

Perguntas de AED no ano de 2019

Big oh, Algoritmo, ficam em baixo as respectivas soluções.

Pergunta 1
Correto
Nota: 2.00 em 2.00
Marcar pergunta

Qual das seguintes ordens de complexidade (usando a notação Big-Oh) indica um algoritmo menos eficiente?

Selecione uma opção de resposta:

- ☐ $O(N \log_2 N)$
- ☒ $O(N^3)$ ✓
- ☐ $O(N^2)$
- ☐ $O(N)$

Pergunta 3
Correto
Nota: 2.00 em 2.00
Marcar pergunta

Qual das seguintes ordens de complexidade (usando a notação Big-Oh) indica um algoritmo menos eficiente?

Selecione uma opção de resposta:

- ☒ $O(N^3)$ ✓
- ☐ $O(N)$
- ☐ $O(N^2)$
- ☐ $O(\log_2 N)$

Mais eficiente!

Pergunta 4
Correto
Nota: 2.00 em 2.00
Marcar pergunta

Qual das seguintes ordens de complexidade (usando a notação Big-Oh) indica um algoritmo mais eficiente?

Selecione uma opção de resposta:


- ☐ $O(N^2)$
- ☐ $O(N^3)$
- ☐ $O(N \log_2 N)$
- ☒ $O(N)$ ✓

Pergunta 3
Incorreto
Nota: 0.00 em 2.00
Marcar pergunta

Qual das seguintes ordens de complexidade (usando a notação Big-Oh) indica um algoritmo mais eficiente?

Selecione uma opção de resposta:

- ☐ $O(N^3)$
- ☐ $O(\log_2 N)$
- ☐ $O(N^2)$
- ☒ $O(N)$ ✗



Pergunta da propriedade

Pergunta 4
Correto
Nota: 2.00 em 2.00
Marcar pergunta

Para demonstrarmos, através da Indução Matemática, que uma propriedade $P(n)$ é verdadeira $\forall n \in \mathbb{N}$, devemos provar que:

Selecione uma opção de resposta:

- ☐ $P(1)$ é verdadeira e $P(n+1) \Rightarrow P(n)$
- ☐ $P(1)$ é falsa e $P(n) \Rightarrow P(n+1)$
- ☐ $P(1)$ é falsa e $P(n-1) \Rightarrow P(n)$
- ☒ $P(1)$ é verdadeira e $P(n) \Rightarrow P(n+1)$ ✓

Pergunta solta

Pergunta 2
Correto
Nota: 2.00 em 2.00
Marcar pergunta

Ocorre recursividade direta quando:

Selecione uma opção de resposta:

- ☐ nenhuma das outras opções
- ☐ uma função recebe sempre os mesmos valores nos parâmetros
- ☐ uma função chama uma outra função
- ☒ uma função se chama a si própria ✓

Big-Oh \rightarrow Ordem de complexidade!

Pergunta 1

Correto

Nota: 4.00 em 4.00

🚩 Marcar pergunta

Usando a notação Big-Oh, qual a ordem de complexidade da execução do seguinte bloco de código?

```
for (int i = 0; i < n; i++) {  
    for (int j = 0; j < i * n; j++) {  
        for (int k = 0; k < j * i * j; k++) {  
            System.out.println(k);  
        }  
    }  
}
```

Selecione uma opção de resposta:

- ☒ $O(N^8)$ ✓
- ☐ $O(N^7)$
- ☐ $O(N^5)$
- ☐ $O(N^6)$

Pergunta 5

Correto

Nota: 4.00 em 4.00

🚩 Marcar pergunta

Usando a notação Big-Oh, qual a ordem de complexidade da execução do seguinte bloco de código?

```
for (int i = 0; i < n; i++) {  
    for (int j = 0; j < i * n; j++) {  
        for (int k = 0; k < j * i * n; k++) {  
            System.out.println(k);  
        }  
    }  
}
```

Selecione uma opção de resposta:

