E-fólio Global | Instruções para a realização do E-fólio

ABERTA

Estruturas de Dados e Algoritmos Fundamentais | 21046

Data de Realização

Dia 6 de junho de 2024

Duração da prova

90 minutos, mais 60 minutos de tolerância

Trabalho a desenvolver

Responder às questões dos Grupos I a III do enunciado que lhe foi atribuído.

Leia estas informações e instruções na totalidade antes de iniciar a resolução da prova.

Critérios de avaliação e cotação

- As cotações são indicadas por grupo e nas próprias questões.
- Esta é uma prova de avaliação individual. As respostas às questões devem fazer sentido, ser coerentes e constituídas por palavras próprias do aluno. Não serão aceites transcrições ou traduções de livros e textos, incluindo textos de orientações de respostas de provas anteriores. As respostas que não respeitem estas condições serão classificadas com zero valores ou fortemente desvalorizadas.
- Exceto indicação contrária, todas as respostas devem ser relativamente desenvolvidas e elaboradas de modo a demonstrar o raciocínio e conhecimento que leva à resposta final. A clareza do texto e da explicação também são levadas em conta.
- Nas questões de escrita de programas, a sua correção tem em conta critérios de proficiência e compreensibilidade do código tais como: legibilidade, indentação, estrutura, comentários e explicação geral do seu funcionamento.
- Apenas é permitida a consulta dos materiais recomendados na unidade curricular.

Instruções e Normas a respeitar

- Este documento tem 4 enunciados: E0, E1, E2 e E3. Calcule o resto R da divisão inteira do seu nº de estudante por 4 e resolva o respetivo enunciado ER. Apenas é necessário considerar os últimos 2 dígitos (dezenas e unidades). Exemplo: Se o seu nº de estudante é -----38, então o resto inteiro de 38/4=9 é R=2 e é-lhe atribuído o enunciado E2.

- Deve redigir o seu e-fólio no ficheiro Folha de Resolução disponibilizado na turma e preencher todos os dados do cabeçalho, incluindo a indicação do enunciado a resolver que lhe é atribuído, em função do nº de estudante. A resolução de outro enunciado é classificada com 0 valores.

- O texto de todas as respostas deve ser introduzido pelo processador de texto, incluindo código de programas, não sendo aceites respostas escritas à mão ou por outros meios, digitalizadas e incluídas no ficheiro como imagens. São exceções figuras, diagramas e expressões matemáticas mais complicadas, desde que sejam todas de autoria do aluno, devendo ter legenda ou identificação de modo a serem referidas nos textos explicativos.

- No caso de código de programas é obrigatório a sua introdução pelo processador de texto utilizando uma fonte monoespaço (por exemplo Courier New).

- Todas as páginas do documento devem ser numeradas.

- O seu e-fólio deve ser constituído por páginas A4, redigidas com tamanho de letra 12. O espaçamento entre linhas deve corresponder a 1,5 linhas, exceto no caso de código de programas. O formato final do ficheiro deve ser exclusivamente em formato pdf, sem restrições (destrancado). Não serão aceites outros tipos de ficheiro.

- Nomeie o ficheiro com o seu número de estudante, seguido da identificação do e-fólio, seguido do enunciado que lhe foi atribuido, segundo o exemplo apresentado: 000000efolioGlobalE0.pdf.

- Deve carregar o referido ficheiro pdf para a plataforma no dispositivo E-fólio Global até à data e hora limite de entrega. Evite a entrega próximo da hora limite para se precaver contra eventuais problemas na composição do documento, conversão para formato pdf e submissão do ficheiro.

- O ficheiro a entregar não deve exceder 8 MB.

Votos de bom trabalho!

Enunciado E0

Grupo I [3 valores]

1.1. [1] Utilizando a definição, prove que $f(n) = n^2 \log(n^2) + n^3 + 10$ é $O(n^3)$.

1.2. Para cada um dos seguintes pares de funções $f(n)$ e $g(n)$, indique se $f(n) = O(g(n))$, $f(n) = \Omega(g(n))$, $f(n) = \Theta(g(n))$ ou nenhum dos casos. Justifique a sua resposta com base apenas na ordem de grandeza relativa das funções.

