1

Marcar para revisão

Na otimização de rotas de entrega, é vital determinar o caminho mais eficiente entre múltiplos pontos. Utilizando grafos, pode-se modelar esta questão para minimizar custos e tempo.

Qual algoritmo é preferencialmente utilizado para determinar o caminho mais curto em um grafo representando rotas de entrega?

A Algoritmo de Floyd-Warshall.

B Algoritmo de Prim.

C Busca em Largura.

D Coloração de grafos.

E Algoritmo de Dijkstra.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

O Algoritmo de Dijkstra é preferencialmente utilizado para determinar o caminho mais curto em um grafo que representa rotas de entrega, pois calcula o caminho de menor custo entre um vértice de origem e todos os outros vértices no grafo, sendo ideal para a otimização de rotas.

2

Marcar para revisão

(CS-UFG - Fundação Unirg - Analista de Sistemas - 2017)
Seja S o grafo de fluxo de controle de um programa P. Se o teste que aplica um conjunto de dados de teste satisfaz o critério todos os ramos de S, então pode-se concluir que esse conjunto também irá satisfazer o critério:

A Todos os caminhos de P.

B Todas as respostas de P.

C Todos os comandos de P.

D Todas as classes de P.

E Todos os predicados de P.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

O critério "todos os ramos" é um critério de cobertura de teste que exige que cada ramo (ou caminho de decisão) em um programa seja executado pelo menos uma vez. Se um conjunto de dados de teste satisfaz este critério, significa que todos os possíveis caminhos de decisão foram testados. Portanto, pode-se concluir que todos os comandos do programa também foram executados, pois cada comando está associado a um ou mais ramos no grafo de fluxo de controle. Assim, a alternativa correta é "Todos os comandos de P".

3

Marcar para revisão

(COMPERVE - UFRN - Engenheiro - Engenharia da Computação - 2019)

O código abaixo pode ser utilizado para atravessar um grafo:

```
Entrada: um gráfico G e um vértice v de G
Saída: todos os vértices alcançáveis de v marcados
função DFS(G,v):
   marque v
   para todas as arestas adjacentes a v, faça
      se vértice w não estiver marcado, então
            Chame recursivamente DFS(G,w)
       fim se
    fim para
fim função
```

Entre os diversos tipos de algoritmos utilizados para atravessar grafos, esse código implementa o algoritmo:

A Busca em profundidade ou depth first search.

B Busca em largura ou breadth first search.

C Busca melhor-primeiro ou best first search.

D Busca exaustiva ou brute force search.

E Busca pelo caminho mínimo (shortest path).

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

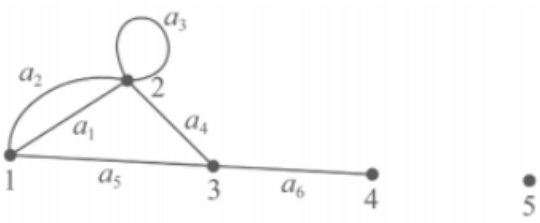
Gabarito Comentado

O código apresentado no enunciado é uma implementação do algoritmo de busca em profundidade, também conhecido como depth first search (DFS). Esse algoritmo é caracterizado por explorar completamente um vértice antes de passar para o próximo. Ele marca o vértice inicial, e então, para cada aresta adjacente a esse vértice, se o vértice adjacente não estiver marcado, o algoritmo é chamado recursivamente para esse vértice. Portanto, a alternativa correta é a "Busca em profundidade ou depth first search".

4

Marcar para revisão

(CESPE/CEBRASPE - IFF - Professor - Engenharia da Computação - 2018)



Considerando o grafo precedente, assinale a opção correta:

A Os nós 1 e 4 são adjacentes.

B O nó 5 é adjacente a si mesmo.

C Os arcos a1 e a2 são arcos irmãos.

D Os nós 2 e 3 têm grau 3.

E O grafo não pode ser classificado como conexo.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

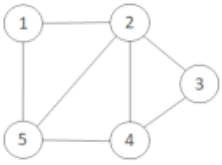
Gabarito Comentado

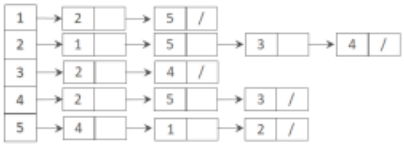
Resposta correta: O grafo não pode ser classificado como conexo.

5

Marcar para revisão

(FCM - IFN-MG - Ciências da Computação: Teoria da Computação - 2018)
Considere o grafo abaixo assim como sua representação por lista de adjacência:

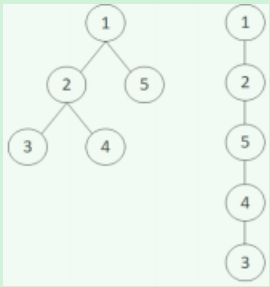




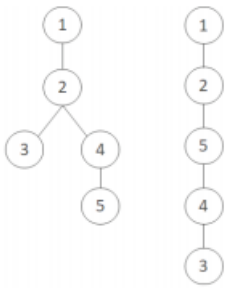
Referência: CORMEN, Thomas H. et al. **Algoritmos**: teoria e prática. Rio de Janeiro: Campus, 2012.

