbres g et d. On suppose que les éléments de a sont tous différents. Quelle(s) proposition(s) suivante(s) sont équivalentes à « a est un arbre binaire de recherche (ABR) » ?

   (a) $r(g) < r(a)$, $r(d) < r(a)$, et g, d sont des ABR.
   (b) $r(g) < r(a) < r(d)$, et g, d sont des ABR.
   (c) Tous les éléments de g sont inférieurs à $r(a)$ et tous les éléments de d sont supérieurs à $r(a)$.
   (d) Le parcours infixe de a est croissant.
   (e) Le parcours préfixe de a est croissant.

10. Dans un ABR de taille n et de hauteur h, on peut ...

    (a) Trouver un élément en $O(n)$.
    (b) Trouver un élément en $O(h)$.
    (c) Trouver un élément en $O(\log n)$.
    (d) Insérer un élément en $O(\log n)$.
    (e) Supprimer un élément en $O(\log n)$.
    (f) Vérifier qu'il est bien ABR en $O(n)$.

11. Les arbres rouge-noir ...

    (a) Sont des ABR.
    (b) Peuvent être utilisés pour implémenter un dictionnaire.
    (c) N'assurent aucune garantie de complexité.
    (d) N'ont pas d'utilité pratique.
    (e) Nécessitent un bit d'information supplémentaire par noeud par rapport à un arbre binaire.
    (f) Ne peuvent pas stocker de valeurs.

12. La structure de tas peut être utilisée dans les algorithmes suivants ...

(a) Kruskal.

(b) Parcours en profondeur.

(c) Kosaraju.

(d) Algorithme du tri par tas.

(e) Dijkstra.

13. Il existe une implémentation de la structure unir et trouver (Union-Find) ayant une complexité ...

    (a) Constante pour l'opération unir.

    (b) Constante pour l'opération trouver.

    (c) Constante pour l'opération trouver et l'opération unir.

    (d) Constante en complexité amortie pour l'opération trouver et l'opération unir.

    (e) En $O(\log n)$ dans le pire cas pour les opérations trouver et unir où n est le nombre d'éléments manipulés.

14. Un graphe est régulier si tous ses sommets ont le même degré. Existe-t-il un graphe régulier ayant ...

    (a) 9 sommets.

    (b) Degré 5 avec 3 composantes connexes.

    (c) Degré 5 avec 9 sommets.

    (d) Degré 6 avec 6 sommets.

    (e) Un degré p et un nombre q de sommets avec des p et q premiers.

15. L'ensemble des graphes orientés à n sommets ...

    (a) Est en bijection avec l'ensemble des graphes non orientés à n sommets.

    (b) Est en bijection avec l'ensemble des graphes non orientés à $\lceil n/2 \rceil$ sommets.

    (c) Est en bijection avec le groupe symétrique S_n.

    (d) Est en bijection avec l'ensemble des mots de $\{a, b\}^{n^2-n}$.

    (e) Est en bijection avec l'ensemble des langages qui contiennent n mots sur $\{a, b\}$.

16. Un graphe non orienté à m arêtes et n sommets est un arbre si et seulement si ...

    (a) Il est acyclique.

    (b) Il est acyclique avec $m = n - 1$.

    (c) Il possède une racine.

    (d) Toute paire de sommets est reliée par un unique chemin.

    (e) Il est connexe avec $m < n$.

17. La complexité d'un parcours en profondeur du graphe $G = (S, A)$ est ...

    (a) En $O(|S|^2)$ quelle que soit la représentation du graphe.

    (b) En $O(|A|)$ si G représenté par listes d'adjacence.

    (c) En $O(|S|^2)$ si G est représenté par matrice d'adjacence.

    (d) En $O(|S|+|A|)$ si G est représenté par matrice d'adjacence.

    (e) En $O(|S|+|A|)$ si G est représenté par liste d'arêtes.

Algorithmique

18. Un invariant de boucle ...

    (a) Aide à prouver la correction d'un algorithme.

    (b) Est vérifié si et seulement si la propriété reste vraie après une itération si on la suppose vraie avant.

