Disciplina: ESTRUTURA DE DADOS
Aluno: XXX XXXXX
Professor: ROBSON LORBIESKI

AV
XXXXXXXX
Turma: 9001

Avaliação: 6,00 pts | Nota SIA: 7,00 pts

7390 - ALGORITMOS E A LINGUAGEM PYTHON

1. Ref.: 7805522 — Pontos: 1,00 / 1,00

Os algoritmos são utilizados em diversos programas de computador para auxiliar no seu funcionamento correto. Dentre suas principais características temos:

- [ ] Passos ambíguos.
- [ ] Complexidade infinita.
- [x] Encerramento garantido.
- [ ] Solução ótima.
- [ ] Solução garantida.

2. Ref.: 7805518 — Pontos: 1,00 / 1,00

Um vetor ou array é uma estrutura de dados simples que armazena elementos sequencialmente em memória. O tamanho em memória necessário para armazenar um vetor de 34 elementos onde cada elemento é uma variável inteira que ocupa 2 bytes é:

- [ ] 34 bytes.
- [x] 68 bytes.
- [ ] 256 bytes.
- [ ] 1156 bytes.
- [ ] 136 bytes.

7391 - LISTAS, PILHAS, FILAS E DEQUES

3. Ref.: 7805594 — Pontos: 0,00 / 1,00

Considerando que em uma estrutura do tipo lista circular simplesmente encadeada e com nó cabeça, a inserção ocorre sempre no início da lista, quais são os passos para realizar a inserção de um novo nó?

- [x] Percorrer a lista até o último nó, apontar o último nó para o novo nó, apontar o novo nó para o nó cabeça.
- [ ] Percorrer a lista até o último nó, apontar o último nó para o novo nó, apontar o novo nó para o último nó.
- [ ] Apontar o nó cabeça para o novo nó, apontar o novo nó para nulo.
- [ ] Apontar o novo nó para o nó cabeça, apontar o nó cabeça para o novo nó.
- [ ] Apontar o novo nó para o seguinte ao nó cabeça, apontar o nó cabeça para o novo nó.

4. Ref.: 7805550 — Pontos: 1,00 / 1,00

Uma Fila é uma estrutura de dados que permite o armazenamento de elementos (ou nós) sequencialmente. Sobre as Filas é possível afirmar que:

- [ ] Permitem inserção ou remoção em qualquer de suas posições.
- [ ] Permitem inserção ou remoção apenas no seu início ou no seu final.
- [x] Permitem inserção no seu final e remoção apenas no seu início.
- [ ] Permitem inserção ou remoção apenas no seu início.
- [ ] Permitem inserção no seu início e remoção apenas no seu final.

5. Ref.: 7805568 Pontos: 1,00 / 1,00

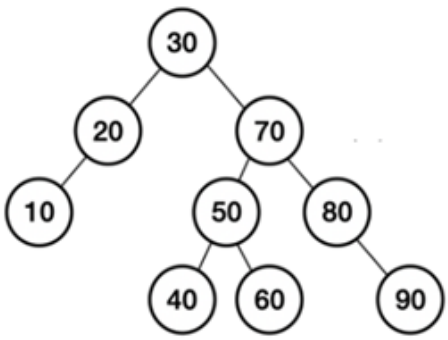
Uma Deque é uma estrutura de dados que permite o armazenamento de elementos (ou nós) sequencialmente. Sobre as Deques é possível afirmar que:

- [ ] Permitem inserção no seu início e remoção apenas no seu final.
- [x] Permitem inserção ou remoção apenas no seu início ou no seu final.
- [ ] Permitem inserção ou remoção apenas no seu início.
- [ ] Permitem inserção ou remoção em qualquer de suas posições.
- [ ] Permitem inserção no seu final e remoção apenas no seu início.

