Disciplina: **ESTRUTURA DE DADOS** AV

Aluno: [redacted]

Professor: **ROBSON LORBIESKI**

Turma: **9001**

DGT1335_AV_[redacted]

Avaliação: **7,00** pts Nota SIA: **9,00** pts

7390 - ALGORITMOS E A LINGUAGEM PYTHON

1. Ref.: 7805519 Pontos: **1,00** / **1,00**

Ao usar laços em Python, você pode facilmente iterar sobre um vetor sem se preocupar em criar uma variável especifica como contador. Entretanto, para algumas aplicações é necessário controlar qual é o número atual de execuções de um laço. Uma das soluções é a criação e manutenção de um contador. Outra solução é o uso de uma função intrínseca do Python, chamada:

- [x] enumerate.
- [ ] count.
- [ ] length.
- [ ] index.
- [ ] find.

2. Ref.: 7805520 Pontos: **1,00** / **1,00**

Durante o uso de vetores e matrizes em um programa, o programador deve ter cuidado com a manipulação dos índices, pois:

- [x] O uso de um índice igual ao tamanho do vetor causará um erro de execução.
- [ ] Não é possível utilizar índices para acessar elementos de matrizes.
- [ ] O uso de um índice maior que o tamanho do vetor causará um erro de compilação.
- [ ] O uso de um índice positivo menor que o tamanho do vetor causará erro de parsing.
- [ ] O uso de um índice negativo sempre causará erro de execução em Python.

7391 - LISTAS, PILHAS, FILAS E DEQUES

3. Ref.: 7805525 Pontos: **1,00** / **1,00**

Uma Lista é uma estrutura de dados simples, que permite o armazenamento de elementos (ou nós) sequencialmente. Sobre as Listas é possível afirmar que:

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

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:

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

5. Ref.: 7805594 Pontos: **0,00** / **1,00**

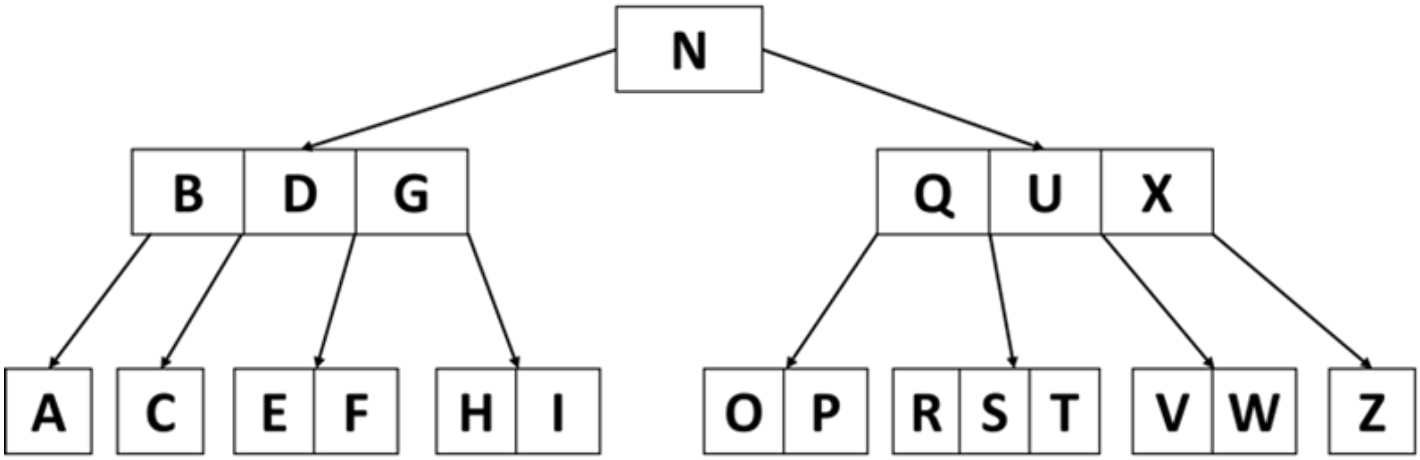
Considerando que em uma estrutura do tipo lista circular simplesmente encadeada e com nó cabeça, a inserção ocorre sempre no início da lista, quais são os passos para realizar a inserção de um novo nó?

