Il peut y avoir plusieurs items à cocher par question. Toutes les questions ont au moins un item à cocher. Une réponse est considérée comme juste si aucune mauvaise proposition est sélectionnée et que toutes les bonnes propositions sont sélectionnées.

1 Programmation

1. En C, une variable globale est allouée :

   (A) sur la pile
   (B) sur le tas
   (C) aucun des deux

2. En C, c'est une erreur de :

   (A) appeler `free` deux fois sur le même pointeur issu d'un `malloc`
   (B) appeler `free` sur un pointeur vers un objet sur la pile
   (C) appeler `free` sur `&t[1]`, si `t` est un pointeur
   (D) ne jamais appeler `free` sur un pointeur issu d'un `malloc`
   (E) appeler `free` sur le pointeur nul

3. Quand on fait `let u = v @ w` en OCaml :

   (A) cela prend un temps proportionnel à $|v|$
   (B) cela prend un temps proportionnel à $|w|$
   (C) cela prend un temps proportionnel à $|v| + |w|$
   (D) u et v sont partiellement aliasées
   (E) u et w sont partiellement aliasées

2 Structure de Données

4. Quelle structure de donnée correspond au principe LIFO ?

   (A) Une pile
   (B) Un tableau dynamique
   (C) Une file
   (D) Une table de hachage
   (E) Une liste doublement chaînée

5. Quelle structure de donnée correspond au principe FIFO ?

   (A) Une pile
   (B) Un tableau dynamique
   (C) Une file
   (D) Une table de hachage
   (E) Une liste doublement chaînée

6. Qu'est-ce qu'une collision ?

   (A) L'absence de réponses lors d'une requête serveur.
   (B) L'arrêt d'un fil d'exécution à cause d'une impossibilité d'entrer en section critique.
   (C) Deux éléments ayant la même valeur de hachage.
   (D) La tentative de retrait d'un élément dans une pile vide.
   (E) La tentative d'ajout d'un élément dans une file pleine.

7. Une file peut :

(A) Être implémentée par deux piles avec des opérations élémentaires en temps constant en complexité amortie.

(B) Être implémentée par deux piles avec des opérations élémentaires en temps constant dans le pire cas.

(C) Être implémentée avec une seule pile.

(D) Simuler une table de hachage.

(E) Être implémentée avec avec une liste simplement chainée mutable et une complexité constante sur les opérations élémentaires.

8. Les tableaux circulaires :

(A) Nécessitent une architecture spéciale.

(B) Permettent d'implémenter les piles.

(C) Permettent d'implémenter les files.

(D) Ont une taille fixe.

(E) Sont spécifiques au C et au Python et ne peuvent être utilisé en OCaml.

9. 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

10. On considère un arbre binaire strict, avec n nœuds internes, f feuilles et hauteur h. Quelles relations sont vérifiées ?

(A) $f \leq n$

(B) $f + n < 2^{h+1} - 1$

(C) $n \geq h$

(D) $f = n + 1$

(E) $f + n + h \neq 0$

11. Un tas(min) binaire :

(A) permet de réaliser efficacement une file de priorité.

(B) permet de lire l'élément minimal en temps constant.

(C) permet d'extraire l'élément minimal en temps constant.

(D) permet de réaliser efficacemment un dictionnaire.

(E) permet d'ajouter un élément en temps logarithmique.

12. Le parcours en largeur d'un arbre est :

(A) Linéaire en le nombre de nœuds.

(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.

13. On peut reconstruire un arbre binaire strict à partir du résultat de :

(A) son parcours en largeur

(B) son parcours en profondeur infixe

(C) son parcours en profondeur préfixe

(D) son parcours en profondeur postfixe

14. Un arbre rouge-noir :

(A) permet de réaliser efficacement une file de priorité.

(B) permet de lire l'élément minimal en temps constant.

(C) permet d'extraire l'élément minimal en temps constant.

(D) permet de réaliser efficacemment un dictionnaire.

(E) permet d'ajouter un élément en temps logarithmique.

(F) n'offre de bonnes garanties de complexité qu'en moyenne.

(G) est une arbre binaire de recherche.

15. Les arbres bicolores

(A) Peuvent être utilisés pour coder une structure de dictionnaire.

(B) N'assurent aucune garantie de complexité

(C) N'ont pas d'utilité pratique.

(D) Nécessitent un bit d'informations supplémentaires par nœuds qu'un arbre binaire.

(E) Ne peuvent pas stocker de valeurs.

