PROVA 1 — SIMULADO

MAC0338 ANÁLISE DE ALGORITMOS

2º SEMESTRE DE 2026

Nome completo: _____

NUSP: _____

Instruções:

(*1*) Esta prova é individual.
(*2*) A prova pode ser feita a lápis. Cuidado com a legibilidade.
(*3*) A prova consiste de **21** questões (contando a Questão 0 nesta página). Todas as questões têm o mesmo valor.
(*4*) Todas as questões, exceto a Questão 0, são de múltipla escolha. Todas elas têm exatamente uma escolha correta, sendo ela possivelmente NDA, "nenhuma das anteriores".
(*5*) **(Importante!)** Não "chute": você perde **25%** do valor de uma questão para cada resposta errada. Isto é, quatro respostas erradas cancelam uma resposta certa. Questões deixadas em branco valem 0 pontos.
(*6*) Suas respostas devem estar compiladas na página final da prova (preencha a lista de respostas).
(*7*) Não é permitido o uso de aparelhos eletrônicos de qualquer natureza.
(*8*) Não destaque as folhas deste caderno.
(*9*) Não use folhas avulsas para rascunho. Não é necessário apagar seus rascunhos.
(*10*) Não é permitido consultar nenhum material ou consultar colegas.
(*11*) **(Atenção!)** Não é permitido sair da sala sem entregar a prova.

Assinatura:

Sua assinatura acima atesta a autenticidade e originalidade de seu trabalho e que você compromete-se a seguir o código de ética da USP em todas as suas atividades, incluindo esta prova.

Boa Sorte!

Q0. Leia o conteúdo desta página e preencha os itens requisitados. Assine acima, e atente ao significado de sua assinatura.

Data: 2026/10/16, 7:42pm

Q1. A recorrência $T(n) = 4T(\lfloor n/2 \rfloor) + \Theta(n^2)$ tem solução

(*a*) $\Theta(n^2)$
(*b*) $\Theta(n^2 \lg n)$
(*c*) $\Theta(n^3)$
(*d*) $\Theta(n^{5/2})$
(*e*) NDA

Q2. A recorrência $T(n) = T(\lfloor n/2 \rfloor) + T(\lfloor n/4 \rfloor) + n$ (com $T(1) = 1$) tem solução

(*a*) $\Theta(n^{\lg 3})$
(*b*) $\Theta(n \lg n)$
(*c*) $\Theta(n^2)$
(*d*) $\Theta(n)$
(*e*) NDA

Q3. Qual das afirmações abaixo é verdadeira?

(*a*) $\lg(n!) = \Theta(n \lg n)$
(*b*) $n^{1/\lg n} = \Theta(\sqrt{n})$
(*c*) $2^{\sqrt{\lg n}} = \Omega(n^{1/100})$
(*d*) $(\lg n)^{\lg n} = O(n^{10})$
(*e*) NDA

Q4. Suponha que, em toda chamada do QUICKSORT, o pivô deixe pelo menos 1/10 dos elementos do subvetor de cada lado. Então o QUICKSORT consome, no pior caso, tempo

(*a*) $\Theta(n \lg^2 n)$
(*b*) $\Theta(n^2)$
(*c*) $\Theta(n^{1.1})$
(*d*) $\Theta(n \lg n)$
(*e*) NDA

Q5. No QUICKSORT aleatorizado aplicado a n elementos distintos, o número esperado de comparações em que o *menor* elemento está envolvido é

(*a*) $\Theta(\sqrt{n})$
(*b*) $\Theta(1)$
(*c*) $\Theta(\lg n)$
(*d*) $\Theta(n)$
(*e*) NDA

Q6. Um vetor com n elementos que assumem apenas *dois* valores distintos é ordenado pelo QUICKSORT com a partição em três partes de Dijkstra (menores, iguais e maiores que o pivô, recorrendo só nas partes dos menores e dos maiores). O consumo de tempo é

(*a*) $\Theta(n^2)$
(*b*) $\Theta(n \lg n)$
(*c*) $\Theta(n)$
(*d*) $\Theta(\lg n)$
(*e*) NDA