    (c) Est vérifié en sortie de boucle.

    (d) Correspond à une formulation de la correction de l'algorithme sur des sous-problèmes.

    (e) Aucune des propositions ci-dessus.

19. Un algorithme qui termine en temps probabiliste avec une réponse exacte est un algorithme de ...

    (a) Atlantic City.

    (b) Atlanta.

    (c) Las Vegas.

    (d) Macao.

    (e) Monte Carlo.

20. Un algorithme dont le temps de calcul est garanti mais dont le résultat peut être inexact avec une certaine probabilité est un algorithme de ...

    (a) Atlantic City.

    (b) Atlanta.

    (c) Las Vegas.

    (d) Macao.

    (e) Monte Carlo.

21. Si pour un problème $\mathcal{P}$ on dispose d'un algorithme $\mathcal{A}$ qui s'exécute en temps polynomial, sans faux négatif et avec probabilité de faux positif inférieure à 1/3 ...

    (a) $\mathcal{P} \in P$.

    (b) On peut résoudre $\mathcal{P}$ sans faux négatif et avec une probabilité de faux positif arbitrairement petite.

    (c) Si $\mathcal{A}$ affirme qu'une instance de $\mathcal{P}$ est négative, c'est bien le cas.

    (d) Il existe un algorithme qui résout $\mathcal{P}$ en temps polynomial en moyenne.

22. L'algorithme du tri rapide ...

    (a) A une complexité pire cas en $\Theta(n \log n)$ et une complexité moyenne en $\Theta(n)$.

    (b) A une complexité pire cas en $\Theta(n^2)$ et une complexité moyenne en $\Theta(n \log n)$.

    (c) Est un algorithme de type Las Vegas.

(d) A la même complexité dans le pire cas et en moyenne.

(e) Peut terminer plus tard dans sa version probabiliste que dans le pire cas de la méthode déterministe.

23. Le retour sur trace (backtracking) ...

(a) Correspond à un parcours en largeur.

(b) Correspond à un parcours en profondeur.

(c) Peut entraîner des boucles.

(d) Nécessite d'organiser les données.

(e) Est utilisé par l'algorithme de Quine.

24. Un algorithme de type branch-and-bound ...

(a) S'applique à un problème de décision.

(b) Est une variante d'algorithme diviser pour régner.

(c) Est une variante d'un algorithme de backtracking.

(d) Renvoie une approximation de la solution.

(e) Est utile pour trouver un chemin dans un graphe.

25. Un algorithme glouton ...

(a) N'est utile que s'il donne une solution exacte.

(b) Peut être un algorithme d'approximation.

(c) Est forcément polynomial.

(d) Est un cas particulier de backtracking.

(e) Ne revient jamais sur une décision prise à une étape précédente.

26. Quels sont les algorithmes gloutons parmi les suivants ?

(a) Huffman.

(b) Dijkstra.

(c) Kruskal.

(d) Algorithme pour sac à dos en prenant les objets par ordre croissant de valeur.

27. On note $(I_n)_{n\in\mathbb{N}}$ une suite d'instances d'un problème de minimisation $\mathcal{P}$, $\mathcal{A}$ un algorithme qui donne des solutions aux instances de $\mathcal{P}$, $C_n^\star$ le coût optimal pour I_n et C_n le coût de la solution trouvée par $\mathcal{A}$ sur I_n.

(a) Pour tout $n \in \mathbb{N}$, $C_n \leq C_n^\star$.

(b) S'il existe $\alpha > 0$ tel que $\forall n \in \mathbb{N}$, $C_n \leq \alpha C_n^\star$, alors $\mathcal{A}$ est une α-approximation pour $\mathcal{P}$.

(c) Si $C_n^\star/C_n \to +\infty$ à l'infini, alors $\mathcal{A}$ n'est pas un algorithme d'approximation pour $\mathcal{P}$.

(d) Si $C_n/C_n^\star \to +\infty$ à l'infini, alors $\mathcal{A}$ n'est pas un algorithme d'approximation pour $\mathcal{P}$.