A Árvore em Largura e a Árvore em Profundidade, respectivamente, tendo como raiz o vértice 1, são:

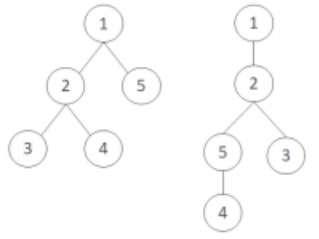
A



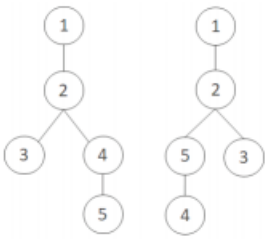
B



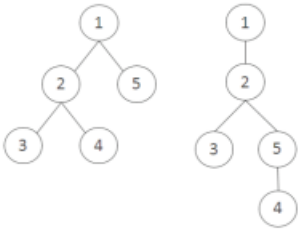
C



D



E

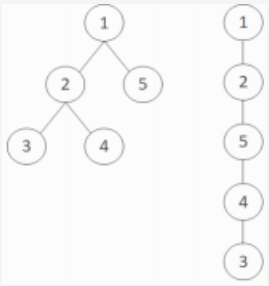


Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

Resposta correta:



1

Marcar para revisão

Árvores binárias podem ser usadas para representar expressões aritméticas. Como um exemplo de expressão, podemos ter: a * b + f sen - h * j com os elementos enumerados "Em-ordem". Nesse caso, a árvore binária terá como raiz:

A o átomo a.

B o átomo j.

C o átomo +.

D o átomo sen.

E o átomo *.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

Na representação de expressões aritméticas por meio de árvores binárias, a raiz da árvore é o operador de maior precedência, que, no caso da expressão dada, é o operador '+'. Portanto, o átomo '+' é a raiz da árvore binária que representa a expressão aritmética a * b + f sen - h * j.

2

Marcar para revisão

Um programador está implementando uma árvore binária de busca. Ele precisa garantir que as buscas, inserções e remoções sejam realizadas de forma eficiente. A estrutura da árvore deve evitar o pior caso, onde ela se degenera em uma lista linear.
Qual método de percurso de árvore é ideal para imprimir todos os elementos de uma árvore binária de busca em ordem crescente?

A Percurso em pré-ordem.

B Percurso em pós-ordem.

C Percurso em ordem simétrica.

D Percurso em nível.

E Percurso em espiral.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

O percurso em ordem simétrica (in-order traversal) é o mais adequado para imprimir os elementos de uma árvore binária de busca em ordem crescente, pois visita os nós da árvore de forma sequencial e ascendente.

3

Marcar para revisão

Em um sistema de gerenciamento de dados, uma árvore AVL foi implementada para otimizar as buscas. Devido ao grande volume de dados, a eficiência na inserção e remoção é crucial. A árvore AVL é escolhida por sua capacidade de auto-balanceamento após cada operação.
Qual é a principal característica de uma árvore AVL que a diferencia de uma árvore binária de busca comum?

A Altura máxima de log n.

B Não permite valores duplicados.

C Cada nó tem até dois filhos.

D Sempre armazena dados inteiros.

E Nós com um único filho são proibidos.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

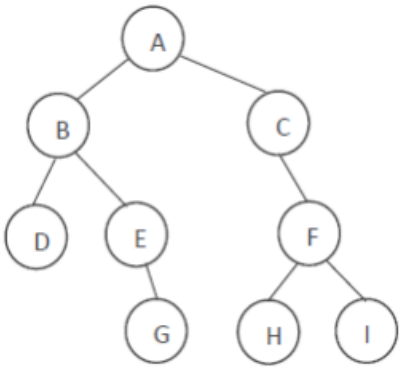
Gabarito Comentado

A característica distintiva de uma árvore AVL é sua altura máxima de log n, garantindo que as operações sejam eficientes. Essa propriedade é mantida através do auto-balanceamento após cada inserção ou remoção.

4

Marcar para revisão

Analise a seguinte árvore binária e assinale a alternativa correta.



A "A" é filho de todos.

B "B" e "C" são caules da árvore.

C "B" tem grau de saída 3 e "C" grau 2.

D TA é a subárvore enraizada em "A", portanto toda a árvore.

E Com exceção do nó "A", que é raiz, os demais nós são conhecido como folhas.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

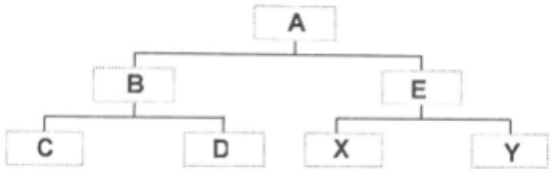
A resposta correta é: TA é a subárvore enraizada em "A", portanto toda a árvore.

5

Marcar para revisão

A estrutura abaixo representa uma célula de uma árvore em linguagem C;

```
typedef struct _no {
int chave;
struct _no *esq, *dir;
} no;
```



Assinale a alternativa correta sobre qual sequência será impressa ao executar um caminhamento na árvore abaixo, conforme o código escrito em linguagem C a seguir:

```
void ordem (no *arvore) {
if (arvore != NULL) {
printf ( "%d", arvore -> chave);
ordem ( arvore -> esq );
ordem ( arvore -> dir );
}
}
```

A CBDAXEY

B ABCDEXY

C ABDCEYX

D YXEABBC

E AEXYBCD

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

A resposta correta é: ABCDEXY

6

Marcar para revisão

Uma árvore binária de busca é utilizada em um software de inventário para organizar itens. A árvore permite buscas eficientes, mas há preocupações com o desempenho quando a árvore se torna desequilibrada.
Qual é a complexidade de tempo no pior caso para a busca em uma árvore binária de busca?

A O(log n).

B O(n log n).

C O(n).

D O(1).

E $O(n^2)$.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

No pior caso, onde a árvore binária de busca se degenera em uma lista linear, a complexidade de tempo para a busca é O(n), pois cada elemento precisa ser verificado sequencialmente.

7

Marcar para revisão

Para melhorar a eficiência de uma aplicação de banco de dados, um desenvolvedor opta por utilizar uma árvore AVL. Essa escolha é devido à necessidade de manter a estrutura da árvore balanceada, assegurando tempos de busca, inserção e remoção consistentemente rápidos.
Em uma árvore AVL, qual é o propósito da rotação de nós?

A Aumentar a altura da árvore.

