Usuário	
Curso	ESTRUTURAS DE DADOS
Teste	QUESTIONÁRIO UNIDADE III
Iniciado	
Enviado	
Status	Completada
Resultado da tentativa	4 em 4 pontos
Tempo decorrido	
Resultados exibidos	Todas as respostas, Respostas enviadas, Respostas corretas, Comentários, Perguntas respondidas incorretamente

Pergunta 1

0,4 em 0,4 pontos

Com relação à recursividade, assinale a alternativa correta:

Resposta Selecionada: c. As funções recursivas têm duas partes fundamentais: a regra geral e o ponto de parada.

Respostas:

a. O ponto de parada é resolvido com recursividade, sendo este ponto raramente um limite inferior ou superior.

b. O caso geral consiste na ampliação do problema por meio da invocação recursiva de casos mais gerais.

c. As funções recursivas têm duas partes fundamentais: a regra geral e o ponto de parada.

d. As informações guardadas na invocação de uma função recursiva são: endereços de memória, estado da memória RAM, ponteiros e variável de retorno.

e. Uma função recursiva é também chamada de iterativa.

Comentário da resposta:

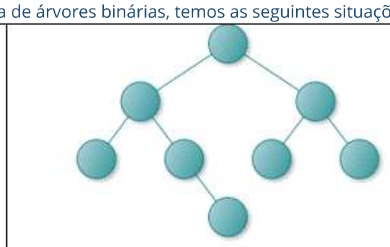
Resposta: C
Comentário:
As funções recursivas têm duas características importantes:
• Ponto de parada: geralmente, um limite superior ou inferior da regra geral;
• Regra geral: reduz a resolução do problema por meio da invocação recursiva de casos menores, resolvidos mediante a resolução de casos ainda menores, e assim sucessivamente, até atingir o ponto de parada, que finaliza o método.
As funções que não são recursivas são chamas de iterativas.

Pergunta 2

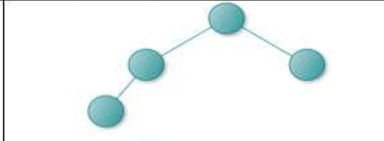
0,4 em 0,4 pontos

Sobre altura de árvores binárias, temos as seguintes situações:

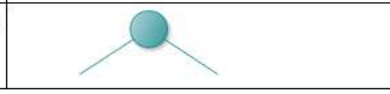
I.



II.



III.



Assinale a alternativa com as alturas corretas de I, II e III respectivamente.

Resposta Selecionada: e. 3,2,0.

Respostas:

a. 4,3,1.

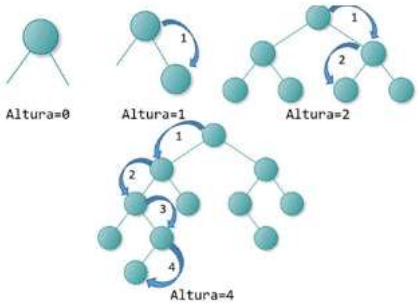
b. 4,3,0.

c. 3,2,1.

d. 4,2,1.

e. 3,2,0.

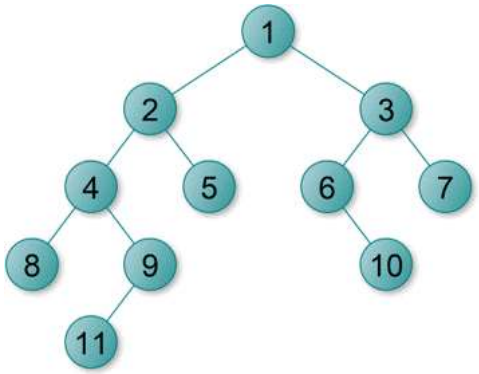
Comentário da resposta: Resposta: E
Comentário:
Em uma árvore binária, o caminho percorrido para sair de um nó e chegar a outro sempre será único. Assim, a propriedade fundamental da árvore binária é que, da raiz até qualquer um dos seus nós, haverá somente um caminho. A quantidade de nós ou saltos percorridos da raiz (sem contá-la) até a folha mais distante determina a altura da árvore. Uma árvore apenas com a raiz tem altura zero.