28. Un algorithme de programmation dynamique ...

(a) Nécessite de stocker des valeurs intermédiaires.

(b) Consiste exclusivement à mémoïser des résultats.

(c) Consiste à partitionner les solutions en sous-problèmes distincts.

(d) Consiste à formuler la solution d'un problème en fonction de la solution à des sous problèmes.

(e) Est inadapté lorsque les sous problèmes se recoupent.

29. Quels algorithmes utilisent de la programmation dynamique ?

(a) Dijkstra.

(b) Floyd-Warshall.

(c) Kosaraju.

(d) Calcul des attracteurs.

(e) Quine.

30. Quels algorithmes utilisent la méthode diviser pour régner ?

(a) Tri rapide.

(b) Tri fusion.

(c) Recherche par dichotomie dans un tableau trié.

(d) Construction d'un arbre k-d.

(e) ID3.

31. En posant $C(0) = C(1) = 1$, pour quelles formules de récurrence obtient-on $C(n) = \Theta(\lambda^n)$ avec $\lambda > 1$?

(a) $C(n) = 3C(n-1)$.

(b) $C(n) = \displaystyle\sum_{i=0}^{n-1} C(i)$.

(c) $C(n) = 3C(n/2) + O(n^2 \log n)$.

(d) $C(n) = nC(n/2)$.

(e) $C(n) = 2C(n/2) + \Theta(2^n)$.

32. Pour rechercher si un motif m apparaît dans un texte t on peut ...

(a) Utiliser un automate.

(b) Utiliser l'algorithme de Lempel-Ziv-Welch.

(c) Utiliser l'algorithme de Rabin-Karp.

(d) Utiliser l'algorithme de Boyer-Moore.

(e) Conclure en temps $O(|m||t|)$.

33. Un ordre topologique d'un graphe orienté G :

(a) existe ssi G est acyclique.

(b) est unique s'il existe.

(c) peut être calculé avec un parcours en largeur.

(d) peut être calculé en temps O(|S| + |A|)

34. L'algorithme de Kosaraju appliqué à un graphe ...
    (a) Calcule les composantes fortement connexes.
    (b) Utilise un double parcours en profondeur.
    (c) Utilise le graphe transposé.
    (d) Permet de résoudre 2SAT en temps polynomial.
    (e) Utilise une structure de pile.

35. Lesquels de ces algorithmes permettent de calculer la plus petite distance entre deux sommets d'un graphe ?
    (a) $A^\star$.
    (b) Alpha-beta.
    (c) Dijkstra.
    (d) Boyer-Moore.
    (e) Floyd-Warshall.

36. L'algorithme $A^\star$ avec une heuristique h ...
    (a) Correspond à l'algorithme de Floyd-Warshall si h est identiquement nulle.
    (b) Trouve un plus court chemin si h est admissible.
    (c) Trouve un plus court chemin si h est monotone.
    (d) Est polynomial en temps si h est admissible.
    (e) Est polynomial en temps si h est monotone.
    (f) N'est pas intéressant à appliquer sur un graphe non pondéré.

37. Quelles propriétés sont vraies ?
    (a) Tout graphe possède un arbre couvrant.
    (b) Dans un graphe G, tout arbre couvrant de poids minimum (ACM) contient au moins une arête de poids minimal de G.
    (c) L'arête de poids maximal n'appartient à aucun ACM.
    (d) Si les poids de toutes les arêtes de G sont deux à deux distincts et que G est connexe alors G admet un unique ACM.
    (e) Si G admet un unique ACM, les poids de ses arêtes sont deux à deux distincts.

38. Quelles propriétés sont vraies ?
    (a) Un couplage maximal est maximum.
    (b) Un couplage maximum est maximal.
    (c) Un graphe admettant un couplage parfait est nécessairement biparti.
    (d) Dans un graphe biparti, on peut calculer un couplage maximal en temps polynomial.
    (e) Un couplage sans chemin augmentant dans un graphe biparti est maximum.