B Reduzir o número de nós.

C Balancear a árvore.

D Converter em árvore binária de busca.

E Eliminar nós duplicados.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

A rotação de nós em uma árvore AVL tem como objetivo principal balancear a árvore. Isso assegura que a altura da árvore se mantenha dentro do limite de log n, mantendo a eficiência das operações.

8

Marcar para revisão

Acerca das estruturas de dados Árvores, analise as afirmativas a seguir.
I. A árvore AVL é uma árvore binária com uma condição de balanço, porém não completamente balanceada.
II. Árvores admitem tratamento computacional eficiente quando comparadas às estruturas mais genéricas como os grafos.
III. Em uma Árvore Binária de Busca, todas as chaves da subárvore esquerda são maiores que a chave da raiz.
Assinale:

A Se somente a afirmativa I estiver correta.

B Se somente as afirmativas I e II estiverem corretas.

C Se somente as afirmativas I e III estiverem corretas.

D Se somente as afirmativas II e III estiverem corretas.

E Se todas as afirmativas estiverem corretas.

Resposta correta
Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado
As afirmativas I e II estão corretas. A árvore AVL é uma árvore binária que mantém suas operações eficientes através de uma condição de balanço, embora não seja completamente balanceada. Além disso, as árvores permitem um tratamento computacional eficiente quando comparadas a estruturas mais genéricas, como os grafos. No entanto, a afirmativa III está incorreta. Em uma Árvore Binária de Busca, todas as chaves da subárvore esquerda são menores, e não maiores, que a chave da raiz.

9

Marcar para revisão

Ao construir um sistema de indexação para um banco de dados, um engenheiro de software escolhe usar uma árvore AVL devido à sua eficiência. A árvore precisa ser mantida balanceada após cada inserção e remoção para garantir um desempenho otimizado. O balanceamento é essencial para manter a altura da árvore dentro do limite desejado.
Qual é a consequência de uma árvore binária de busca estar desbalanceada?

A Aumento da eficiência de busca.

B Redução do espaço necessário para armazenamento.

C Complexidade de tempo de busca reduzida para O(log n).

D Aumento da complexidade de tempo de busca.

E Conversão automática para árvore AVL.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

Quando uma árvore binária de busca está desbalanceada, isso resulta em um aumento na complexidade de tempo de busca. No pior caso, a árvore pode se degenerar em uma lista linear, levando a uma complexidade de busca O(n), que é menos eficiente do que a desejada O(log n).

8

Marcar para revisão

Considere que os percentuais foram inseridos no vetor vet de 5 posições, a partir da posição 1, na seguinte sequência: 25.33, 27.72, 27.10, 26.90 e 27.31, ou seja, com os dados de 2008 até 2012. Um técnico em processamento de dados do TCE-RS utilizou um método para ordenar os dados de vet. O método realizou os seguintes passos no processo de ordenação:

- Passo 1 - 25.33 27.72 27.10 26.90 27.31;
- Passo 2 - 25.33 27.10 27.72 26.90 27.31;
- Passo 3 - 25.33 26.90 27.10 27.72 27.31;
- Passo 4 - 25.33 26.90 27.10 27.31 27.72.

Trata-se do método de ordenação:

A *Bubble sort*

B *Selection sort*

C *Quick sort*

D *Fast sort*

E *Insertion sort*

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

O método de ordenação utilizado pelo técnico em processamento de dados do TCE-RS é o *Insertion sort*. Esse método de ordenação é caracterizado por dividir o vetor em duas partes: uma ordenada e outra não ordenada. A cada passo, o algoritmo pega o primeiro elemento da parte não ordenada e insere na posição correta na parte ordenada. Isso é feito repetidamente até que todos os elementos estejam na parte ordenada. No exemplo dado, podemos observar que a cada passo, o elemento é inserido na posição correta na parte já ordenada, caracterizando o método *Insertion sort*.

5

Marcar para revisão

Em um projeto de software, foi identificada a necessidade de um algoritmo de ordenação eficiente para grandes volumes de dados. A equipe decidiu utilizar um algoritmo avançado de ordenação para melhorar a performance.
Qual algoritmo de ordenação avançado seria mais adequado para este cenário?

A Bubble Sort.

B Insertion Sort.

C Quick Sort.

D Selection Sort.

E Linear Search.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

O Quick Sort é ideal para grandes volumes de dados devido à sua alta eficiência e velocidade. Diferentemente de algoritmos elementares como Bubble Sort ou Selection Sort, o Quick Sort divide os dados em subconjuntos menores para ordená-los rapidamente, sendo mais adequado para o cenário descrito.

1

Marcar para revisão

Assinale a alternativa correta a respeito dos algoritmos de ordenação *bubble sort* e *quick sort*:

A O *quick sort* tem um tempo de execução logarítmico no pior caso.

B O *bubble sort* tem um tempo de execução logarítmico em média.

C O *bubble sort* e o *quick sort* têm um tempo de execução quadrático no pior caso.

D O *quick sort* efetua a ordenação da lista, realizando trocas de ordem sucessivas de elementos subsequentes.

E O *bubble sort* é um algoritmo recursivo que efetua, a cada passo, o particionamento da lista que será ordenada em duas sublistas - uma com os elementos maiores que um elemento escolhido como pivô, e outra com os elementos maiores que este.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

Os algoritmos de ordenação *bubble sort* e *quick sort* possuem um tempo de execução quadrático no pior caso. Isso significa que, no pior cenário possível, o tempo de execução desses algoritmos aumenta proporcionalmente ao quadrado do tamanho da entrada. No caso do *bubble sort*, isso ocorre porque ele compara cada par de elementos adjacentes e os troca se estiverem na ordem errada, repetindo esse processo até que a lista esteja ordenada. Já no caso do *quick sort*, o tempo de execução quadrático ocorre no pior caso quando o pivô escolhido é o menor ou o maior elemento da lista, fazendo com que uma das partições seja vazia.