16. La structure de tas est utilisée dans les algorithmes suivants :

(A) Algorithme de Huffman

(B) Algorithme de parcours en profondeur

(C) Algorithme de Kosaraju

(D) Algorithme de tri par tas

(E) Algorithme de Dijkstra

17. Un tas à n noeuds est

(A) Un arbre binaire parfait.

(B) Avec $\lceil \frac{n}{2} \rceil$ feuilles.

(C) De hauteur $O(\log_2(n))$.

(D) Représentable facilement par un tableau de taille n.

(E) Un arbre binaire de recherche équilibré.

18. Il existe une implémentation de la structure unir et trouver qui permet une complexité :

(A) Constante pour l'opération unir

(B) Constante pour l'opération trouver

(C) Constante dans le pire cas pour l'opération unir et l'opération trouver.

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

(E) En $o(\log(n))$ dans le pire cas pour les opérations unir et trouver.

19. Un ABR :

(A) permet de chercher un élément en temps linéaire en sa hauteur.

(B) permet de chercher un élément en temps logorithmique en sa taille.

(C) a des étiquettes croissantes dans l'ordre préfixe.

(D) De degré 6 avec 6 sommets

(E) Avec un degré p et un nombre de sommets q tels que p et q sont premiers.

20. Le nombre chromatique d'un graphe G

(A) Correspond au plus grand entier k tel qu'il existe une k-coloration de G.

(B) Correspond au plus petit entier k tel qu'il existe une k-coloration de G.

(C) Dépend de l'implémentation des graphes en machines.

(D) Est majoré par le degré maximum du graphe.

(E) Est minoré par le degré minimum du graphe.

21. Un tableau redimensionnable :

(A) permet l'accès à un élément par son indice en temps constant dans le pire cas.

(B) permet l'ajout d'un élément en temps contant dans le pire cas.

(C) permet l'ajout d'un élément à droite en temps constant amorti.

(D) permet l'ajout d'un élément à une position quelconque en temps constant amorti.

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

(A) Il est acyclique.
(B) Il est acyclique avec $m = n - 1$.
(C) Toute paire de sommets est reliée par un unique chemin.
(D) Il possède une racine.
(E) Il est connexe avec $m < n$.

23. Le poids d'un sous-graphe T de G

(A) Ne peut pas être négatif.
(B) Est majoré par la somme du poids de toutes les arêtes du graphe G.
(C) Correspond à la somme des poids des arêtes de T.
(D) Dépend du sens de parcours du graphe.
(E) Aucune des réponses ci-dessus.

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

(A) En $O\left(|S|^2\right)$ quelle que soit la représentation du graphe.
(B) En $O\left(|A|\right)$ pour un graphe sous forme de liste d'adjacence.
(C) En $O\left(|S|^2\right)$ pour une représentation sous forme de matrice d'adjacence.
(D) En $O\left(|S| + |A|\right)$ pour une représentation par matrice d'adjacence.
(E) Est en $O\left(|S| + |A|\right)$ pour une représentation par matrice d'incidence.

25. La représentation naturelle du graphe des pages webs reliés par un lien hypertexte est :

(A) Par matrice d'incidence.
(B) Par matrice d'adjacence.
(C) Par liste d'adjacence.
(D) Par forêt.
(E) Aucune des propositions ci-dessus.

26. Soit A un ensemble construit par induction.

(A) Tout ensemble E possédant les mêmes propriétés inductives que A englobe E.
(B) Certains objets de A ont été construits par une infinité d'opérations.
(C) A est en bijection avec $\mathbb{N}$.

3 Algorithmie

27. Un invariant de boucle

(A) Permet de prouver la correction d'un algorithme.
(B) Est vérifié si et seulement si la propriété est vraie lorsqu'on la suppose vraie au début de la boucle.
(C) Peut ne pas être vérifié à la sortie de la boucle.
(D) Correspond à une formulation de la correction de l'algorithme sur des sous-problèmes.
(E) Aucune des propositions ci-dessus.

28. Le tri par tas :

(A) est un tri par comparaisons.
(B) peut-être réalisé en place.
(C) a un complexité en $O(n \log(n))$ dans le pire cas.

29. Le tri rapide :

(A) est un tri par comparaisons.
(B) peut-être réalisé en place.
(C) a une complexité en $O(n \log(n))$ dans le pire cas.
(D) peut gagner à être randomisé.
(E) a une complexité linéaire dans le meilleur cas.