1.2.1. [0.5] $f(n) = 2^n + n$, $g(n) = n^4 + 1000$

1.2.2. [0.5] $f(n) = \sqrt{n} + 2^n + 200$, $g(n) = 3^n + 10$

1.3. [1] Considere a complexidade do seguinte segmento de código em termos do nº $f(n)$ de operações aritméticas realizadas na variável a. Determine justificando a expressão de $f(n)$ e indique a sua complexidade na notação $O(.)$.

```
for(a=0,i=1; i<=n; i*=2)
   for(j=i; j<=n; ++j)
      a++;
```

Grupo II [5 valores]

2.1. Considere uma árvore de promoção (Splay Tree) inicialmente vazia.
Nota: A inserção/remoção numa árvore de promoção é considerada igual à efetuada numa árvore de pesquisa binária (BST) regular.

2.1.1. [1] Insira na árvore as chaves 8, 12, 10, 4, 14, 2, 6, 7, 5, 3 pela ordem indicada. Desenhe a árvore obtida após efetuadas todas as inserções (total de 1 desenho). Justifique os passos intermédios / raciocínio apenas para as duas últimas inserções.
Nas alíneas seguintes considere a árvore obtida como a árvore "original".

2.1.2. [1] Remova da árvore original a chave 4 utilizando o algoritmo de remoção por cópia (Deletion by Copying) com a chave sucessora. Desenhe a árvore obtida justificando os passos intermédios / raciocínio.

2.1.3. [1] Considere que na árvore original foi efetuado um acesso à chave 5. Desenhe a árvore obtida após o acesso justificando os passos intermédios / raciocínio.

2.2. [2] Considere uma árvore B (B-Tree) de ordem 3 inicialmente contendo apenas o nó raiz com a chave 7. Desenhe a árvore após cada uma das seguintes operações de inserção (I) e remoção (R) pela ordem indicada: I 3, 1, 4, 9, 10; R 1; (total de 6 desenhos). Justifique os passos intermédios / raciocínio para cada operação.

Enunciado E0

Grupo III [4 valores]

3.1. [2] Considere uma tabela T de dispersão (hash) com dimensão 11 e função de hash $h(x) = x \bmod 11$. As colisões são resolvidas com sondagem (probing) linear. Considerando a tabela inicialmente vazia, determine o conteúdo final da tabela após a inserção das chaves 3, 15, 19, 4, 14, 17, 11 pela ordem apresentada. Justifique os cálculos efetuados para cada inserção.

3.2. [2] Considere o vetor [3 7 6 9 5 1 4 2 8]. Ordene o vetor, utilizando o algoritmo de ordenação Comb Sort, apresentando e justificando os passos intermédios durante a ordenação.

Enunciado E1

Grupo I [4 valores]

1.1. [2] Considere uma tabela T de dispersão (hash) com dimensão 9 e função de hash $h(x) = x \bmod 9$. As colisões são resolvidas com sondagem (probing) linear. Considerando a tabela inicialmente vazia, determine o conteúdo final da tabela após a inserção das chaves 13, 17, 3, 4, 12, 15, 9 pela ordem apresentada. Justifique os cálculos efetuados para cada inserção.

1.2. [2] Considere o vetor [2 9 8 6 5 4 1 3 7]. Ordene o vetor, utilizando o algoritmo de ordenação Comb Sort, apresentando e justificando os passos intermédios durante a ordenação.

Grupo II [3 valores]

2.1. [1] Utilizando a definição, prove que $f(n) = n^2 \log(n^2) + n^3 + 10$ é $O(n^3)$.

2.2. Para cada um dos seguintes pares de funções $f(n)$ e $g(n)$, indique se $f(n) = O(g(n))$, $f(n) = \Omega(g(n))$, $f(n) = \Theta(g(n))$ ou nenhum dos casos. Justifique a sua resposta com base apenas na ordem de grandeza relativa das funções.