39. Soient C_1, C_2 deux couplages d'un graphe G. Alors :
    (a) $C_1 \cup C_2$ est un couplage.
    (b) $C_1 \cap C_2$ est un couplage.
    (c) $C_1 \Delta C_2$ ($C_1 \cup C_2 \setminus C_1 \cap C_2$) est un couplage.
    (d) $C_1 \setminus C_2$ est un couplage.

40. L'algorithme des k plus proches voisins ...
    (a) Est un algorithme d'apprentissage non supervisé.
    (b) Gagne en précision lorsqu'on augmente k.
    (c) Ne peut pas être acceléré par prétraitement.
    (d) Aucune des réponses ci-dessus.

41. Un arbre de décision ...
    (a) Peut être construit par un algorithme d'apprentissage supervisé.
    (b) Classe toujours correctement tous les éléments de l'ensemble d'apprentissage.
    (c) Peut être amélioré à l'aide d'arbres k-d.
    (d) Peut être calculé sans calcul d'entropie par l'algorithme ID3.
    (e) A une hauteur bornée par la dimension des données.

42. L'algorithme des k moyennes ...
    (a) Repose sur l'entropie de Shannon.
    (b) Converge vers une réponse optimale.
    (c) Ne converge pas nécessairement.
    (d) Ne reconnaît pas les classes non convexes.
    (e) Nécessite un calcul de médioïde.

43. La classification obtenue par un algorithme de regroupement hiérarchique ascendant ...
    (a) Est indépendante de la distance choisie.
    (b) Permet de choisir le nombre de clusters.
    (c) Ne reconnaît pas les classes non convexes.
    (d) Nécessite un calcul de centroïde.
    (e) A une complexité dépendante de la distance considérée.

44. Un algorithme reposant sur une heuristique ...
    (a) Ne peut pas donner une réponse optimale à coup sûr.
    (b) Ne peut généralement pas garantir le temps d'exécution et l'optimalité du résultat.
    (c) Ne peut pas être exécuté avec une heuristique différente.
    (d) Peut être ajusté pour une utilisation pratique précise en adaptant l'heuristique considérée.
    (e) Aucune des réponses ci-dessus.

45. Les positions gagnantes pour un joueur J_1 dans un jeu à deux joueurs ...
    (a) Peuvent être calculées en temps polynomial en la taille du jeu.
    (b) Dépendent des réponses de l'adversaire.
    (c) Sont nécessairement des sommets contrôlés par le joueur J_1.
    (d) Peuvent mener à une défaite de J_1.
    (e) Aucune des réponses ci-dessus.

Logique

46. La formule « $(A \to B) \leftrightarrow (\neg A \lor B)$ » :
    (a) est vraie au sens de la logique propositionnelle
    (b) est prouvable dans la logique minimale
    (c) est prouvable dans la logique classique (en ajoutant tiers exclu ou raa à la logique minimale)

47. Quelles formules sont des tautologies ?
    (a) $((A \to B) \to C) \leftrightarrow (A \to (B \to C))$.
    (b) $(A \lor B) \land (\neg A \lor \neg B)$.
    (c) $\neg(A \land (A \lor B)) \leftrightarrow \neg A$.
    (d) $((A \lor B) \land (C \lor D)) \leftrightarrow (A \land B) \lor (B \land D)$.
    (e) $(A \to B) \leftrightarrow (\neg A \lor B)$.

48. La taille d'une formule est majorée par sa hauteur.
    (a) Vrai.
    (b) Faux.

49. Une variable libre ...
    (a) Peut être liée à un autre endroit dans la formule.
    (b) A une portée.
    (c) N'influe pas la sémantique d'une formule.
    (d) Est toujours associée à une variable liée.

50. Soit φ et ψ deux formules telles que $\psi \models \varphi$. Cela signifie ...
    (a) Que φ est conséquence de ψ.
    (b) Que ψ est conséquence de φ.
    (c) Que les valuations satisfaisant φ satisfont aussi ψ.
    (d) Si φ est une antilogie (toujours fausse), alors ψ aussi.
    (e) Cela n'a pas de sens, la notation $\models$ n'existe que sous la forme $v \models \varphi$ avec v une valuation.