30. Un algorithme qui se termine en un temps probabiliste, mais avec une réponse exacte est un algorithme de

(A) Atlanta
(B) Bellagio
(C) Las Vegas
(D) Macao
(E) Monte-Carlo

31. Un algorithme dont le temps de calcul est garanti, mais dont le résultat n'est correct qu'avec une certaine probabilité est un algorithme de

(A) Atlanta
(B) Bellagio
(C) Las Vegas
(D) Macao
(E) Monte-Carlo

32. L'algorithme du tri rapide

(A) A une complexité dans le pire cas en $\Theta(n \log(n))$ et une complexité moyenne en $\Theta(n)$
(B) A une complexité dans le pire cas en $\Theta(n^2)$ et une complexité moyenne en $\Theta(n \log(n))$
(C) Est un algorithme d'Atlanta.
(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.

33. Parmi les algorithmes suivants lesquels permettent de trouver les occurrences d'une chaine de caractères dans une autre :

(A) Huffman
(B) Bellman Ford
(C) Boyer Moore
(D) Rabin Karp
(E) Liv Zempel Welch

34. Un algorithme par retour sur trace :

(A) Correspond à un parcours en largeur d'un arbre.
(B) Correspond à un parcours en profondeur d'une arbre.
(C) est un algorithme de recherche exhaustive.
(D) Nécessite d'ordonner les données.
(E) Permet une résolution polynomiale d'un Sudoku.
(F) Demande de pouvoir rejeter des solutions partielles.

35. Un parcours en profondeur permet de :

(A) Décider si un graphe est 2-coloriable.
(B) Calculer les distances entre chaque sommet et un point donner.
(C) Calculer les composantes connexes.
(D) Obtenir un tri topologique.
(E) Détecter les cycles.
(F) Calculer les composantes fortement connexes.

36. Un algorithme par séparation et évaluation

(A) Permet de résoudre certains problèmes entiers en se basant sur une résolution fractionnaire.
(B) Utilise une heuristique.
(C) Dans un problème de minimisation consiste à minorer la valeur des solutions partielles.
(D) Dans un problème de minimisation consiste à majorer la valeur des solutions partielles.
(E) Nous donne un algorithme polynomial pour résoudre MAX-2-SAT.
(F) est une variante du backtracking.
(G) s'applique à des problèmes de décision.
(H) fournit des solutions optimales.

37. Quels algorithmes sont des algorithmes gloutons ?

(A) Algorithme de Huffman
(B) Algorithme de Dijkstra
(C) Algorithme de Kruskal
(D) Algorithme de résolution du sac à dos fractionnaire.
(E) Algorithme d'ordonnancement des tâches de durée unaire.

38. Pour lesquels de ces problèmes un algorithme par retour sur trace est-il une solution pertinente ?

(A) Trouver un chemin entre deux points d'un graphe.
(B) Trouver un chemin eulérien dans un graphe.
(C) Trouver un chemin hamiltonien dans un graphe.
(D) Résoudre SAT.
(E) Résoudre MaxSat.
(F) 3-colorier un graphe.

39. Un algorithme de programmation dynamique

(A) Nécessite de stocker des valeurs dans un tableau.
(B) se résoput de manière récursive.
(C) Consiste à partitionner les solutions en sous-problèmes disjoints.
(D) Consiste à formuler la solution d'un problème en fonction de la solution à des sous-problèmes.
(E) utilise plus d'espace pour améliorer la conmplexité en temps.

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

(A) $C(n) = 3C(n-1)$
(B) $C(n) = \sum_{i=0}^{n-i} C(i)$
(C) $C(n) = 3C\left(\frac{n}{2}\right) + O(n^2 \log(n))$
(D) $C(n) = 2C\left(\frac{n}{2}\right) + O(n)$
(E) $C(n) = 2C\left(\frac{n}{2}\right) + \Theta(2^n)$

41. 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 2-SAT en temps polynomial.
(E) Utilise une structure de pile.

42. Lesquels de ces algorithmes permettent de calculer la plus petite distance entre deux points ?

(A) A^*
(B) L'algorithme alpha-beta
(C) L'algorithme de Dijkstra
(D) L'algorithme de Boyer-Moore
(E) L'algorithme de Floyd-Warshall

43. Quelles propriétés sont justes sur les arbres couvrants ?

(A) Tout graphe possède un arbre couvrant.

(B) Si le poids de toutes les arêtes de G sont deux à deux distincts et que G est connexe, G admet un unique arbre couvrant de poids minimum.