7392 - ÁRVORES DE BUSCA

6. Ref.: 7805541 Pontos: 0,00 / 1,00

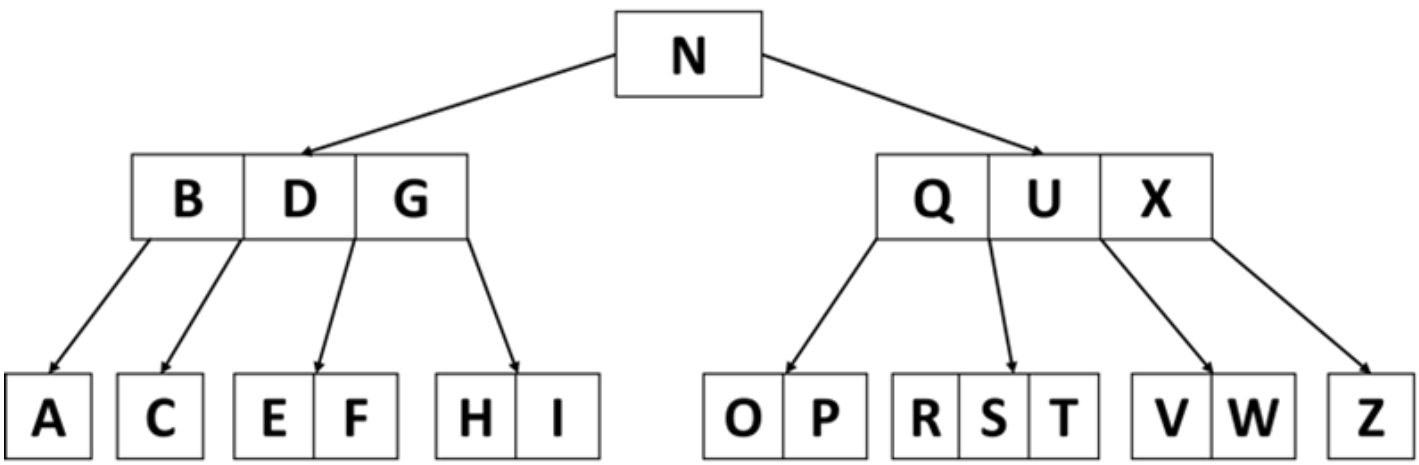
Seja a seguinte árvore AVL abaixo. Com a inserção da chave 65, marque a opção que indica exatamente o que acontecerá com a árvore resultante após essa inserção:



- [ ] Irá desbalancear o nó raiz da árvore AVL.
- [ ] Irá desbalancear o nó 30 à direita.
- [ ] O 65 será inserido à esquerda do nó 80, não causando desbalanceamento.
- [x] Irá desbalancear o nó 70 à esquerda.
- [ ] Irá desbalancear o nó 20 à direita.

7. Ref.: 7805536 Pontos: 1,00 / 1,00

Seja a operação de busca de chaves em uma Árvore B. Na seguinte árvore B abaixo, o resultado da sequência de chaves visitadas até encontrar a chave S é:



- [x] N-Q-S.

- ☐ N-R-S.
- ☐ N-X-S.
- ☐ N-U-S.
- ☐ N-T-S.

8. Ref.: 7805544 Pontos: 0,00 / 1,00

Sobre as árvores binárias de busca balanceadas, analise as afirmativas abaixo:

I - Tem altura proporcional a log n.

II - As árvores completas são balanceadas.

III - Existe algoritmo capaz de transformar uma árvore binária de busca não balanceada em balanceada em O(n).

IV - Toda árvore balanceada é completa.

V - A busca ocorre em um tempo proporcional a log n nas árvores balanceadas.

- ☐ I, II, III, IV e V são corretas.
- ☐ I, III, IV e V são corretas.
- ☐ ✔ I, II, III e V são corretas.
- ☒ ✘ I, II, IV e V são corretas.
- ☐ I, II, III e IV são corretas.

7408 - ÁRVORES EM PHYTON

9. Ref.: 7805610 Pontos: 1,00 / 1,00

Seja o seguinte código em Python cujo principal objetivo é implementar uma árvore binária. Marque a alternativa correta quanto a execução do código:

```
class NoArvore:
    def __init__(self, chave = None, esquerda = None, direita = None):
        self.chave = chave
        self.esquerda = esquerda
        self.direita = direita

if __name__ == '__main__':
    raiz = NoArvore(55)

    raiz.esquerda = NoArvore(35)
    raiz.direita  = NoArvore(75)

    raiz.direita.esquerda  = NoArvore(65)
    raiz.direita.direita   = NoArvore(85)
    raiz.esquerda.esquerda = NoArvore(25)
    raiz.esquerda.direita  = NoArvore(45)
```

- ☐ A árvore criada no código é binária de busca com altura 6, isto é, com 6 níveis distintos.
- ☐ A árvore criada no código acima não é binária de busca.
- ☐ A classe NotArvore implementa regras que garantem que os nós inseridos respeitam a ordem de inserção dos nós (maiores a direita e menores a esquerda).
- ☐ Não é possível inferir a topologia da árvore com base no código.
- ☒ ✔ A árvore criada no código acima é uma árvore binária de busca com todas as folhas no último nível.

10. Ref.: 7805614 Pontos: 0,00 / 1,00

Seja a seguinte função **funcaoBST** em Python que executa uma operação em uma árvore binária de busca:

```
def  funcaoBST (raiz, chave):
    if raiz is None:
        return NoArvore(chave)
    else:
        if raiz.chave == chave:
            return raiz
        elif raiz.chave < chave:
            raiz.direita = funcaoBST(raiz.direita, chave)
        else:
            raiz.esquerda = funcaoBST(raiz.esquerda, chave)
    return raiz
```

O que é executado na função acima é:

- [x] Inserção de chave em árvore binária de busca.
- [ ] Percurso de chaves em uma árvore binária de busca.
- [ ] Busca de chaves em árvore binária de busca.
- [ ] Remoção de chave em uma árvore binária de busca.
- [x] Percurso *in-ordem* em árvore binária.