- ☐ $O(N^5)$
- ☒ $O(N^7)$ ✓
- ☐ $O(N^8)$
- ☐ $O(N^6)$

HARMÓNICOS

Pergunta 2

Correto

Nota: 3.00 em 3.00

🚩 Marcar pergunta

Em matemática, a função que calcula números harmônicos é dada por:

$$H(n) = \begin{cases} 1 & \Leftarrow n = 1 \\ \frac{1}{n} + H(n-1) & \Leftarrow n \in \mathbb{N} \wedge n > 1 \end{cases}$$

Considere a seguinte implementação:

```
private double h(int n) {  
    if (n < 1) {  
        throw new IllegalArgumentException("n não pode ser inferior a 1");  
    }  
    if (n == 1) {  
        return 0;  
    }  
    return 1.0 / n + h(n - 1);  
}
```

Há algum problema nessa implementação?

Selecione uma opção de resposta:

- ☐ sim, no tipo de retorno
- ☐ sim, a chamada recursiva não converge
- ☒ sim, o caso base está mal definido ✓
- ☐ não

Pergunta 3

Correto

Nota: 3.00 em
3.00🚩 Marcar
pergunta

Em matemática, a função que calcula números harmônicos é dada por:

$$H(n) = \begin{cases} 1 & \Leftarrow n = 1 \\ \frac{1}{n} + H(n-1) & \Leftarrow n \in \mathbb{N} \wedge n > 1 \end{cases}$$

Considere a seguinte implementação:

```
private double h(int n) {  
    if (n < 1) {  
        throw new IllegalArgumentException("n não pode ser inferior a 1");  
    }  
    if (n == 1) {  
        return 1;  
    }  
    return 1.0 / n + h(n + 1);  
}
```

Há algum problema nessa implementação?

Selecione uma opção de resposta:

- ☐ não
- ☐ sim, no tipo de retorno
- ☒ sim, a chamada recursiva não converge ✓
- ☐ sim, o caso base está mal definido

Pergunta 5

Correto

Nota: 3.00 em
3.00🚩 Marcar
pergunta

Em matemática, a função que calcula números harmônicos é dada por:

$$H(n) = \begin{cases} 1 & \Leftarrow n = 1 \\ \frac{1}{n} + H(n-1) & \Leftarrow n \in \mathbb{N} \wedge n > 1 \end{cases}$$

Considere a seguinte implementação:

```
private double h(int n) {  
    if (n < 1) {  
        throw new IllegalArgumentException("n não pode ser inferior a 1");  
    }  
    if (n == 1) {  
        return 1;  
    }  
    return 1.0 / n + h(n - 1);  
}
```

Há algum problema nessa implementação?

Selecione uma opção de resposta:

- ☐ sim, no tipo de retorno
- ☒ não ✓
- ☐ sim, a chamada recursiva não converge
- ☐ sim, o caso base está mal definido

Pergunta 5

Por responder

Nota de 3.00

Marcar pergunta

Em matemática, a função que calcula números harmônicos é dada por:

$$H(n) = \begin{cases} 1 & \Leftarrow n = 1 \\ \frac{1}{n} + H(n-1) & \Leftarrow n \in \mathbb{N} \wedge n > 1 \end{cases}$$

Considere a seguinte implementação:

```
private long h(int n) {  
    if (n < 1) {  
        throw new IllegalArgumentException("n não pode ser inferior a 1");  
    }  
    if (n == 1) {  
        return 1;  
    }  
    return 1.0 / n + h(n - 1);  
}
```

Há algum problema nessa implementação?

Selecione uma opção de resposta:

- ☐ sim, o caso base está mal definido
- ☐ não
- ☒ sim, no tipo de retorno
- ☐ sim, a chamada recursiva não converge

Pergunta de pesquisa

Considere uma função de pesquisa que devolva a lista de índices (indicativos das posições) de um dado elemento numa sequência ordenada de elementos. Indique qual das seguintes implementações resolve o problema de forma mais eficiente.