51. Une équivalence entre formules du calcul propositionnel ...
    (a) Signifie qu'elles ont le même arbre de dérivation.
    (b) Peut se prouver à l'aide de tables de vérité.
    (c) Peut se prouver à l'aide de substitutions dans des formules qu'on sait déjà être équivalentes.
    (d) Ne peut pas être démontrée sans la déduction naturelle.
    (e) Peut se prouver par une étude de la matrice de confusion.

52. Pour toute formule du calcul du calcul propositionnel il existe une formule équivalente ...
    (a) Sous forme normale conjonctive.
    (b) Sous forme normale disjonctive.
    (c) Sous forme normale de Quine.
    (d) Sous forme normale littérale.
    (e) Sous forme normale de Chomsky.

53. On sait résoudre le problème SAT en temps polynomial sur des instances ...
    (a) Sous 2-CNF (forme normale conjonctive).
    (b) Sous 3-CNF.
    (c) Sous 2-DNF (forme normale disjonctive).
    (d) Sous 3-DNF.
    (e) Ne faisant intervenir que les connecteurs $\land$ et $\neg$.

54. Quels séquents sont prouvables en logique classique ?
    (a) $\neg\neg A \vdash A$.
    (b) $\neg\neg\neg A \vdash \neg A$.
    (c) $A \land B \vdash A \lor B$.
    (d) $A \lor B \vdash A \land B$.
    (e) $A, A \to B \vdash B$.

55. Pour la déduction naturelle en logique propositionnelle, $\Gamma \vdash F$ si et seulement si $\Gamma \models F$.
    (a) Vrai.
    (b) Faux.

Théorie des langages

56. Sur l'aphabet $\Sigma = \{a, b\}$:
    (a) $\emptyset$ est une lettre.
    (b) $\emptyset$ est un mot.
    (c) $\emptyset$ est un langage.
    (d) ε est une lettre.
    (e) ε est un mot.
    (f) ε est un langage.

57. Si u et v sont deux mots, quelles propriétés parmi les suivantes sont vraies ?
    (a) $|u| > 0$.
    (b) u est préfixe de v ou v est préfixe de u.
    (c) Si u est préfixe de v alors v est suffixe de u.
    (d) Si $u = vw$ alors v est un sous-mot de u.
    (e) Si $u = vw$ alors v est un facteur de u.

58. Si L, L' et L'' sont des langages alors ...

(a) Si L et L' sont finis, $|LL'| = |L| \times |L'|$.

(b) $L^\star$ est infini.

(c) $L \subset LL$.

(d) $L(L' \cap L'') = LL' \cap LL''$.

59. Si L et L' sont deux langages tels que $L \subset L'$, quelles affirmations sont vraies ?

(a) Si L' est reconnaissable alors L aussi.

(b) Si L' n'est pas régulier, L non plus.

(c) Si L' est fini, L est régulier.

(d) Si L est reconnaissable, il l'est par un automate déterministe et complet.

(e) Si L est reconnaissable, il l'est par un automate émondé (tous les états sont accessibles et co-accessibles) et complet.

60. Le langage $L = \{a^n b^m \mid n \equiv m \bmod 2\}$ est dénoté par l'expression régulière ...

(a) $(aa|bb)^\star$.

(b) $(aa|ab|ba|bb)^\star$.

(c) $((a(a|b)^\star b)^\star + (b(a|b)^\star a)^\star)^\star$.

(d) $a^n b^m$.

61. Les langages réguliers sont stables par ...

(a) Préfixe.

(b) Miroir.

(c) Complémentaire.

(d) Intersection.

62. Un automate fini déterministe et complet ...

(a) Est nécessairement émondé.

(b) Est unique pour chaque langage à renommage près des états.

(c) Permet de tester l'appartenance d'un mot à un langage en temps linéaire.

(d) Donne aisément un automate reconnaissant le complémentaire du langage.

(e) Est de taille exponentielle en la taille d'un automate non déterministe reconnaissant le même langage.

