UNIVERSIDADE DO ALGARVE
FACULDADE DE CIÊNCIAS E TECNOLOGIA

Algoritmos e Estruturas de Dados

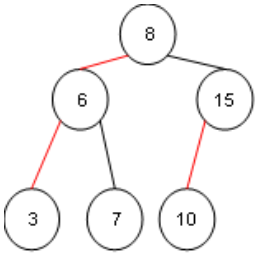
A • Exame Época Normal • 31 de janeiro de 2022 • 16:00 – 18:00

- Para perguntas com resposta de escolha múltipla, respostas erradas com cotação c e n respostas possíveis descontam $-c / (n - 1)$.
- As respostas às perguntas de desenvolvimento deverão ser as mais detalhadas possíveis, indicando os vários passos intermédios da resolução.
- Na mesa em que está a fazer o exame deve ter apenas lápis/caneta, identificação, este enunciado, e folha de teste.
- Identifique cada folha de teste com o seu nome e número, em letra bem legível.

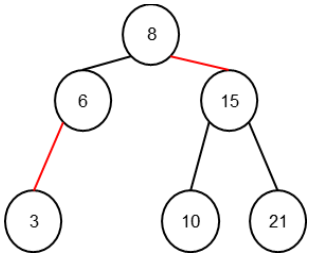
Grupo I: Perguntas de escolha múltipla

1. (0.5) Qual das seguintes árvores é uma árvore red-black válida:

a)

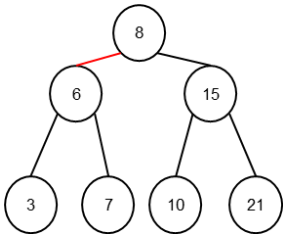


b)

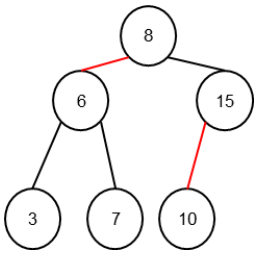


c) Nenhuma das árvores apresentadas é uma árvore red-black válida

d)



e)



2. (0.5) Considere o algoritmo de ordenação Shellsort usando a sequência de Knuth aplicado ao seguinte array [4, 2, 5, 0, 1, 7, 3, 8, 9, 6]. Indique qual das seguintes sequências corresponde a uma sequência válida e pela ordem correta do estado do array após operações de h-sorting. Assuma que h inicial = 4.

Selecione uma opção de resposta:

a)
[1, 2, 3, 0, 4, 6, 5, 8, 9, 7]
[1, 0, 3, 2, 4, 6, 5, 7, 9, 8]
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

b)
[0, 1, 5, 3, 2, 7, 4, 8, 9, 6]
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

c)
[0, 1, 5, 3, 2, 7, 4, 8, 9, 6]
[0, 1, 2, 3, 4, 6, 5, 7, 9, 8]
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

d)
[1, 2, 3, 0, 4, 6, 5, 8, 9, 7]
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

3. (0.5) Considere o seguinte código:

```
int sum = 0;
for(int i = 0; i < n; i++ )
{
   for(int j = 1; j < n-1 ; j++)
   {
      sum++;
   }
}
```

Indique a complexidade temporal do código apresentado.

Selecione uma opção de resposta:

a) $O(n^3)$
b) O(n log n)
c) O(log n)
d) O(n)
e) $O(n^2)$

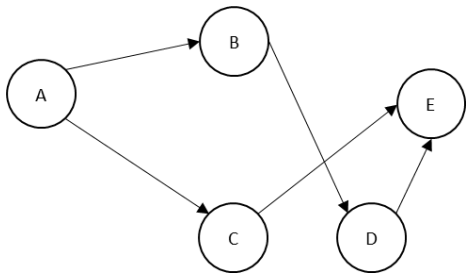
4. (0.5) Indique a aproximação tilde mais correta para a seguinte expressão:

$$\frac{3n + logn + 2n^3 \log n}{2log\ n}$$

Selecione uma opção de resposta:

a) ~1
b) $\sim\frac{3n+1+2n^3}{2\log n}$
c) $\sim 2n^3 \log n.$
d) $\sim n^3$

5. (0.5) Considere o grafo dirigido representado abaixo. Quantas ordenações topológicas distintas existem para o grafo?



Selecione uma opção de resposta:

a) nenhuma
b) 1
c) 2
d) 3
e) 4

Grupo II: Perguntas de desenvolvimento

6. (2.5) Escreva um método estático chamado somaNPares em linguagem Java, que dado um inteiro n, e um array de inteiros a, retorna um inteiro que corresponde à soma dos primeiros n elementos de a que sejam pares (caso existam menos pares do que n, são somados apenas os pares existentes).

Por exemplo, se n=3, a=[1,2,3,4,5], então a função devolve 6. Se n=3, e a=[8,6,5,4,3,2,1] a função devolve 8 + 6 + 4 = 18.