2.2.1. [0.5] $f(n) = 2^n + n$, $g(n) = n^4 + 1000$

2.2.2. [0.5] $f(n) = \sqrt{n} + 2^n + 200$, $g(n) = 3^n + 10$

2.3. [1] Considere a complexidade do seguinte segmento de código em termos do nº $f(n)$ de operações aritméticas realizadas na variável a. Determine justificando a expressão de $f(n)$ e indique a sua complexidade na notação $O(.)$.

```
for(a=0,i=1; i<=n; i*=2)
   for(j=i; j<=n; ++j)
      a++;
```

Enunciado E1

Grupo III [5 valores]

3.1. Considere uma árvore de promoção (Splay Tree) inicialmente vazia.
Nota: A inserção/remoção numa árvore de promoção é considerada igual à efetuada numa árvore de pesquisa binária (BST) regular.

3.1.1. [1] Insira na árvore as chaves 19, 7, 15, 3, 25, 29, 12, 17, 23, 4 pela ordem indicada. Desenhe a árvore obtida após efetuadas todas as inserções (total de 1 desenho). Justifique os passos intermédios / raciocínio apenas para as duas últimas inserções.
Nas alíneas seguintes considere a árvore obtida como a árvore "original".

3.1.2. [1] Remova da árvore original a chave 7 utilizando o algoritmo de remoção por cópia (Deletion by Copying) com a chave antecessora. Desenhe a árvore obtida justificando os passos intermédios / raciocínio.

3.1.3. [1] Considere que na árvore original foi efetuado um acesso à chave 17. Desenhe a árvore obtida após o acesso justificando os passos intermédios / raciocínio.

3.2. [2] Considere uma árvore B (B-Tree) de ordem 3 inicialmente contendo apenas o nó raiz com a chave 6. Desenhe a árvore após cada uma das seguintes operações de inserção (I) e remoção (R) pela ordem indicada: I 13, 2, 8, 17, 10; R 17; (total de 6 desenhos). Justifique os passos intermédios / raciocínio para cada operação.

Enunciado E2

Grupo I [5 valores]

1.1. Considere uma árvore de promoção (Splay Tree) inicialmente vazia.
Nota: A inserção/remoção numa árvore de promoção é considerada igual à efetuada numa árvore de pesquisa binária (BST) regular.

1.1.1. [1] Insira na árvore as chaves 8, 12, 10, 4, 14, 2, 6, 7, 5, 3 pela ordem indicada. Desenhe a árvore obtida após efetuadas todas as inserções (total de 1 desenho). Justifique os passos intermédios / raciocínio apenas para as duas últimas inserções.
Nas alíneas seguintes considere a árvore obtida como a árvore "original".

1.1.2. [1] Remova da árvore original a chave 4 utilizando o algoritmo de remoção por cópia (Deletion by Copying) com a chave sucessora. Desenhe a árvore obtida justificando os passos intermédios / raciocínio.

1.1.3. [1] Considere que na árvore original foi efetuado um acesso à chave 5. Desenhe a árvore obtida após o acesso justificando os passos intermédios / raciocínio.

1.2. [2] Considere uma árvore B (B-Tree) de ordem 3 inicialmente contendo apenas o nó raiz com a chave 7. Desenhe a árvore após cada uma das seguintes operações de inserção (I) e remoção (R) pela ordem indicada: I 3, 1, 4, 9, 10; R 1; (total de 6 desenhos). Justifique os passos intermédios / raciocínio para cada operação.

Grupo II [4 valores]

2.1. [2] Considere uma tabela T de dispersão (hash) com dimensão 9 e função de hash $h(x) = x \bmod 9$. As colisões são resolvidas com sondagem (probing) linear. Considerando a tabela inicialmente vazia, determine o conteúdo final da tabela após a inserção das chaves 13, 17, 3, 4, 12, 15, 9 pela ordem apresentada. Justifique os cálculos efetuados para cada inserção.

2.2. [2] Considere o vetor [2 9 8 6 5 4 1 3 7]. Ordene o vetor, utilizando o algoritmo de ordenação Comb Sort, apresentando e justificando os passos intermédios durante a ordenação.

Enunciado E2

Grupo III [3 valores]

3.1. [1] Utilizando a definição, prove que $f(n) = n^2 + \log(n^n) + 4$ é $O(n^2)$.