2

Marcar para revisão

Em relação aos algoritmos de ordenação, avalie se as afirmativas a seguir são verdadeiras (V) ou falsas (F):

I. O algoritmo *quick sort* é muito eficiente quando há uma quantidade pequena de elementos a ordenar.
II. O algoritmo *shell sort* utiliza intensamente a inserção direta.
III. No algoritmo *bubble sort*, o número de variáveis envolvidas é pequeno.

As afirmativas I, II e III são, respectivamente:

A V, F e V

B F, V e V

C V, F e F

D F, F e V

E V, V e V

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

A alternativa correta é a letra B, que indica que as afirmativas I, II e III são, respectivamente, Falsa, Verdadeira e Verdadeira. A primeira afirmativa é falsa porque o algoritmo Quick Sort é mais eficiente quando há uma grande quantidade de elementos a ordenar, não uma pequena. A segunda afirmativa é verdadeira, pois o algoritmo Shell Sort realmente utiliza intensamente a inserção direta. A terceira afirmativa também é verdadeira, pois no algoritmo Bubble Sort, o número de variáveis envolvidas é realmente pequeno.

3

Marcar para revisão

"Quick Sort" é um dos algoritmos de ordenação mais rápidos e eficientes disponíveis, e é frequentemente a escolha preferida em aplicações práticas quando a estabilidade não é uma preocupação principal.
Qual é o método utilizado no particionamento do "Quick Sort"?

A Utilizar o elemento médio do vetor.

B Utilizar o primeiro ou o último elemento do vetor.

C Utilizar um elemento aleatório do vetor.

D Utilizar o elemento mais frequente do vetor.

E Utilizar o elemento central do vetor.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

No particionamento, o pivô pode ser considerado como o último elemento que se deseja encontrar, e utilizar o último ou o primeiro elemento é um mecanismo para facilitar a implementação.

4

Marcar para revisão

O "Selection Sort", ou ordenação por seleção, é outro algoritmo simples de ordenação. O princípio básico deste método é dividir o array em duas partes: a parte já ordenada e a parte não ordenada.

Em relação à eficiência para grandes conjuntos de dados, como o ¿Selection Sort¿ se comporta?

A Muito eficiente.

B Medianamente eficiente.

C Ineficiente.

D Depende do tipo de dados.

E Altamente otimizado para grandes conjuntos.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

Como ele possui uma complexidade $O(n^2)$, ele é ineficiente para grandes conjuntos de dados.

5

Marcar para revisão

Uma empresa de tecnologia está desenvolvendo um sistema de arquivos que exige ordenação estável para manter a ordem de registros iguais.
Qual algoritmo de ordenação oferece a característica de ser estável?

A Quick Sort.

B Heap Sort.

C Shell Sort.

D Merge Sort.

E Radix Sort.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

O Merge Sort é um algoritmo de ordenação estável, o que significa que mantém a ordem relativa de registros iguais. Essa característica é crucial em sistemas onde a ordem dos registros deve ser preservada.

6

Marcar para revisão

A ordenação de elementos em um vetor pode ser executada a partir de diversos algoritmos conhecidos que são adequados para situações específicas. Sobre algoritmos de ordenação, analise as seguintes afirmativas:
I. O algoritmo *bubble sort* é eficiente para ordenar poucos elementos, mas é lento para ordenar muitos itens.
II. O algoritmo *selection sort* para ordenação crescente consiste em mover o menor valor do vetor para a primeira posição; depois, o segundo menor para a segunda posição; e assim sucessivamente, até os dois últimos valores.
III. O algoritmo *quick sort* ordena os valores de um vetor por meio de sucessivas seleções do elemento correto a ser posicionado em um segmento ordenado.
Estão corretas as afirmativas:

A I apenas

B II apenas

C I e II

D I e III

E I, II e III

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

As afirmativas I e II estão corretas. A afirmativa I está correta porque o algoritmo bubble sort é eficiente para ordenar poucos elementos, mas torna-se lento quando o número de itens a serem ordenados aumenta. A afirmativa II também está correta, pois descreve corretamente o funcionamento do algoritmo selection sort, que seleciona o menor valor do vetor e o move para a primeira posição, repetindo o processo até que todos os valores estejam ordenados. No entanto, a afirmativa III está incorreta, pois descreve erroneamente o algoritmo quick sort. Este algoritmo funciona escolhendo um "pivô" e particionando os outros elementos em dois sub-arrays, de acordo com se são menores ou maiores que o pivô. O processo é então repetido para os sub-arrays.

7

Marcar para revisão

O "Bubble Sort", ou "Ordenação por Bolha" em tradução livre, é um dos algoritmos de ordenação mais simples.

O que acontece com o maior valor na lista durante o processo de ordenação crescente do "Bubble Sort"?

A É movido para a esquerda.

B É deixado onde está.

C É movido para o centro.

D É empurrado para baixo.

E É continuamente empurrado até o fim da passagem.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

O maior valor na lista está em alguma comparação. Como a ordenação é do tipo crescente, esse valor será continuamente empurrado até o fim da passagem.

8

Marcar para revisão

O algoritmo de ordenação Shell Sort, ou simplesmente Shell Sort, é uma generalização do algoritmo de inserção que permite a troca de itens distantes.
Qual é o propósito das sentinelas em algumas variações da ordenação "Shell Sort"?

A Aumentar a eficiência do algoritmo.

B Guardar valores especiais de terminação.

C Organizar a sequência de incrementos.

D Facilitar a visualização do código.

E Indicar o começo e o fim do array.

Resposta correta
Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado
As sentinelas guardam valores especiais de terminação.