63. Si e est une expression régulière, le problème consistant à savoir si un mot u appartient à $L(e)$ ...

(a) Nécessite au moins une complexité linéaire en $|u|$.

(b) Peut être résolu en temps polynomial en $|u| + |e|$.

(c) Est un problème indécidable.

(d) N'a pas d'application pratique.

64. Tout automate est équivalent à...

(a) un automate déterministe, construit en complexité polynomiale.

(b) un automate complet.

(c) un automate déterministe complet.

(d) un automate déterministe complet émondé.

(e) un automate sans ε-transition.

65. Un automate produit de deux automates A_1 et A_2 d'états Q_1 et Q_2...

(a) possède $Q_1 \times Q_2$ comme ensemble d'états.

(b) peut permettre de reconnaître $L(A_1)L(A_2)$.

(c) peut permettre de reconnaître $L(A_1) \cup L(A_2)$.

(d) peut permettre de reconnaître $L(A_1) \cap L(A_2)$.

(e) peut permettre de reconnaître $L(A_1) \setminus L(A_2)$.

(f) peut permettre de reconnaître $L(A_1) \Delta L(A_2)$.

66. On peut montrer que tout langage régulier est reconnaissable par un automate fini ...

(a) En construisant l'automate des parties.

(b) En utilisant les automates de Thompson.

(c) En utilisant la méthode d'élimination des états.

(d) Avec l'algorithme de McNaughton Yamada.

(e) En utilisant l'algorithme de Berry-Sethi.

67. Parmi les langages suivants, lesquels sont réguliers ?

(a) $\{a^n b^n \mid n \geq 0\}$.

(b) Les mots bien parenthésés.

(c) $\{a^n b^m \mid n \equiv m \bmod 2\}$.

(d) Les mots sur $\{0, 1\}$ correspondant aux écritures binaires d'entiers divisibles par 3.

(e) $\{a^p \mid p \text{ est premier}\}$.

68. Un langage L est local ssi...

(a) L est reconnu par un automate local.

(b) L est inclus dans un langage régulier.

(c) $L \backslash \{\varepsilon\} = P(L)\Sigma^* \cap \Sigma^* S(L) \backslash \Sigma^* N(L) \Sigma^*$ où $N(L) = \Sigma^2 \backslash F(L)$.

(d) $u \in L \iff u_1 \in P(L) \wedge u_n \in S(L) \wedge \forall k, u_k u_{k+1} \in F(L)$

69. Parmi les conditions suivantes, lesquelles sont suffisantes pour que L soit régulier ?

(a) L est dénombrable.

(b) L est reconnu par un automate non déterministe.

(c) L est le complémentaire d'un langage régulier.

(d) $L^\star$ est régulier.

(e) $L \subset \{a\}^\star$.

(f) L est local.

70. Si un langage vérifie le lemme de l'étoile, alors il est régulier.

(a) Vrai.

(b) Faux.

71. Les langages algébriques ...

(a) Sont inclus dans les langages réguliers.

(b) Sont stables par étoile de Kleene.

(c) Sont stables par complémentaire.

(d) Sont stables par intersection.

(e) Sont stables par concaténation.

72. Quelles sont les grammaires ambiguës ?

(a) $S \to SS \mid a$.

(b) $S \to ST \mid TS$, $T \to SS$.

(c) $S \to aSbS \mid bSaS \mid \varepsilon$.

(d) $S \to aSb \mid ab \mid \varepsilon$.

(e) $S \to aS \mid bS \mid \varepsilon$.

73. Les dérivations droites sont en bijection avec les arbres de dérivation.

(a) Vrai.

(b) Faux.

74. Un analyseur lexical ...

(a) Intervient avant l'analyseur syntaxique.

(b) Identifie les lexèmes.

(c) Détermine la structure d'un programme.

(d) Peut utiliser un automate fini.

75. Un analyseur syntaxique ...

(a) Ne peut gérer que des grammaires non ambiguës.

(b) Considère les lexèmes comme des terminaux.

(c) N'est pas utilisé en pratique.

(d) Choisit la grammaire à considérer en fonction des données en entrée.

