


**1^a**

Questão

Acerto: **0,2** / **0,2**

Em Python é possível implementar um array utilizando o tipo padrão list. Essa implementação permite o uso das seguintes funções para inserir e remover um elemento, respectivamente:

- ☐ impose, remove/destroy.
- ☐ insert, delete/pop.
- ☒  append, remove/pop.
- ☐ append, pop/delete.
- ☐ insert, remove/destroy.

Respondido em 08/11/2023 23:26:11

Explicação:

Em Python a função append insere um elemento ao final da lista. As funções "remove" e "pop" podem remover um elemento, de maneiras diferentes. Remove tira um elemento conhecido usando o seu conteúdo, já pop remove um elemento usando seu índice, ou seja, a sua posição na lista.

**2^a**

Questão

Acerto: **0,2** / **0,2**

Uma lista L encadeada e ordenada está armazenada em memória seguindo o exemplo abaixo. Após a remoção do nó de chave 3, quais alterações terão ocorrido?

Endereço	Chave	Próximo
128	5	64
64	8	32
32	11	null
L-----> 24	3	128

- ☐ O endereço 24 conterá a chave 5 e próximo 64.
- ☒ A variável L apontará para 128.
- ☐ O endereço 32 terá seu campo próximo apontando para 24.
- ☐ L terá sido apagada.
- ☐ O conteúdo armazenado no endereço 32 será apagado.

Respondido em 08/11/2023 23:26:38

Explicação:

A remoção solicitada é do primeiro elemento da lista encadeada. Para realizar esse tipo de remoção, basta apontar a variável que guarda o primeiro elemento (L) para o endereço do segundo elemento. Este endereço está armazenado no campo próximo do primeiro elemento. Ou seja, a variável L deverá apontar para 128.

A resposta endereço 24 conterà a chave 5 está errada pois na lista encadeada, os elementos não precisam ser puxados após uma remoção.

A resposta endereço 32 terá seu campo próximo alterado está errada, pois isso adicionaria um elemento ao final da lista, no caso tornando-a circular.

As demais respostas estão erradas pois nada será apagado.



Questão

Acerto: 0,2 / 0,2

A raiz é o ponto de partida para acessar todos os elementos de uma árvore. Marque a opção correta acerca dos principais conceitos de árvore binária de busca:

- ☐ Novas chaves maiores que a raiz sempre serão inseridas à esquerda.
- ☒ Em todas as estruturas de dados onde se realiza busca, inserção e remoção não são admitidas duplicidade de chaves. Isto também inclui as árvores binárias de busca.
- ☐ Qualquer nó pode ter um número arbitrário de nós, sempre maior que 2.
- ☐ Dado um nó qualquer da árvore binária, todos os nós à direita dele são menores ou iguais a ele.
- ☐ O objetivo principal da estrutura de dados árvore binária de busca é ordenar uma lista sem a preocupação de implementar de forma eficientemente.

Respondido em 08/11/2023 23:28:52

Explicação:


O grau máximo de um nó em uma árvore binária é 2. A unicidade de chave é um pressuposto para estruturas de busca. O objetivo principal de uma árvore binária de busca é implementar os algoritmos de busca, inserção e remoção de forma otimizada. Chaves maiores que a raiz devem ser inseridas à direita. Dado qualquer nó de uma árvore binária de busca, deve valer recursivamente a propriedade de que as chaves contidas à esquerda são menores que a raiz e a direita maiores.



Questão

Acerto: 0,2 / 0,2

Em uma Árvore B, temos que: Cada nó contém no mínimo m registros ($m+1$ descendentes) e no máximo $2m$ registros (e $2m+1$ descendentes), exceto o nó que é raiz que pode conter entre 1 e $2m$ registros e todos os nós folhas aparecem no mesmo nível. Sobre Árvores B, é correto afirmar:

- ☐ O particionamento de nós em uma Árvore B ocorre quando um registro precisa ser inserido em um nó com menos de $2m$ registros.
- ☐ O particionamento de nós ocorre quando é necessário diminuir a altura da árvore.
- ☐ O particionamento de nós em uma Árvore B ocorre quando um registro precisa ser buscado em um nó com $2m + 1$ registros.
- ☐ O particionamento de nós em uma Árvore B ocorre quando a chave do registro a ser inserido contém um valor(conteúdo) intermediário entre os valores das chaves dos registros contidos no mesmo nó.
- ☒  O particionamento de nós em uma Árvore B ocorre quando um registro precisa ser inserido em um nó com $2m$ registros.