7. (2.0) Considere uma pilha (*stack*) e uma fila (*queue*) inicialmente vazias, sobre as quais é executada a seguinte sequência de instruções:

```
StackList<Integer> s = new StackList<Integer>();
QueueList<Integer> q = new QueueList<Integer>();
int j;

s.push(10);
s.push(5);
s.push(2);
s.push(1);

q.enqueue(3);
for(int i = 0; i<4; i++)
{
    q.enqueue(s.pop());
    q.enqueue(i);
    s.push(q.dequeue());
}
while(!q.isEmpty()) System.out.println(q.dequeue() + "+");
```

Represente a fila q e a pilha s, com os elementos introduzidos e removidos ao longo da execução do código e indique a sequência impressa no ecrã quando o código é executado.

8. (3.0) Resolva os seguintes exercícios sobre montes (heaps).

a) (2.0) Desenhe a representação em árvore dos vários estados de um *max-heap,* obtido quando se insere os seguintes elementos num *heap* inicialmente vazio: 4, 9, 1, 0, 7, 5, 3. Desenhe pelo menos uma árvore por cada troca de elementos.

b) (1.0) Desenhe a representação em árvore dos vários estados de um *max-heap,* obtido quando se remove os dois maiores elementos do *heap* obtido na alínea anterior. Desenhe pelo menos uma árvore por cada troca de elementos

9. (2.5) Considere o método *partition* usado no algoritmo de ordenação ***quicksort tripartido***, com a assinatura *int[] partition(Comparable[] a, int low, int high)*. Suponha que o método *partition é* invocado com os seguintes argumentos:

a=[5, 2, 3, 9, 5, 8, 6, 5, 2], low = 0, high=8

Indique os vários passos do método *partition,* indicando o conteúdo do *array* após cada troca de elementos, assumindo que o pivô é escolhido como sendo o elemento da posição *low*. Indique também qual a posição das duas flags, *lt* (lesser than) e *gt* (greater than) retornadas pelo método.

10. (2.5) Considere uma árvore rubro-negra (red-black) inicialmente vazia, onde ligações vermelhas são representadas como linhas a tracejado e ligações negras como linhas contínuas. Desenhe a representação da árvore, depois de inseridos os elementos no array indicado abaixo, da esquerda para a direita. Desenhe os principais passos intermédios, e indique explicitamente quando efectuar uma das seguintes operações: Rotate Left, Rotate Right, e Flip.

[3, 2, 4, 1, 6, 5]

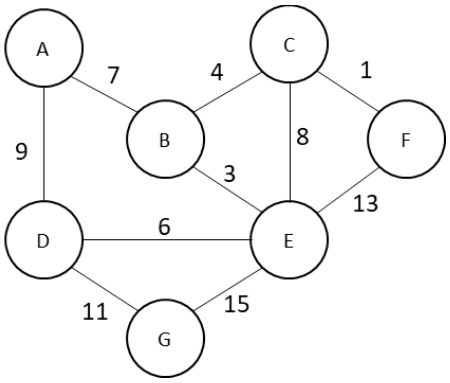
11. (2.0) Considere uma tabela de dispersão de dimensão M = 9, com resolução de colisões por dupla dispersão, e funções de dispersão h1(k) = k mod M e h2(k) = 5 – k mod 5

Sabendo que que inicialmente a tabela se encontra preenchida com as chaves indicadas, indique quais os elementos presentes em cada posição do array de chaves e de valores, apos a inserção da sequência de pares <chave , valor> abaixo. **Indique o valor de h1 e h2 calculado para cada chave**. Indique uma posição vazia com ▬.

<8 , “c”>, <1 , “d”>, <3 , “e”>, < 4, “f”>, <9 , “g”> ,

i	0	1	2	3	4	5	6	7	8
chaves		1	11	12			15		
valores		“b”	“h”	“x”			“a”		

12. (2.0) Considere o seguinte grafo pesado não dirigido. Considere a execução do algoritmo de Prim sobre o grafo pesado não dirigido da figura abaixo. Qual a sequência pela qual o algoritmo de Prim visita as arestas de corte, assumindo que o algoritmo inicia a procura a partir do vértice A? (Em caso de empate, utilize a ordem alfabética dos vértices para desempatar). Desenhe a árvore de cobertura mínima obtida pelo algoritmo.



13. (1.0) Considere o algoritmo *MaxCycleMST* implementado no 4.º projeto. Uma das otimizações possíveis para a versão base deste algoritmo é a seguinte:

- Ordenar uma lista (ou fila prioritária) com todos os arcos do grafo original G, do arco de menor custo para o arco de maior custo.
- Utilizar a ordem da lista anterior para adicionar os arcos à árvore mínima abrangente (MST) a ser construída.
- Se ao adicionar um novo arco à MST for detetada a existência de um ciclo, remover o arco acrescentado.

Indique qual a vantagem desta variante relativamente à versão base, e explique porque é que esta variante é equivalente à versão base do algoritmo.