9

Marcar para revisão

A complexidade de algoritmos é uma medida que indica os recursos necessários para a execução de um algoritmo em função do tamanho da entrada.
Como expressamos a complexidade do "Bubble Sort" quando o tempo computacional varia de forma quadrática com o tamanho do problema?

A O(log n)

B O(n log n)

C O(n)

D $O(n^2)$

E $O(n^3)$

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

Como o tempo computacional varia de forma quadrática com o tamanho do problema, trata-se de um algoritmo de ordem quadrática, e expressamos isso escrevendo $O(n^2)$.

10

Marcar para revisão

Uma aplicação de gerenciamento de inventário precisa ordenar itens com base em seus códigos alfanuméricos. O algoritmo escolhido deve ser eficiente em lidar com uma variedade de padrões de dados.
Qual algoritmo de ordenação atende melhor a esta necessidade?

A Binary Search.

B Quick Sort.

C Heap Sort.

D Radix Sort.

E Shell Sort.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

O Radix Sort é especialmente eficaz para ordenar dados alfanuméricos, pois trata os caracteres de acordo com sua posição individual. Isso o torna mais adequado para a ordenação de códigos alfanuméricos em comparação com outros algoritmos que podem não ser tão eficientes para esse tipo de dado.

8

Marcar para revisão

O "Selection Sort" (Ordenação por Seleção) é um dos algoritmos de ordenação mais simples e didáticos.
Como o algoritmo "Selection Sort" divide o array durante sua execução?

A Em valores crescentes e decrescentes.

B Em partes iguais.

C Em uma parte ordenada e uma parte que ainda não foi ordenada.

D Em valores pares e ímpares.

E Não divide o array.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

O algoritmo divide o array em duas partes: A parte ordenada, à esquerda do elemento analisado. A parte que ainda não foi ordenada, à direita do elemento.

2

Marcar para revisão

Para um algoritmo de busca em uma lista ordenada, a velocidade é crucial. A equipe de desenvolvimento precisa escolher um método de ordenação compatível com uma busca eficiente.
Qual método de ordenação facilita uma busca rápida em uma lista?

A Linear Search.

B Binary Search.

C Quick Sort.

D Bubble Sort.

E Heap Sort.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

O Quick Sort é ideal para preparar listas para buscas rápidas, como a busca binária, devido à sua eficiência na ordenação. Algoritmos como Bubble Sort podem ser menos eficientes, tornando as buscas subsequentes mais lentas.

6

Marcar para revisão

Uma empresa está otimizando sua base de dados de clientes. O desafio é escolher um algoritmo de ordenação que seja eficiente em termos de memória.
Qual algoritmo seria o mais indicado para este caso?

A Merge Sort.

B Quick Sort.

C Shell Sort.

D Heap Sort.

E Bubble Sort.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

O Heap Sort é conhecido por sua eficiência em termos de uso de memória, uma vez que realiza a ordenação no local (in-place). Isso o diferencia de outros algoritmos como o Merge Sort, que requer mais memória devido à sua abordagem de dividir e conquistar.

1

Marcar para revisão

O código abaixo é uma implementação:

```
public class Misterio {
        public static long Misterio(long x) {
                if (x == 1)
                        return 1;
                else
                        return x * Misterio(x-1);
        }
}
```

A Recursiva da exponenciação

B Iterativa da série de Fibonacci

C Recursiva da série de Fibonacci

D Iterativa da exponenciação

E Recursiva do fatorial

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

O código apresentado é uma implementação recursiva do cálculo do fatorial de um número. A função "Misterio" recebe um número 'x' como parâmetro e verifica se 'x' é igual a 1. Se for, a função retorna 1. Caso contrário, a função retorna 'x' multiplicado pelo resultado da função "Misterio" aplicada a 'x-1'. Isso é uma implementação clássica do cálculo do fatorial, onde o fatorial de um número 'n' é o produto de 'n' por 'n-1' por 'n-2'... até 1. Portanto, a alternativa correta é a 'E': Recursiva do fatorial.

2

Marcar para revisão

Um algoritmo recursivo é projetado para encontrar o elemento máximo em uma lista de números. A função divide a lista ao meio a cada chamada, comparando os elementos até encontrar o maior.
Qual abordagem recursiva é mais adequada para este algoritmo?

A Recursividade linear.

B Recursividade múltipla.

C Recursividade aninhada.

D Recursão de cauda.

E Recursividade indireta.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

A recursividade linear, onde a função faz uma única chamada recursiva a cada passo, é a abordagem mais adequada para encontrar o elemento máximo em uma lista. Este método permite dividir e conquistar o problema de forma eficiente, processando a lista progressivamente.

3

Marcar para revisão

Ano: 2014 Banca: FUNCAB Órgão: MDA Prova: FUNCAB - 2014 - MDA - Analista de Negócios
Observe o algoritmo a seguir, que utiliza o conceito de função recursiva.

```
algoritmo "MDA"
var
X, W, N : inteiro
funcao FF(Y:inteiro):inteiro
inicio
   N <- N + 1|
   se Y < 2 então
      retorne 1
   senão
     retorne Y * FF(Y-1)
    fimse
fimfuncao

inicio
  X <-5
  N <-0
  W <- FF(X)
  W <-W-50
  escreval(W,N)
fimalgoritmo
```

Após a execução, o algoritmo, os valores de W e N serão, respectivamente:

A 120 e 5

B 120 e 1

C 70 e 0

D 70 e 5

E 70 e 1

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

O algoritmo apresentado utiliza a função recursiva FF(Y). Inicialmente, X é definido como 5 e N como 0. A função FF(Y) é chamada com X (5) como argumento. A função FF(Y) é uma função recursiva que calcula o fatorial de Y. Portanto, FF(5) retorna 120. Em cada chamada da função FF(Y), N é incrementado em 1. Como a função é chamada 5 vezes (para calcular o fatorial de 5), N se torna 5. Após a execução da função FF(Y), W é definido como FF(X) - 50, ou seja, 120 - 50, resultando em 70. Portanto, após a execução do algoritmo, os valores de W e N são, respectivamente, 70 e 5.