(C) Si G admet un unique arbre couvrant de poids minimum, les poids de ses arêtes sont distincts deux à deux.

(D) L'arête de poids minimum appartient à un arbre couvrant de poids minimum.

(E) L'arête de poids maximum n'appartient à aucun arbre couvrant de poids minimum.

44. Quelles propositions sont vraies sur les couplages ?

(A) Un couplage maximal est nécessairement maximum.

(B) Un couplage maximum est nécessairement maximal.

(C) Tout graphe biparti admet un nombre pair de sommets.

(D) Un couplage sans chemin augmentant dans un graphe biparti est maximum.

45. L'algorithme des k plus proches voisins

(A) Est un algorithme d'apprentissage non supervisé.

(B) Gagne en précision lorsqu'on augmente k.

(C) Nécessite que k soit impair.

(D) Aucune des réponses ci-dessus.

46. Un arbre de décision

(A) Est un algorithme d'apprentissage supervisé

(B) Classe correctement tous les éléments de l'ensemble d'apprentissage

(C) Peuvent être amélioré à l'aide d'arbres k-d.

(D) Peut être construit sans calcul d'entropie grâce à l'algorithme ID3

(E) A une hauteur bornée par la dimension des données.

47. 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 des classes non-convexes.

48. La classification obtenue par un algorithme de regroupement hiérarchique ascendant :

(A) Est indépendante de la distance choisie.

(B) Ne permet pas "la remise en question".

(C) A une complexité dépendante de la complexité du calcul de la distance considérée.

(D) Ne reconnaît pas des classes non-convexes.

(E) Nécessite un calcul de médiane.

49. Quel type d'algorithme nous permet de résoudre le problème des 8 reines le plus efficacement ?

(A) Un algorithme min max.

(B) Un algorithme alpha-beta.

(C) L'algorithme des attracteurs.

(D) Un algorithme glouton.

(E) Un algorithme de retour sur trace.

50. Un algorithme reposant sur une heuristique :

(A) Donne une valeur approchée du résultat avec un coefficient d'approximation α.

(B) Ne peut généralement garantir le temps d'exécution et l'optimalité du résultat.

(C) Ne peut 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.

51. Les positions gagnantes pour J_1 dans un jeu

(A) Peuvent être calculées en temps polynomial en la taille du jeu.
(B) Dépendent des réponses de l'adversaire.
(C) Sont nécessairement dans S_1, les sommets contrôlés par J_1
(D) Peuvent mener à une défaite de J_1 suivant sa stratégie.
(E) Aucune des réponses ci-dessus.

52. Dans un jeu à deux joueurs déterministe et à information complète :

(A) certaines positions peuvent être gagnantes pour aucun des deux joueurs.
(B) certaines positions peuvent être gagnantes pour les deux joueurs.
(C) toute position est gagnante pour exactement un des joueurs.

53. L'algorithme de Dijkstra :

(A) est une variante du parcours en profondeur.
(B) est une variante du parcours en largeur.
(C) a une complexité temporelle linéaire en la taillde du graphe.
(D) a une complexité spatiale linéaire en la taillde du graphe.
(E) nécessite une heuristique.

54. L'algorithme de Floyd Warshall :

(A) a une complexité temporelle en $O(n^2)$ où n est le nombre de sommets du graphe.
(B) permet de gérer les arêtes de poids négatifs.
(C) permet de gérer les cycles de poids négatifs.
(D) est bien adapté si on souhaite calculer les distances entre tous les couples de sommets.
(E) nécessite une heuristique.

55. L'algorithme A* :

(A) a une complexité temporelle équivalente à celle de Dijkstra si on utilise une heuristique admissible.
(B) calcule bien les plus courts chemins si on utilise une heuristique admissible.
(C) calcule bien les plus courts chemins si on utilise une heuristique monotone.
(D) est bien adapté si on souhaite calculer les distances entre tous les couples de sommets.
(E) nécessite une heuristique.

56. Un ordre topologique sur un graphe :

(A) est unique s'il existe.
(B) n'a de sens que si le graphe est orienté.
(C) peut être calculé à l'aide d'un parcours en profondeur.
(D) existe pour tout graphe orienté.
(E) peut toujours être calculé pour le graphe des composantes fortement connexe d'un graphe orienté.

57. Pour un graphe à n sommets et p arêtes, la complexité temporelle d'un parcours en profondeur est :

(A) $O(n+p)$ si le graphe est représenté sous forme de listes d'adjacence.
(B) $O(n+p)$ si le graphe est représenté sous forme de matrice d'adjacence.
(C) $O(n^2)$ dans les deux cas.