Selecione uma opção de resposta:

☐ a.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,  
    T elemento, T... elementos) {  
    LinkedList<Integer> indices = new LinkedList<>();  
    for (int i = 0; i < elementos.length; i++) {  
        estatistica.incrementarComparacoes();  
        if (criterio.comparar(elemento, elementos[i]) == 0) {  
            indices.add(i);  
            while (criterio.comparar(elemento, elementos[i]) == 0 && ++i < elementos.length) {  
                indices.add(i);  
            }  
            return indices;  
        }  
    }  
    return indices;  
}
```

☐ b.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,  
    T elemento, T... elementos) {  
    LinkedList<Integer> indices = new LinkedList<>();  
    for (int i = 0; i < elementos.length; i++) {  
        estatistica.incrementarComparacoes();  
        if (criterio.comparar(elemento, elementos[i]) == 0) {  
            indices.add(i);  
        }  
    }  
    return indices;  
}
```

☒ c.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,  
    T elemento, T... elementos) {  
    LinkedList<Integer> indices = new LinkedList<>();  
    for (int i = 0; i < elementos.length; i++) {  
        estatistica.incrementarComparacoes();  
        if (criterio.comparar(elemento, elementos[i]) == 0) {  
            indices.add(i);  
            while (++i < elementos.length && criterio.comparar(elemento, elementos[i]) == 0) {  
                indices.add(i);  
            }  
            return indices;  
        }  
    }  
    return indices;  
}
```

☐ d.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,  
    T elemento, T... elementos) {  
    LinkedList<Integer> indices = new LinkedList<>();  
    for (int i = 0; i < elementos.length; i++) {  
        estatistica.incrementarComparacoes();  
        if (criterio.comparar(elemento, elementos[i]) == 0) {  
            indices.add(i);  
            return indices;  
        }  
    }  
    return indices;  
}
```

Pergunta 2

Correto

Nota: 6,00 em
6,00

Considere uma função de pesquisa que devolva a lista de índices (indicativos das posições) de um dado elemento numa sequência ordenada de elementos.
Indique qual das seguintes implementações resolve o problema de forma eficiente.

Selecione uma opção de resposta:

- ☐ a.)
- ```
estatistica.incrementarComparacoes();
if (criterio.comparar(elemento, elementos[i]) == 0) {
 indices.add(i);
 while (criterio.comparar(elemento, elementos[i]) == 0 && ++i < elementos.length) {
 indices.add(i);
 }
 return indices;
}
return indices;
```
- ☐ b. )
- ```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,  
    T elemento, T... elementos) {  
    LinkedList<Integer> indices = new LinkedList<>();  
    for (int i = 0; i < elementos.length; i++) {  
        estatistica.incrementarComparacoes();  
        if (criterio.comparar(elemento, elementos[i]) == 0) {  
            indices.add(i);  
            while (++i < elementos.length && criterio.comparar(elemento, elementos[i]) == 0) {  
                indices.add(i);  
            }  
            return indices;  
        }  
    }  
    return indices;  
}
```
- ☐ c.)
- ```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
 T elemento, T... elementos) {
 LinkedList<Integer> indices = new LinkedList<>();
 for (int i = 0; i < elementos.length; i++) {
 estatistica.incrementarComparacoes();
 if (criterio.comparar(elemento, elementos[i]) == 0) {
 indices.add(i);
 return indices;
 }
 }
 return indices;
}
```
- ☒ d. )
- ```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,  
    T elemento, T... elementos) {  
    LinkedList<Integer> indices = new LinkedList<>();  
    for (int i = 0; i < elementos.length; i++) {  
        estatistica.incrementarComparacoes();  
        if (criterio.comparar(elemento, elementos[i]) == 0) {  
            indices.add(i);  
        }  
    }  
    return indices;  
}
```

Pergunta 5

Correto

Nota: 6.00 em 6.00

Marcar pergunta

Considere uma função de pesquisa que devolva a lista de índices (indicativos das posições) de um dado elemento numa sequência ordenada de elementos. Indique qual das seguintes implementações resolve o problema de forma eficiente.