Respondido em 08/11/2023 23:32:05

Explicação:

O particionamento de nós em uma Árvore B ocorre quando um registro precisa ser inserido em um nó com $2m$ registros.



Questão

Acerto: 0,2 / 0,2

O uso de funções recursivas pode facilitar a implementação de diversos algoritmos. Toda recursão depende de dois elementos: o caso base e o passo recursivo. Dentre as opções a seguir, a que apresenta um passo recursivo é:

- ☐ $f(n)=g(n-1)$
- ☐ $fat(1)=1$
- ☐ $par(n)=par(n)$
- ☐ $fib(n)=n-1 + n-2$
- ☒ $fat(n)=n*fat(n-1)$

Respondido em 08/11/2023 23:32:24

Explicação:

O passo recursivo é o elemento que faz o cálculo da função recursiva mover-se em direção ao resultado. Deve envolver a chamada da própria função com um valor diferente de entrada. Isso só acontece na resposta correta: $fat(n)=n*fat(n-1)$, passo recursivo da função de cálculo de fatorial.

$fat(1)=1$ é o caso base dessa mesma função. $par(n)=par(n)$ é uma tautologia, e não uma recursão. As demais respostas são funções que não chamam a si mesmas, não podendo ser passos recursivos.



Questão

Acerto: 0,2 / 0,2

Uma Deque é uma estrutura de dados mais generalista que as pilhas e filas. Para implementá-la de forma eficiente, você pode usar:

- ☐ Pilha com 1 variável: topo.
- ☒ Lista duplamente encadeada com 2 variáveis: início e final.
- ☐ Lista simplesmente encadeada com nó cabeça.
- ☐ Fila com 2 variáveis: início e final.
- ☐ Lista contígua com 1 variável: início.

Respondido em 08/11/2023 23:32:38

Explicação:

Para implementar uma deque eficientemente, você precisa ter um ponteiro para o início e o final da deque, permitindo inserções e remoções em ambas as pontas com complexidade $O(1)$, sem a necessidade de percorrer a estrutura, o que seria $O(n)$.

Além disso, a fila é uma especialização da deque. Ou seja, toda fila é um deque, mas nem toda deque é uma fila. Podemos assim eliminar a resposta contendo fila. A resposta restante que possui 2 variáveis é a correta. Lista duplamente encadeada. Ela permite a inserção e remoção nas extremidades com complexidade $O(1)$.

A lista contígua e a simplesmente encadeada com nó cabeça levariam a operação de inserção e remoção ao final da fila terem complexidade $O(n)$ por precisarem percorrer toda a estrutura, sendo também descartadas.



7ª

Questão

Acerto: 0,2 / 0,2

As operações de busca, remoção e inserção de nós em uma **árvore binária de busca** levam determinado tempo de execução de seus algoritmos. Esses tempos são dados pela alternativa:

- ☐ Busca: $O(\log n)$ / Remoção: $O(n)$ / Inserção: $O(\log n)$
- ☒ Busca: $O(n)$ / Remoção: $O(n)$ / Inserção: $O(n)$
- ☐ Busca: $O(n)$ / Remoção: $O(\log n)$ / Inserção: $O(\log n)$
- ☐ Busca: $O(n)$ / Remoção: $O(n)$ / Inserção: $O(\log n)$
- ☐ Busca: $O(1)$ / Remoção: $O(\log n)$ / Inserção: $O(\log n)$

Respondido em 08/11/2023 23:33:54

Explicação:

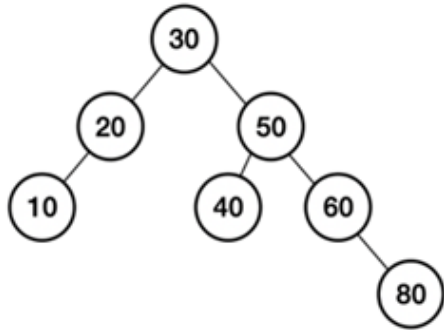
No pior caso uma árvore binária de busca com n chaves tem n níveis. Assim, o pior caso da busca, é buscar o nó mais profundo da árvore que demandará n comparações. Como a busca é subrotina da inserção e da remoção, então as três operações terão complexidade de pior caso de $O(n)$.