Complexité et calculabilité

76. Parmi les problèmes ci-dessous, lesquels sont des problèmes de décision ?

(a) La résolution d'une grille de Sudoku.

(b) La primalité d'un entier.

(c) Le plus court chemin entre deux points.

(d) Le problème de l'arrêt.

(e) L'égalité entre deux langages.

77. Parmi les problèmes ci-dessous, lesquels sont des problèmes d'optimisation ?

(a) La résolution d'une grille de Sudoku.

(b) La primalité d'un entier.

(c) Le plus court chemin entre deux points.

(d) Le problème de l'arrêt.

(e) L'égalité entre deux langages.

78. Si n est la taille de l'entrée, pour quelles complexités peut-on dire que l'algorithme considéré est polynomial ?

(a) $O(\log n!)$.

(b) $2^{o(n)}$.

(c) $O((\log n)^n)$.

(d) $O(\sqrt{n})$.

(e) $O(2^n)$.

79. L'algorithme cherchant à déterminer si un entier n est premier en calculant son modulo par tous les entiers entre 2 et $\lceil \sqrt{n} \rceil$ est de complexité ...

(a) Quasi-linéaire.

(b) Polynomiale.

(c) Sous-linéaire.

(d) Exponentielle.

(e) Linéaire.

80. La complexité d'un problème de décision correspond à ...

(a) La complexité pire cas du meilleur algorithme qui le résout.

(b) La complexité pire cas du pire algorithme qui le résout.

(c) La complexité meilleur cas du meilleur algorithme qui le résout.

(d) Dépend du format d'entrée des instances.

(e) N'est pas définie.

81. On suppose que A est NP-complet. Soit $B \in NP$. Pour montrer que B est NP-complet, il suffit de montrer que...

(a) A se réduit polynomialement à B.

(b) B se réduit polynomialement à A.

(c) $A \leq_p B$.

(d) $B \leq_p A$.

(e) À chaque instance I_A de A, on peut associer une instance I_B de B en temps polynomial telle que I_A est une instance positive de A ssi I_B est une instance positive de B.

(f) À chaque instance I_B de B, on peut associer une instance I_A de A en temps polynomial telle que I_A est une instance positive de A ssi I_B est une instance positive de B.

82. Si le problème A se réduit polynomialement au problème B alors ...
    (a) Si A se résout en temps polynomial, B aussi.
    (b) Si B se résout en temps polynomial, A aussi.
    (c) A est considéré comme plus difficile que B.
    (d) Une instance de A peut être transformée en une instance de B.
    (e) Si A est décidable alors B aussi.

83. Si un problème est dans NP alors ...
    (a) On ne sait pas le résoudre en temps polynomial.
    (b) Il peut être résolu en temps exponentiel.
    (c) Il peut être résolu en espace polynomial.
    (d) Il s'agit d'un problème fonctionnel.
    (e) Pour toute instance du problème, on peut créer un certificat en temps polynomial.

84. Lesquels de problèmes suivants sont NP-complets ?
    (a) `3SAT`.
    (b) Le problème de l'arrêt.
    (c) L'existence d'un chemin hamiltonien.
    (d) L'appartenance d'un mot à un langage régulier.
    (e) Le problème CLIQUE.

85. Les problèmes indécidables ...
    (a) Ne sont pas les mêmes suivant le langage de programmation choisi.
    (b) Ne peuvent pas être réduits à un autre problème.
    (c) Dépendent de la façon dont sont représentées les instances du problème.
    (d) Ne concernent que des problèmes se rapportant au fonctionnement d'algorithmes.

86. Parmi les suivants, quels sont les problèmes indécidables ?
    (a) Le problème de l'arrêt sur les entrées de taille inférieure à 5.
    (b) Le problème de la terminaison en moins de n étapes sur toute instance de taille inférieure à n.
    (c) Le problème de la terminaison en strictement plus de n étapes sur toute instance de taille inférieure à n.
    (d) Le problème de l'équivalence de deux expressions régulières.
    (e) Le problème de l'équivalence entre deux formules du calcul propositionnel.