4 Concurrence et Synchronisation

58. Un fil d'exécution

(A) Termine nécessairement.
(B) Peut partager de la mémoire avec les autres fils d'exécutions.
(C) A une exécution entrelacée avec les autre fils.
(D) N'a pas d'utilité si on ne dispose que d'un seul processeur.
(E) Entraîne un non-déterminisme.

59. L'ordonnanceur est implémenté par le système d'exploitation.

(A) Vrai
(B) Faux

60. La création d'une section critique

(A) Ne peut être assurée sans risque de famine que pour deux fils d'exécution.
(B) Doit satisfaire le principe d'exclusion mutuelle.
(C) Impose une attente active pour certains fils d'exécution.

61. Parmi les problèmes suivants, lequel est le plus grave ?

(A) Famine probabiliste
(B) Famine avérée
(C) Interblocage
(D) Situation de compétition
(E) Ils sont tous aussi grave

62. L'algorithme de la boulangerie de Lamport

(A) Possède un risque de famine avéré.
(B) Nécessite de savoir déterminer la maximum d'un tableau de mnaière atomique.
(C) Utilise l'attente active.
(D) Ne favorise aucun processus.
(E) Nécessite de connaitre le nombre de threads qui voint utiliser le verrou.

63. Une "situation de compétition"se produit quand :

(A) deux fils tentent de lire une même variable en même temps sans synchronisation.
(B) deux fils tentent d'écrire dans une même variable en même temps sans synchronisation.
(C) un fils essaie de lire pendant qu'un autre fils essaie d'écrire dans une même variable.

64. Pour les sémaphores, quels noms d'opérations représentent une incrémentation.

(A) `P(s)`
(B) `init(s))`
(C) `lock(s)`
(D) `wait(s)`
(E) `post(s)`

5 Logique

65. Quelles formules sont des tautologies ?

(A) $((A \to B) \to C) \leftrightarrow (A \to (B \to C))$
(B) $(A \vee B) \wedge (\neg A \vee \neg B)$
(C) $\neg(A \wedge (A \vee B)) \leftrightarrow \neg A$
(D) $((A \vee B) \wedge (C \vee D)) \leftrightarrow (A \wedge C) \vee (B \wedge D)$
(E) $(A \wedge \neg B) \to (\neg A \wedge B)$

66. Une variable libre

(A) Peut-être liée au même endroit dans la formule.
(B) Peut-être liée à un autre endroit dans la formule.
(C) A une portée.
(D) N'a pas d'incidence sur l'évaluation d'une formule.
(E) Est toujours associé à une variable liée.

67. Soit φ et ψ deux formules telles que $\psi \models \varphi$. Cela signifie que :

(A) Que φ est une conséquence de ψ
(B) Que φ est une sous-alternative à ψ
(C) Que les valuations satisfaisantes φ satisfont aussi ψ
(D) Si φ est une antilogie, alors ψ est une antilogie.
(E) La notation $\models$ n'existe que sous la forme $v \models \varphi$ avec v une valuation.

68. Une équivalence entre formules.

(A) Signifie qu'elles ont le même arbre de dérivation.
(B) Peut se prouver par l'étude des tables de vérité.
(C) Peut se prouver à l'aide de substitution dans des formules qu'on sait équivalentes.
(D) Ne peut être démontrée sans la déduction naturelle.
(E) Peut se prouver par une étude de la matrice de confusion.

69. Pour toute formule sans quantificateur, il existe une formule équivalente :

(A) Sous forme normale conjonctive
(B) Sous forme normale disjonctive
(C) Sous forme normale négative
(D) Sous forme normale de Chomsky
(E) Avec quantificateur

70. On sait résoudre le problème SAT en temps polynomial sur des instances

(A) 2-FNC
(B) 3-FNC
(C) 2-FND
(D) 3-FND

71. Une règle dérivée

(A) Peut être utilisée dans une autre preuve.
(B) Peut être prouvée par table de vérité.
(C) Permet d'augmenter le nombre de séquents prouvables.
(D) Aucune des réponses ci-dessus.

72. On peut prouver $A \vdash \neg\neg A$ avec les règles de base de la déduction naturelle (logique intuitionniste).

(A) Vrai
(B) Faux

73. En logique propositionnelle intuitionniste, $\Gamma \vdash C$ si et seulement si $\Gamma \models C$

(A) Vrai
(B) Faux