8ª

Questão

Acerto: 0,2 / 0,2

Seja a seguinte árvore AVL abaixo. Com a inserção da chave 90, marque a opção que indica exatamente o que acontecerá com a árvore resultante após essa inserção:

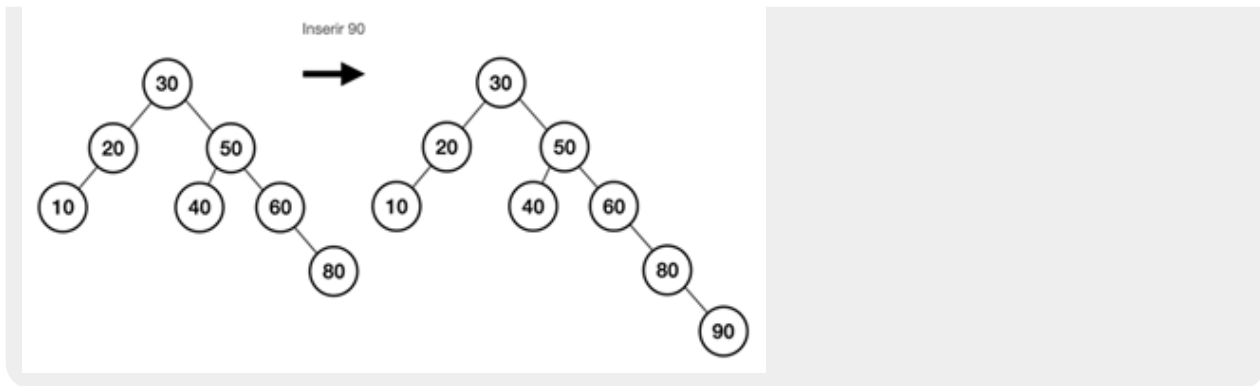


- ☐ A árvore resultante irá desbalancear à direita do nó de chave 40.
- ☐ A árvore resultante irá desbalancear à direita do nó de chave 80.
- ☐ A árvore resultante irá manter o balanceamento geral da árvore.
- ☒  A árvore resultante irá desbalancear à esquerda do nó de chave 60.
- ☐ A árvore resultante irá desbalancear à esquerda do nó de chave 10.

Respondido em 08/11/2023 23:34:15

Explicação:

Ao inserir o nó de chave 90, ele é maior que o nó 80, sendo assim, inserido ao lado direito de 80, causando desbalanceamento do nó 60 que tem altura da subárvore direita 2 e esquerda 0.



9ª

Questão

Acerto: 0,2 / 0,2

Dada a seguinte matriz M, inicializada com o código:

$M = [[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12], [13, 14, 15, 16]]$

O código em Python para imprimir cada elemento da coluna iniciada pelo elemento 3 é:

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

☐ for linha in M:

- ☐ `print(linha)`
- ☐ `print(M[2])`
- ☐ `for linha in M:`
- ☒ `print(linha[3])`
- ☒ `for linha in M:`
- ☐ `print(linha[2])`
- ☐ `for coluna in M:`
- `print(coluna)`

Respondido em 08/11/2023 23:34:56

Explicação:

O laço deve percorrer uma coluna, iterando linha a linha e extraíndo dela o seu terceiro elemento, ou seja `linha[2]`. A resposta correta itera pelas linhas e imprime o elemento `[2]` de cada uma.

Dentre as respostas erradas, apenas escrever `print(linha)` imprimirá cada linha como um todo, resultando na impressão de toda a matriz, linha a linha.

A resposta "`print(coluna)`" terá o mesmo resultado pois para o código `linha` e `coluna` são apenas nomes escolhidos pelo programador. Poderia ser `i`, `aux` ou qualquer outra variável escolhida.

Já "`print(linha[3])`" está com o índice errado, imprimindo os elementos da coluna iniciada por 4. E `print(M[2])` imprime toda a linha iniciada por 9.

Em uma implementação da estrutura de dados do tipo fila, você possui um espaço de memória contíguo a ela alocada com capacidade para M nós. A variável da fila é F, e duas variáveis guardam os índices do início e final da fila (inícioF e finalF). Em uma implementação otimizada de F, como podemos identificar que a fila está cheia?

- ☐ FinalF == M
- ☒ InícioF == (finalF + 1) mod M
- ☐ InícioF = M
- ☐ InícioF == finalF
- ☐ InícioF == finalF + 1

Respondido em 08/11/2023 23:35:49

Explicação:

Em uma implementação otimizada da fila, é usado um sistema modular, onde o início e o final da fila se movem a cada inserção e remoção. A cada inserção, finalF aumenta em 1, até o máximo M, depois volta para 0 e assim por diante. A cada remoção inícioF aumenta em 1, até o máximo M e depois volta a 0. dessa forma a fila está cheia quando $(\text{finalF} + 1) \bmod M$ é igual a início.