Q7. Considere vetores de n elementos distintos formados por n/k blocos consecutivos de k elementos, em que todo elemento de um bloco é menor que todo elemento do bloco seguinte (mas cada bloco está em ordem arbitrária). O número de comparações que um algoritmo *ótimo* de ordenação por comparações faz, no pior caso, para ordenar tais vetores é

(a) $\Theta(n)$
(b) $\Theta(n \lg k)$
(c) $\Theta(n \lg n)$
(d) $\Theta((n/k) \lg k)$
(e) NDA

Q8. No Counting Sort visto em sala, a última fase percorre A de trás para a frente, colocando cada $A[j]$ na posição $C[A[j]]$ de B e decrementando $C[A[j]]$. Se essa fase percorresse A da frente para trás, sem nenhuma outra mudança, o algoritmo

(a) continuaria ordenando, mas deixaria de ser estável
(b) continuaria ordenando e continuaria estável
(c) deixaria de ordenar corretamente
(d) passaria a consumir tempo $\Theta(n \lg n)$
(e) NDA

Q9. Para ordenar n inteiros em $\{0, \ldots, n^3 - 1\}$, o Radix Sort na base n, usando Counting Sort em cada dígito, consome tempo

(a) $\Theta(n^{3/2})$
(b) $\Theta(n \lg n / \lg \lg n)$
(c) $\Theta(n^3)$
(d) $\Theta(n \lg n)$
(e) NDA

Q10. No Select-MoM, troque os grupos de 5 por grupos de g elementos. Com a mesma análise vista em sala (contando os elementos garantidamente menores e maiores que a mediana das medianas), obtém-se consumo de tempo linear

(a) para $g = 3$, mas não para $g = 7$
(b) apenas para $g = 5$
(c) para $g = 3$ e para $g = 7$
(d) para $g = 7$, mas não para $g = 3$
(e) NDA

Q11. Dado um vetor de n elementos distintos e um inteiro k, queremos os k menores elementos (em qualquer ordem). Qual afirmação é correta?

(a) isso pode ser feito em tempo $O(n)$, para todo k
(b) isso exige tempo $\Omega(n \lg k)$ no pior caso
(c) isso exige ordenar o vetor e, portanto, tempo $\Omega(n \lg n)$
(d) isso pode ser feito em tempo $O(n)$ apenas se $k = O(n / \lg n)$
(e) NDA

Q12. O algoritmo de Strassen visto em sala supõe que n é uma potência de 2. Para multiplicar matrizes $n \times n$ com n qualquer, completamos as duas matrizes com zeros até a ordem $m = 2^{\lceil \lg n \rceil}$, aplicamos o algoritmo e extraímos o bloco $n \times n$ do resultado. O consumo de tempo, em função de n, é

(*a*) $\Theta(n^{\lg 7})$
(*b*) $\Theta(m^{\lg 7})$, que não é $O(n^{\lg 7})$ no pior caso
(*c*) $\Theta(n^3)$, pois m pode ser quase o dobro de n
(*d*) $\Theta(n^{\lg 7} \lg n)$
(*e*) NDA

Q13. Suponha que seja possível multiplicar matrizes 3×3 com k multiplicações (sem usar comutatividade), com $k > 9$, o que dá um algoritmo com $T(n) = kT(n/3) + \Theta(n^2)$. Use, se quiser, as tabelas abaixo (valores aproximados).

t	3	4	5	6	7	8	9	10
$\log_2 t$	1.585	2.000	2.322	2.585	2.807	3.000	3.170	3.322

t	15	16	17	18	19	20	21	22	23	24	25	26
$\log_3 t$	2.465	2.524	2.579	2.631	2.680	2.727	2.771	2.814	2.854	2.893	2.930	2.966

Esse algoritmo é assintoticamente mais rápido que o de Strassen se e somente se

(*a*) $k \leq 26$
(*b*) $k \leq 23$
(*c*) $k \leq 18$
(*d*) $k \leq 21$
(*e*) NDA

Q14. Seja X^R a sequência $X[1\,..\,n]$ invertida. Sobre uma subsequência palindrômica máxima de X, qual afirmação é correta?