4

Marcar para revisão

Sobre o conceito de Algoritmos Recursivos, analise as afirmações abaixo e, a seguir, assinale a alternativa correta.
I. Um programa tem um número limitado de procedimentos recursivos.
II. Recursividade é utilizada exclusivamente quando não se sabe solucionar um problema de maneira imediata, então é realizada a divisão em problemas menores para alcançar o resultado desejado.
III. Todos os problemas computacionais resolvidos de maneira iterativa gastam mais memória que se resolvidos de forma recursiva.

A Somente a afirmação I está correta

B Somente a afirmação II está correta

C Somente a afirmação III está correta

D As afirmações I e II estão corretas

E Nenhuma das afirmações está correta

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

As afirmações apresentadas na questão não estão corretas. A afirmação I é incorreta porque um programa pode ter um número ilimitado de procedimentos recursivos, desde que não cause um estouro de pilha. A afirmação II também é incorreta, pois a recursividade não é utilizada exclusivamente quando não se sabe como resolver um problema de maneira imediata. Existem situações em que a recursividade pode ser uma escolha de design, mesmo quando uma solução iterativa é conhecida. Por fim, a afirmação III é incorreta porque não é verdade que todos os problemas computacionais resolvidos de maneira iterativa gastam mais memória do que se resolvidos de forma recursiva. Na verdade, em muitos casos, a recursividade pode consumir mais memória devido ao uso da pilha para armazenar os estados de chamada recursiva.

5

Marcar para revisão

Considerando um algoritmo recursivo que calcula fatorial de um número, onde a função fatorial chama a si mesma com o valor decrementado, até que o caso base (fatorial de 0 ou 1) seja alcançado.
Qual é o caso base mais apropriado para essa função fatorial?

A Fatorial de 2

B Fatorial de 1

C Fatorial de 3

D Fatorial de 10

E Fatorial de 0

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

O caso base para uma função fatorial recursiva eficiente é geralmente o fatorial de 1, pois fatorial de 1 é igual a 1. Este caso base impede que a função continue a se chamar infinitamente, proporcionando um ponto de parada claro para a recursão.

6

Marcar para revisão

Um programa usa recursividade indireta para calcular resultados. Duas funções, A e B, chamam uma à outra alternadamente. A função A inicia o processo e passa o controle para B, que por sua vez chama A novamente com parâmetros atualizados.
Qual é uma possível desvantagem dessa abordagem de recursividade indireta?

A Maior clareza no código.

B Redução no uso de memória.

C Aumento da complexidade do código.

D Melhora na eficiência computacional.

E Facilidade na depuração.

Resposta correta
Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado
A recursividade indireta, onde duas ou mais funções se chamam mutuamente, pode aumentar significativamente a complexidade do código. Isso torna o programa mais difícil de entender e de depurar, especialmente em situações onde as chamadas são frequentes e o fluxo de controle é complexo.

7

Marcar para revisão

Ano: 2020 Banca: FAPEC Órgão: UFMS Prova: FAPEC - 2020 - UFMS - Técnico de Tecnologia da Informação
Considere a seguinte função recursiva: função recursiva(x : inteiro): inteiro início

```
se x = 1 então
   retorne -x
senão
    retorne -5 * recursiva(x - 1) + x
```

fimse

fimfuncao

Qual é o valor retornado pela função se ela for chamada com x = 4?

A -143

B -56

C 143

D 56

E 164

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

A função recursiva apresentada no enunciado retorna um valor baseado em uma operação que envolve o próprio resultado da função (recursividade) e o valor de entrada decrementado. Quando x = 1, a função retorna -1. Para x = 4, a função será chamada recursivamente até que x seja igual a 1, realizando a operação -5 * recursiva(x - 1) + x em cada chamada. Seguindo essa lógica, a função retornará o valor 164 quando x = 4, o que corresponde à alternativa E.

8

Marcar para revisão

Em um sistema de gerenciamento de biblioteca, uma função recursiva é utilizada para calcular o número total de livros em uma pilha, considerando que cada livro pode conter referências a outros livros. A função soma as referências recursivamente.
Qual o principal desafio ao implementar essa função recursiva em sistemas de gerenciamento?

A Risco de duplicação de dados.

B Dificuldade em acessar dados externos.

C Complexidade na interface gráfica.

D Consumo excessivo de memória.

E Limitação na quantidade de livros.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

Em funções recursivas, especialmente aquelas que envolvem muitas chamadas, como no caso da contagem de livros em uma biblioteca, o consumo excessivo de memória é um desafio comum. Cada chamada recursiva consome memória adicional, o que pode levar a problemas de desempenho ou estouros de pilha.

9

Marcar para revisão

Um algoritmo recursivo de busca em árvore binária verifica se um elemento está presente na árvore. A cada passo, a função compara o elemento com o nó atual e decide continuar a busca na subárvore esquerda ou direita, até encontrar o elemento ou atingir um nó folha.
Qual é o principal benefício dessa abordagem recursiva na busca em árvore binária?

A Velocidade constante em todas as buscas.

B Menor uso de recursos de rede.

C Simplificação do código de busca.

D Independência do tamanho da árvore.

E Eliminação de erros de comparação.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

O principal benefício de usar uma abordagem recursiva na busca em árvore binária é a simplificação do código. A recursividade permite dividir o problema em casos menores mais gerenciáveis, o que torna o código mais claro e fácil de entender, especialmente em estruturas de dados complexas como árvores binárias.

10

Marcar para revisão

Ano: 2019 Banca: UFSC Órgão: UFSC Prova: UFSC - 2019 - UFSC - Técnico de Tecnologia da Informação

A respeito de um algoritmo recursivo, analise as afirmativas abaixo e assinale a alternativa correta.