- ☐ 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 novo nó para o seguinte ao nó cabeça, apontar o nó cabeça para o novo nó.
- ☐ Percorrer a lista até o último nó, apontar o último nó para o novo nó, apontar o novo nó para o nó cabeça.
- ☐ 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.

7392 - ÁRVORES DE BUSCA

6. Ref.: 7805536 Pontos: **0,00** / **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:



- ☐ N-X-S.
- ☐ N-U-S.
- ☒ ✘ N-R-S.
- ☐ ✔ N-Q-S.
- ☐ N-T-S.

7. Ref.: 7805562 Pontos: **1,00** / **1,00**

Árvores de busca são estruturas de dados que permitem armazenar e recuperar informações de maneira eficiente. Marque a opção correta sobre árvores perfeitamente balanceadas:

- ☐ Toda árvore balanceada é estruturada em zig-zag.
- ☐ Toda árvore balanceada tem altura proporcional à O(n) .
- ☐ Toda árvore balanceada tem altura maior 3.
- ☒ ✔ Toda árvore perfeitamente balanceada tem altura proporcional a log n.
- ☐ Toda árvore balanceada é complexa.

8. Ref.: 7805543 Pontos: **1,00** / **1,00**

As árvores binárias de busca são especializações das árvores binárias que permitem uma melhor organização dos algoritmos de busca. Sobre a inserção de uma nova chave em uma árvore binária de busca é correto afirmar que:

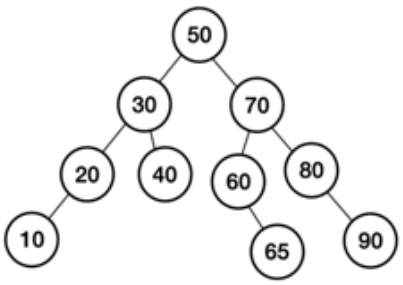
- ☐ A complexidade da inserção é sempre O(n), independentemente da altura da árvore.

- ☐ Para determinar a posição da nova chave é necessário calcular o percurso em ordem simétrica da árvore obtida. Com este percurso, verifica-se se a sequência está ordenada em ordem crescente. Caso esteja, a posição da nova chave está correta.
- ☐ O algoritmo de inserção em árvores binárias de busca é estático, isto é, é necessário recalcular toda árvore para inserir uma nova chave.
- ☐ Toda nova chave é inserida obrigatoriamente na raiz.
- ☒ ✔ Todas as chaves são inseridas em folhas, a posição da folha é determinada pela busca.

7408 - ÁRVORES EM PHYTON

9. Ref.: 7805602 Pontos: **0,00** / **1,00**

Seja a seguinte árvore binária de busca, marque a alternativa correta:



- ☐ A árvore binária de busca acima possui 4 nós folhas e 2 raízes.
- ☐ O certo em uma árvore binária de busca é que os nós possuam no mínimo grau 3. Logo, essa regra não é respeitada na árvore acima.
- ☐ O nó de chave 40 está inserido no lado errado, pois deveria estar ao lado do nó 20.
- ☒ ✖ Existe um erro conceitual na árvore binária acima, pois os nós 90 e 65 deveriam estar dispostos do lado esquerdo da árvore em relação à raiz.
- ☐ ✔ Todos os nós da árvore binária estão corretamente dispostos na árvore, respeitando as regras conceituais de árvores binárias de busca.

10. Ref.: 7805622 Pontos: **1,00** / **1,00**

Seja a função de percurso *in-ordem* em Python. Marque a opção que apresenta a complexidade de execução:

- ☒ ✔ A complexidade computacional do algoritmo para percurso em ordem simétrica é $O(n)$.
- ☐ A complexidade computacional do algoritmo para percurso em ordem simétrica é $O(1)$.
- ☐ A complexidade computacional do algoritmo para percurso em ordem simétrica constante.
- ☐ A complexidade computacional do algoritmo para percurso em ordem simétrica é $O(n \log n)$.
- ☐ A complexidade computacional do algoritmo para percurso em ordem simétrica é $O(\log n)$.