Pergunta 3

0,4 em 0,4 pontos

Considere a árvore:



Assinale a alternativa com o percurso em ordem (infixo) e pós-ordem (pós-fixo):

Resposta Selecionada:
a. 8 4 11 9 2 5 1 6 10 3 7
8 11 9 4 5 2 10 6 7 3 1

Respostas:
a. 8 4 11 9 2 5 1 6 10 3 7
8 11 9 4 5 2 10 6 7 3 1

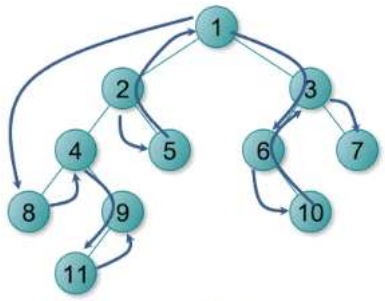
b. 8 4 11 9 5 2 1 10 6 3 7
8 4 9 11 5 2 1 3 6 10 7

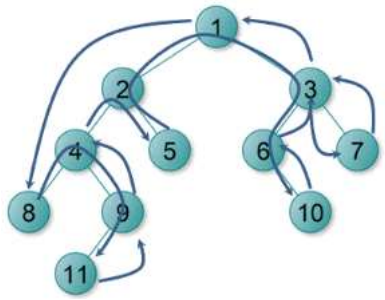
c. 8 4 11 9 2 5 1 6 10 3 7
8 4 9 11 5 2 1 3 6 10 7

d. 8 4 11 9 5 2 1 10 6 3 7
8 11 9 4 5 2 10 6 7 3 1

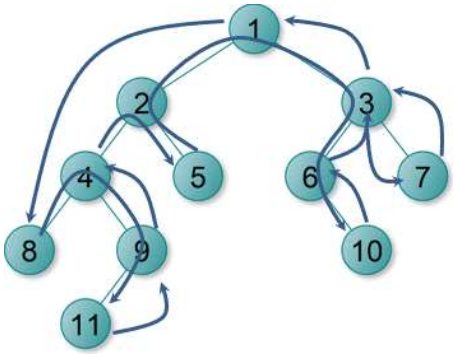
e. 1 2 4 8 9 11 5 3 6 10 7
8 11 9 4 5 2 10 6 7 3 1

Comentário da resposta: Resposta: A
Comentário
In order. ELDr





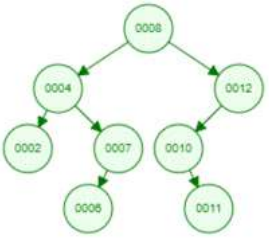
8 4 11 9 2 5 1 6 10 3 7
Pós-ordem: EDLr



8 11 9 4 5 2 10 6 7 3 1

Pergunta 4 0,4 em 0,4 pontos

A árvore abaixo é uma árvore binária de busca:



Assinale a alternativa que **não** monta esta árvore:

Resposta Selecionada: c. 8 4 2 7 6 12 11 10.

Respostas:

a. 8 4 12 2 7 6 10 11.

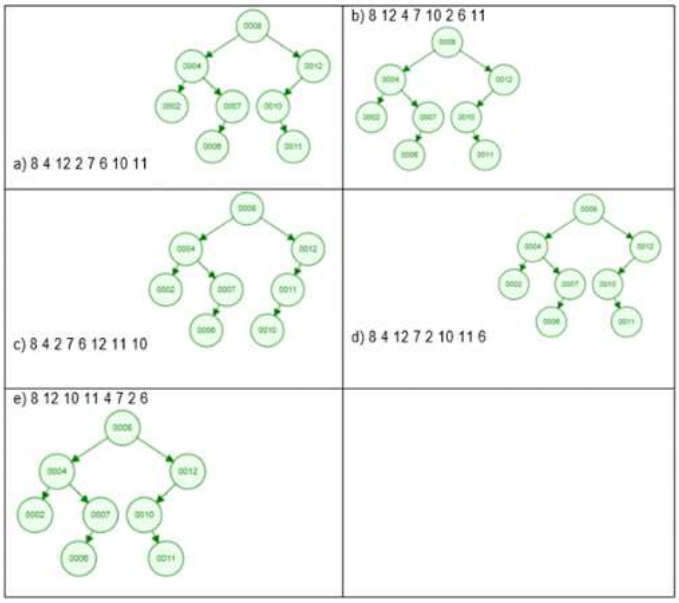
b. 8 12 4 7 10 2 6 11.

c. 8 4 2 7 6 12 11 10.

d. 8 4 12 7 2 10 11 6.

e. 8 12 10 11 4 7 2 6.