I. Deve conter pelo menos uma estrutura de repetição.

II. Deve conter pelo menos uma estrutura de seleção.

III. Deve invocar a si mesmo pelo menos uma vez ao ser executado.

A Todas as afirmativas estão corretas.

B Somente a afirmativa II está correta.

C Somente as afirmativas I e II estão corretas.

D Somente a afirmativa I está correta.

E Somente as afirmativas II e III estão corretas.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

A alternativa correta é a B, que afirma que "Somente a afirmativa II está correta". Isso porque um algoritmo recursivo não necessariamente precisa conter uma estrutura de repetição (afirmativa I), mas sim uma estrutura de seleção (afirmativa II), que é usada para determinar as condições de parada da recursão. A afirmativa III também está incorreta, pois um algoritmo recursivo deve invocar a si mesmo em todas as suas execuções, e não apenas uma vez.

1

Marcar para revisão

Leia as afirmativas a seguir considerando que f(n) e g(n) são funções positivas.
I- Se g(n) é O(f(n)), um algoritmo de função de complexidade de tempo f(n) possui Ordem de complexidade g(n).
II- Se g(n) é O(f(n)), f(n) é um limite superior para g(n).
III- Se a função g(n) = 7.log(n) +6 , então a função g(n) é O(log(n)).
IV- Se $g(n)=n^2$ e $f(n)=(n+1)^2$ temos que g(n) é O(f(n)) e f(n) é O(g(n)).
V- Se g(n) = 2n+1 e f(n) = 2n temos que g(n) = O(f(n)).
Assinale a alternativa que apresenta somente as afirmativas:

A I, II, IV, V.

B II, III, IV.

C II, III, IV, V.

D I, III, IV, V.

E II, III, V.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

A afirmativa I é falsa. Para ilustrar, considere um algoritmo A cuja complexidade é $O(n^3)$, ou seja, a função $f(x)=x^3$. A função g(x)=x é O(f(x)), porém a complexidade do algoritmo não é linear por hipótese, logo a proposição é falsa.

A afirmativa II é verdadeira, pois decorre diretamente da definição: g(n) é O(f(n)) se existe um certo n_0 tal que para todo $n > n_0$ $kf(n) > g(n)$ e isto configura uma cota assintótica superior.

A afirmativa III é verdadeira, pois $7\log(n)+6 < 8\log(n)$, ou seja, fazendo k=8 a função log(x) já é cota assintótica superior para 7log(n)+6.

A afirmativa IV é verdadeira, pois $n^2 < n^2 + 2n + 1$ isto é k=1, já configura que f é cota assintótica superior para g e $n^2 + 2n + 1 < 2n^2$, isto é k=2, já configura que g é cota assintótica superior para f.

A afirmativa V é verdadeira, pois f(n)=2n é cota assintótica superior para g(n)=2n+1 se k=2.

Portanto, as afirmativas II, III, IV e V são verdadeiras, o que corresponde à alternativa C.

2

Marcar para revisão

Um algoritmo para um aplicativo de previsão do tempo precisa processar rapidamente grandes volumes de dados meteorológicos. A eficiência na manipulação desses dados é crucial para a precisão e rapidez das previsões.
Qual é a notação adequada para descrever a eficiência de um algoritmo que processa dados meteorológicos?

A Notação ASCII.

B Notação Big O.

C Notação de Peano.

D Notação UML.

E Notação Sigma.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

A notação Big O é utilizada para descrever a eficiência de um algoritmo, especialmente em termos de tempo de execução e espaço necessário. No contexto de um aplicativo de previsão do tempo que processa grandes volumes de dados, a notação Big O é a mais adequada para avaliar a eficiência do algoritmo.

3

Marcar para revisão

Ao desenvolver um jogo de computador, um programador utiliza ponteiros para otimizar o gerenciamento de memória, especialmente para objetos que são frequentemente criados e destruídos durante o jogo.

Qual é a principal vantagem do uso de ponteiros no gerenciamento de memória em um jogo de computador?

A Aumento da velocidade de execução.

B Melhoria na interface gráfica.

C Redução do uso de memória.

D Facilitação da programação orientada a objetos.

E Melhor distribuição de tarefas entre processadores.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

A principal vantagem do uso de ponteiros no gerenciamento de memória em um jogo de computador é a redução do uso de memória. Ponteiros permitem um controle mais eficiente e dinâmico da memória, especialmente útil em aplicações como jogos, onde objetos são frequentemente criados e destruídos.

4

Marcar para revisão

Considere o algoritmo em pseudocódigo, descrito a seguir.

```
Para i=0 até n
Inicio
      j = 1
      enquanto j<n
       inicio
            j = 2 x j
            para k = 0 até j
            inicio
                  execute f
            fim
      fim
fim
```

Calcule a complexidade do algoritmo, sabendo que a função f tem complexidade igual a $O(n^2)$.

A $O(n^2\log^2(n))$

B $O(n^3)$

C $O(n^3\log(n))$

D $O(n^4\log(n))$

E $O(n^5)$

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

Vamos analisar o código simplificado abaixo:

J=1

Enquanto j < n

J = 2xj

Para k = 0 ate j

Operação elementar

Para facilitar, vamos fazer $n = 2^k$

J = 1 à j = 2, com 3 (2^1+1) iterações

J = 2 à j = 4 com 5 $(2^2+1$)iterações

J= 4 à j = 8 com 9 (2^3+1) iterações

J = 8 à j = 16 com 17 (2^4+1) iterações

J = 2^k à j = $(2^{k+1} + 1)$ iterações

O total de iterações é a soma $\sum_{i=1}^{\log n} (2^i + 1) < 2^k \sum_{j=1}^{\log n} 1 = 2^k \log n$, porém 2^k=n, assim a complexidade do código é n log n. Considerando OP com complexidade constante. Como OP é quadrática, temos que o código analisado é $n^3 \log n$.