3.2. Para cada um dos seguintes pares de funções $f(n)$ e $g(n)$, indique se $f(n) = O(g(n))$, $f(n) = \Omega(g(n))$, $f(n) = \Theta(g(n))$ ou nenhum dos casos. Justifique a sua resposta com base apenas na ordem de grandeza relativa das funções.

3.2.1. [0.5] $f(n) = \sqrt{n} + 2^n + 200,\ \ g(n) = n^4 + 1000$

3.2.2. [0.5] $f(n) = 2^n + n,\ \ g(n) = 3^n + 10$

3.3. [1] Considere a complexidade do seguinte segmento de código em termos do nº $f(n)$ de operações aritméticas realizadas na variável a. Determine justificando a expressão de $f(n)$ e indique a sua complexidade na notação $O(.)$.

```
for(a=0,i=1; i<=n*n; i*=2)
   for(j=i; j<=n; ++j)
      a++;
```

Enunciado E3

Grupo I [4 valores]

1.1. [2] Considere uma tabela T de dispersão (hash) com dimensão 11 e função de hash $h(x) = x \bmod 11$. As colisões são resolvidas com sondagem (probing) linear. Considerando a tabela inicialmente vazia, determine o conteúdo final da tabela após a inserção das chaves 3, 15, 19, 4, 14, 17, 11 pela ordem apresentada. Justifique os cálculos efetuados para cada inserção.

1.2. [2] Considere o vetor [3 7 6 9 5 1 4 2 8]. Ordene o vetor, utilizando o algoritmo de ordenação Comb Sort, apresentando e justificando os passos intermédios durante a ordenação.

Grupo II [5 valores]

2.1. Considere uma árvore de promoção (Splay Tree) inicialmente vazia.
Nota: A inserção/remoção numa árvore de promoção é considerada igual à efetuada numa árvore de pesquisa binária (BST) regular.

2.1.1. [1] Insira na árvore as chaves 19, 7, 15, 3, 25, 29, 12, 17, 23, 4 pela ordem indicada. Desenhe a árvore obtida após efetuadas todas as inserções (total de 1 desenho). Justifique os passos intermédios / raciocínio apenas para as duas últimas inserções.
Nas alíneas seguintes considere a árvore obtida como a árvore "original".

2.1.2. [1] Remova da árvore original a chave 7 utilizando o algoritmo de remoção por cópia (Deletion by Copying) com a chave antecessora. Desenhe a árvore obtida justificando os passos intermédios / raciocínio.

2.1.3. [1] Considere que na árvore original foi efetuado um acesso à chave 17. Desenhe a árvore obtida após o acesso justificando os passos intermédios / raciocínio.

2.2. [2] Considere uma árvore B (B-Tree) de ordem 3 inicialmente contendo apenas o nó raiz com a chave 6. Desenhe a árvore após cada uma das seguintes operações de inserção (I) e remoção (R) pela ordem indicada: I 13, 2, 8, 17, 10; R 17; (total de 6 desenhos). Justifique os passos intermédios / raciocínio para cada operação.

Enunciado E3

Grupo III [3 valores]

3.1. [1] Utilizando a definição, prove que $f(n) = n^2 + \log(n^n) + 4$ é $O(n^2)$.

3.2. Para cada um dos seguintes pares de funções $f(n)$ e $g(n)$, indique se $f(n) = O(g(n))$, $f(n) = \Omega(g(n))$, $f(n) = \Theta(g(n))$ ou nenhum dos casos. Justifique a sua resposta com base apenas na ordem de grandeza relativa das funções.

3.2.1. [0.5] $f(n) = \sqrt{n} + 2^n + 200,\ \ g(n) = n^4 + 1000$

3.2.2. [0.5] $f(n) = 2^n + n,\ \ g(n) = 3^n + 10$

3.3. [1] Considere a complexidade do seguinte segmento de código em termos do nº $f(n)$ de operações aritméticas realizadas na variável a. Determine justificando a expressão de $f(n)$ e indique a sua complexidade na notação $O(.)$.

```
for(a=0,i=1; i<=n*n; i*=2)
   for(j=i; j<=n; ++j)
      a++;
```

FIM