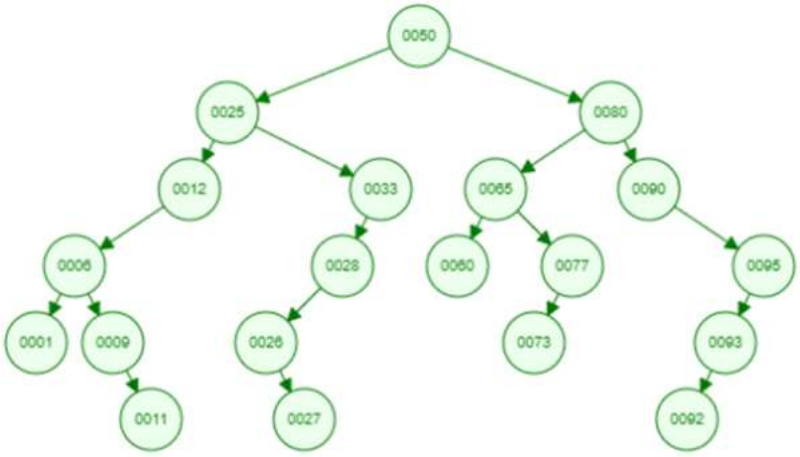
Comentário da resposta: Resposta: C
Comentário:
Se a chave a ser inserida for menor que a chave do nó analisado, inseriremos a chave na subárvore esquerda;
• Se a chave a ser inserida for maior que a chave do nó analisado, inseriremos a chave na subárvore direita;
• Se a subárvore estiver ocupada, seguiremos com a busca.
Montando cada uma das árvores.



Somente C monta árvore diferente.

Pergunta 5 0,4 em 0,4 pontos

Considere a Árvore Binária de Busca abaixo:



Ao remover os nós 25 e 80, assinale a alternativa com os valores que podem substitui-los mantendo a integridade da árvore binária de busca:

Resposta Selecionada: d. 12, 26 e 77, 90.

Respostas:
a. 12, 33 e 65, 90.
b. 11, 26 e 77, 92.
c. 12, 27 e 73, 90.
d. 12, 26 e 77, 90.
e. 11, 27 e 73, 92.

Comentário da resposta:
Resposta: D
Comentário:
• Remoção na folha, basta remover o nó da árvore;
• Remoção de um nó com um filho. Ao excluir o nó, o filho sobe para a posição do pai;
• Remoção de um nó com dois filhos. O nó a ser retirado é substituído pelo nó mais à direita da subárvore esquerda ou o nó a ser retirado é substituído pelo nó mais à esquerda da subárvore direita.
Conforme a observação: o maior valor do lado esquerdo ou menor valor do lado direito.
Remoção do 25, maior valor do lado direito 12, menor valor do lado direito 26.
Remoção do 80, maior valor do lado direito 77, menor valor do lado direito 90.

Pergunta 6 0,4 em 0,4 pontos

Dadas as seguintes afirmações em relação à Busca Binária:

I. O vetor necessita estar ordenado para realizar a busca binária.
II. Em uma busca binária, é possível ignorar partes do vetor.
III. A implementação da busca binária é bastante flexível, pode ser recursiva ou iterativa.

Assinale a alternativa correta em relação às afirmações acima:

Resposta Selecionada: d. Mais de uma das afirmações está correta.

Respostas:

a. Apenas a afirmação I é correta.

b. Apenas a afirmação II é correta.

c. Apenas a afirmação III é correta.

d. Mais de uma das afirmações está correta.

e. Nenhuma das afirmações está correta.

Comentário da resposta: Resposta: D
Comentário:
- A busca ou pesquisa binária é realizada num vetor ordenado. Ela implementa o paradigma conhecido como divisão e conquista (Divide and Conquer).
- Caso o valor seja menor, ignoramos o resto do vetor que seja maior que este valor e buscar somente na metade "menor" do vetor.

Pergunta 7

0,4 em 0,4 pontos

Sobre algoritmos de ordenação, assinale a alternativa **incorreta**:

Resposta Selecionada: b.
BubbleSort: Percorre o vetor várias vezes colocando o maior elemento no início do vetor a cada iteração e continuando a partir do próximo elemento não ordenado.

Respostas:

a.
SelectionSort: Encontra o menor elemento do vetor e troca com o primeiro não ordenado, repetindo o processo sempre começando da próxima posição não ordenada.

b.
BubbleSort: Percorre o vetor várias vezes colocando o maior elemento no início do vetor a cada iteração e continuando a partir do próximo elemento não ordenado.

c.
InsertionSort: Cada elemento é inserido respectivamente na sua respectiva posição de ordem conforme vão sendo lidos do vetor original.

d.
QuickSort: Divide o vetor em dois através de um pivô com os maiores elementos de um lado e os menores do outro, continuando esta divisão de forma recursiva.

e.
HeapSort: Utiliza uma estrutura de árvore binária para ordenar os elementos, utilizando o princípio do Heap máximo da árvore como vetor.

Comentário da resposta: Resposta: B
Comentário: A ideia básica do BubbleSort é percorrer os dados sequencialmente várias vezes. Em cada passagem pelos dados, os elementos são comparados com o seu sucessor. Se os elementos não estiverem ordenados, devemos trocá-los de posição. O processo é repetido seguidamente do início até o final do vetor até que o vetor fique ordenado. Neste método a ordenação acontece colocando os maiores valores no final do vetor.
Em B - BubbleSort: Percorre o vetor várias vezes colocando o maior elemento no início do vetor a cada iteração e continuando a partir do próximo elemento não ordenado.