(*a*) ela é sempre um segmento (subvetor contíguo) de X
(*b*) seu comprimento é igual ao comprimento de uma subsequência comum máxima de X e X^R
(*c*) seu comprimento é metade do comprimento de uma subsequência comum máxima de X e X^R
(*d*) seu comprimento é sempre pelo menos $n/2$
(*e*) NDA

Q15. No algoritmo de programação dinâmica para a árvore de busca binária ótima com n chaves, que consome tempo $\Theta(n^3)$, seja raiz$[i, j]$ a raiz escolhida para as chaves $i, \ldots, j$. Knuth provou que raiz$[i, j-1] \leq$ raiz$[i, j] \leq$ raiz$[i+1, j]$. Se, para cada par $i < j$, procurarmos a raiz apenas nesse intervalo, o consumo de tempo passa a ser

(*a*) $\Theta(n \lg n)$
(*b*) $\Theta(n^3)$: o intervalo pode ter $\Theta(n)$ chaves
(*c*) $\Theta(n^2 \lg n)$
(*d*) $\Theta(n^{5/2})$
(*e*) NDA

Q16. Na mochila booleana, seja $t[i, w]$ o valor máximo que se obtém com os itens $1, \ldots, i$ e capacidade w; o item i tem peso w_i e valor v_i. Para $w_i \leq w$, vale

(a) $t[i, w] = \max\{t[i-1, w],\ t[i-1, w-w_i]\}$
(b) $t[i, w] = v_i + t[i-1, w-w_i]$
(c) $t[i, w] = \max\{t[i-1, w],\ v_i + t[i-1, w-w_i]\}$
(d) $t[i, w] = \max\{t[i-1, w],\ v_i + t[i, w-w_i]\}$
(e) NDA

Q17. Na árvore de Huffman para 6 símbolos com frequências $1, 1, 2, 3, 5, 8$, o comprimento máximo de uma palavra do código é

(a) 4
(b) 5
(c) 6
(d) 3
(e) NDA

Q18. Temos n tarefas, todas de duração 1, com prazos inteiros $d_1, \ldots, d_n$, executadas uma de cada vez a partir do instante 0. Uma tarefa é *pontual* se termina até o seu prazo. Para maximizar o *número* de tarefas pontuais,

(a) nenhum critério guloso é ótimo; é preciso programação dinâmica
(b) basta executar as tarefas em ordem crescente de prazo
(c) percorremos as tarefas em ordem crescente de prazo, aceitando cada uma se o número de tarefas já aceitas é menor que o seu prazo; executamos as aceitas nessa ordem e as demais no final
(d) basta executar as tarefas em ordem decrescente de prazo
(e) NDA

Q19. Seja G um grafo conexo com custos *distintos* nas arestas. Considere as afirmações:

(I) a aresta de custo máximo de qualquer circuito de G não pertence à árvore geradora mínima;
(II) a aresta de custo mínimo de qualquer circuito de G pertence à árvore geradora mínima;
(III) os algoritmos de Kruskal e de Prim devolvem a mesma árvore.

Quais delas são verdadeiras?

(a) apenas I
(b) apenas III
(c) apenas I e III
(d) I, II e III
(e) NDA

Q20. Aplicando o algoritmo da paciência (cada c_i vai para a pilha mais à esquerda cujo topo é maior que c_i, ou para uma pilha nova) à sequência $5, 2, 8, 6, 3, 7, 4, 9$, o número de pilhas no final é

(a) 6
(b) 3
(c) 5
(d) 2
(e) NDA

MAC0338 Análise de Algoritmos — Prova 1 — Simulado

Folha de respostas

Nome completo: _____

NUSP: _____

Marque com um **X** a alternativa escolhida em cada questão (no máximo uma por questão). Somente as respostas marcadas nesta folha serão consideradas.

Q	*a*	*b*	*c*	*d*	*e*
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					

Q	*a*	*b*	*c*	*d*	*e*
11					
12					
13					
14					
15					
16					
17					
18					
19					
20					

Assinatura:

Sua assinatura acima atesta a autenticidade e originalidade de seu trabalho e que você compromete-se a seguir o código de ética da USP em todas as suas atividades, incluindo esta prova.