O for mais externo se repete n vezes, assim a complexidade total do algoritmo é $n^4 \log n$

5

Marcar para revisão

Registros são exemplos de tipos de dados heterogêneos. Assim, sobre tipos de dados elementares e estruturados, é correto afirmar que os elementos de um registro são de tamanhos potencialmente diferentes e residem em posições de memória:

A Flexíveis

B Aleatórias

C Adjacentes

D Procedimentais

E Espalhadas

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

Os registros são tipos de dados heterogêneos, ou seja, podem conter diferentes tipos de dados em sua estrutura. Quando falamos sobre a alocação de memória desses elementos, eles são alocados em posições adjacentes de memória. Isso significa que eles são armazenados em locais de memória que estão próximos uns dos outros, não em posições aleatórias ou espalhadas. Portanto, a alternativa correta é a "C", que afirma que os elementos de um registro residem em posições de memória adjacentes.

6

Marcar para revisão

Analise as seguintes afirmações relacionadas a conceitos básicos sobre Programação:

I. Um procedimento é um conjunto de comandos para uma tarefa específica referenciada por um nome no algoritmo principal, retornando um determinado valor no seu próprio nome.
II. Podem-se inserir módulos em um algoritmo. Para isso, pode-se utilizar "Procedimentos" ou "Funções". As ações das "Funções" e dos "Procedimentos" são hierarquicamente subordinadas a um módulo principal.
III. Cada "Função" ou "Procedimento" pode utilizar constantes ou variáveis do módulo principal ou definir suas próprias constantes ou variáveis.
IV. Uma variável global indica o endereço onde um valor é armazenado na memória do computador, enquanto um ponteiro representa um valor numérico real.

Indique a opção que contenha todas as afirmações **verdadeiras.**

A I e II.

B II e III.

C III e IV.

D I e III.

E II e IV.

Resposta correta
Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado
Os procedimentos não retornam valores. Variáveis globais não indicam endereços. Ponteiro não representa um valor numérico real, eles representam endereços.

7

Marcar para revisão

Para otimizar o desempenho de um software de análise financeira, um programador decide reestruturar um algoritmo crucial, dividindo-o em módulos menores e mais gerenciáveis.
Qual técnica o programador está aplicando ao dividir o algoritmo em módulos menores?

A Refatoração.

B Encapsulamento.

C Herança.

D Polimorfismo.

E Modularização.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

Ao dividir um algoritmo complexo em módulos menores e mais gerenciáveis, o programador está aplicando a técnica de modularização. Esta abordagem facilita a manutenção, a compreensão e a otimização do código.

8

Marcar para revisão

Considere os algoritmos a seguir e as suas correspondentes complexidades indicadas:

Algoritmo	Complexidade
I - Busca Sequencial de um elemento em um vetor	O(N)
II - Busca, via pesquisa binária, de um elemento em um vetor ordenado de tamanho N	$O(\log_2 N)$
III – Somar todos os números de um vetor	O (N)
IV – Merge de duas listas	$O(n^2)$
V - Inclusão de um elemento em um vetor ordenado de tamanho N, mantendo-se a ordenação	O(1)

Estão corretas apenas as complexidades indicadas para os algoritmos:

A I, II e III.

B I, II e IV.

C II, III e V.

D II, III, IV e V.

E I, III, IV e V.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

A resposta correta é: I, II e III.

I - Correta, o pior caso da busca é não se encontrar o elemento buscado, só podemos concluir que o elemento não está na lista após compara o elemento buscado com todos os elementos, configurando O(N).

II - Correta, o pior caso também é não encontrar o elemento buscado. Fazemos divisões sucessivas até a lista tornar-se unitária e cada dada divisão dividimos o vetor de tamanho original por 2^k, paramos quando $1 = 2^k$, assim $k = \log_2 n$.

III - Correta, uma soma para cada elemento do vetor, como o vetor tem n elementos O(N).

IV - Falsa, no caso geral, O(N), mesmo se as listas forem ordenadas ainda é válido o resultado. Basta inserir no fim da lista o menor elemento das listas ainda não concatenadas.

V - Na pior hipótese, vamos inserir o novo elemento na primeira posição. Assim, teremos que mover todos os elementos 1 posição a frente, isto é, N-1 operações de atribuição O(N).

9

Marcar para revisão

Em um sistema de gerenciamento de biblioteca, um algoritmo foi desenvolvido para organizar livros. Utilizando uma estrutura de dados homogênea, o algoritmo categoriza os livros por gênero e autor, melhorando a eficiência da busca.
Qual estrutura de dados homogênea é mais adequada para este algoritmo?

A Lista encadeada.

B Pilha.

C Árvore binária.

D Array.

E Grafo.

Resposta correta

Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado

A estrutura de dados mais adequada para a organização de livros por gênero e autor em um sistema de gerenciamento de biblioteca é o Array. Ele permite armazenar elementos de um mesmo tipo (homogêneo) de forma sequencial, facilitando a busca e a organização.

10

Marcar para revisão

Um desenvolvedor está implementando um sistema de gerenciamento de estoque. Ele opta por uma estrutura de dados que permite armazenar e acessar informações de forma não sequencial.
Qual estrutura de dados é ideal para armazenar informações de forma não sequencial?

A Array.

B Lista duplamente encadeada.

C Pilha.

D Árvore.

E Fila.

Resposta correta
Parabéns, você selecionou a alternativa correta. Confira o gabarito comentado!

Gabarito Comentado
A estrutura de dados Árvore é ideal para armazenar e acessar informações de forma não sequencial, especialmente em um sistema de gerenciamento de estoque. Ela permite organizar dados de forma hierárquica, facilitando a busca e a atualização de informações.