Pergunta 8

0,4 em 0,4 pontos

Considere o grafo criado abaixo:
Assinatura:
criaAresta(GRAFO* gr, int verticeInicial, int verticeFim, PESO p)

```
int main(void) {
    GRAFO* gr = criaGrafo(5);
    criaAresta(gr, 0, 1, 2);
    criaAresta(gr, 1, 0, 2);
    criaAresta(gr, 0, 2, 7);
    criaAresta(gr, 2, 0, 7);
    criaAresta(gr, 1, 3, 3);
    criaAresta(gr, 3, 1, 3);
    criaAresta(gr, 1, 4, 1);
    criaAresta(gr, 4, 1, 1);
    criaAresta(gr, 2, 3, 6);
    criaAresta(gr, 3, 2, 6);
    criaAresta(gr, 2, 4, 3);
    criaAresta(gr, 4, 2, 3);
    criaAresta(gr, 3, 4, 4);
    criaAresta(gr, 4, 3, 4);
    imprime(gr);
    int* r = dijkstra(gr, 0);
}
```

Assinale a resposta com o menor caminho entre V0 e V2.

Resposta Selecionada: c. V0 V1 V4 V2

Respostas:

a. V0 V2

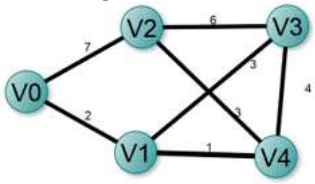
b. V0 V1 V2

c. V0 V1 V4 V2

d. V0 V1 V3 V2

e. V0 V1 V3 V4 V2

Comentário da resposta: Resposta: C
Comentário 3.5 Grafos: conceitos, representação e aplicações.
Montando o grafo:



O caminho mais curto é: V0 2 V1 2+1=3 V4 3+3=6 V2 distância de 6.

Pergunta 9

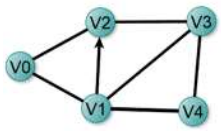
0,4 em 0,4 pontos

Considere a matriz de adjacência abaixo:

	V0	V1	V2	V3	V4
V0		1	1		
V1	1		1	1	1
V2	1			1	
V3		1	1		1
V4		1		1	

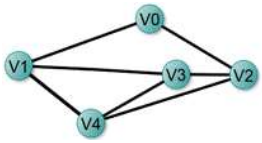
Assinale a alternativa com o grafo da Matriz de Adjacência acima:

Resposta Selecionada:

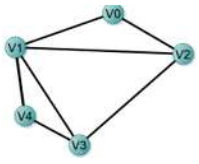


d.

Respostas:

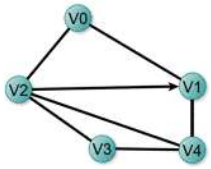


a.

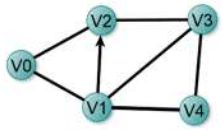


b.

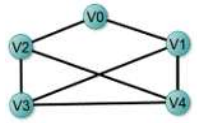
c.



d.



e.



Comentário da resposta: Resposta: D
Comentário:
Montando o grafo:

	V0	V1	V2	V3	V4
V0		1	1		
V1	1		1	1	1
V2	1			1	
V3		1	1		1
V4		1		1	

Pergunta 10

0,4 em 0,4 pontos

Sobre a Tabela Hash, afirma-se:

I. Toda função hash utiliza o quociente da divisão para prencher a tabela.
II. As colisões acontecem quando a função hash acha uma posisão vaga.
III. A função de hash, determina a posição na qual o elemento se encontra armazenado na tabela.

Assinale a alternativa correta em relação às afirmações acima:

Resposta Selecionada: c. Apenas a afirmação III é correta.

Respostas:

a. Apenas a afirmação I é correta.

b. Apenas a afirmação II é correta.

c. Apenas a afirmação III é correta.

d. Mais de uma das afirmações está correta.

e. Nenhuma das afirmações está correta.

Comentário da resposta:	Resposta: C Comentário: As seguintes propriedades são importantes para definir uma função de *hash*. • É necessário para termos acesso rápido e eficiente para avaliar a função de *hash*, que determina a posição na qual o elemento se encontra armazenado na tabela. • Espalhar bem as chaves de busca: isto é necessário para minimizarmos as ocorrências de colisões, dois valores eventualmente sendo direcionados para a mesma célula da tabela. Uma das formas clássicas e simples para explicar o uso de Hash é a operação resto (%) da chave em um vetor com a dimensão igual ao divisor.

← OK