Selecione uma opção de resposta:

☒ a. Nenhuma

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
                                     T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (criterio.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
            return indices;
        }
    }
    return indices;
}
```

☐ b.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
                                     T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (criterio.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
        }
    }
    return indices;
}
```

☐ c.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
                                     T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (criterio.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
            while (criterio.comparar(elemento, elementos[i]) == 0 && ++i < elementos.length) {
                indices.add(i);
            }
            return indices;
        }
    }
    return indices;
}
```

☐ d.

Considere uma função de pesquisa que devolva a lista de índices (indicativos das posições) de um dado elemento numa sequência ordenada de elementos. Indique qual das seguintes implementações resolve o problema de forma eficiente.

Selecione uma opção de resposta:

☒ a.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
                                     T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (criterio.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
        }
    }
    return indices;
}
```



☐ b.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
                                     T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (criterio.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
            while (criterio.comparar(elemento, elementos[i]) == 0 && ++i < elementos.length) {
                indices.add(i);
            }
            return indices;
        }
    }
    return indices;
}
```

☐ c. Nenhuma



d.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
                                     T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (criterio.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
            return indices;
        }
    }
    return indices;
}
```


Pergunta 1

Correto

Nota: 6.00 em 6.00

Marcar pergunta

Considere uma função de pesquisa que devolva a lista de índices (Indicativos das posições) de um dado elemento numa sequência não ordenada de elementos.
Indique qual das seguintes implementações resolve o problema.

Selecione uma opção de resposta:

- ☐ a.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
    T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (critério.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
            while (++i < elementos.length && critério.comparar(elemento, elementos[i]) == 0) {
                indices.add(i);
            }
            return indices;
        }
    }
    return indices;
}
```
- ☒ b.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
    T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (critério.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
        }
    }
    return indices;
}
```
- ☐ c.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
    T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (critério.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
            return indices;
        }
    }
    return indices;
}
```
- ☐ d.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
    T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (critério.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
            while (critério.comparar(elemento, elementos[i]) == 0 && ++i < elementos.length) {
                indices.add(i);
            }
            return indices;
        }
    }
    return indices;
}
```

Pergunta 2

Incorreto

Nota: 0.00 em 6.00

Marcar pergunta

Considere uma função de pesquisa que devolva a lista de índices (Indicativos das posições) de um dado elemento numa sequência não ordenada de elementos.
Indique qual das seguintes implementações resolve o problema.

Selecione uma opção de resposta:

- ☐ a.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
    T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (critério.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
        }
    }
    return indices;
}
```
- ☐ b.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
    T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (critério.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
            while (++i < elementos.length && critério.comparar(elemento, elementos[i]) == 0) {
                indices.add(i);
            }
            return indices;
        }
    }
    return indices;
}
```
- ☐ c.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
    T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (critério.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
            return indices;
        }
    }
    return indices;
}
```
- ☐ d.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
    T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (critério.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
            while (critério.comparar(elemento, elementos[i]) == 0 && ++i < elementos.length) {
                indices.add(i);
            }
            return indices;
        }
    }
    return indices;
}
```

Pergunta 1

Correto

Nota: 6.00 em 6.00

Marcar pergunta

Considere uma função de pesquisa que devolva a lista de índices (indicativos das posições) de um dado elemento numa sequência não ordenada de elementos. Indique qual das seguintes implementações resolve o problema.

Selecione uma opção de resposta:

- ☐ a.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
    T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (criterio.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
            while (++i < elementos.length && criterio.comparar(elemento, elementos[i]) == 0) {
                indices.add(i);
            }
        }
    }
    return indices;
}
```
- ☐ b.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
    T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (criterio.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
        }
    }
    return indices;
}
```
- ☐ c.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
    T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (criterio.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
            while (criterio.comparar(elemento, elementos[i]) == 0 && ++i < elementos.length) {
                indices.add(i);
            }
        }
    }
    return indices;
}
```
- ☒ d.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
    T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (criterio.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
        }
    }
    return indices;
}
```

Pergunta 4

Correto

Nota: 6.00 em 6.00

Marcar pergunta

Considere uma função de pesquisa que devolva a lista de índices (indicativos das posições) de um dado elemento numa sequência não ordenada de elementos. Indique qual das seguintes implementações resolve o problema.

Selecione uma opção de resposta:

- ☒ a. Nenhuma
- ☐ b.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
    T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (criterio.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
            while (criterio.comparar(elemento, elementos[i]) == 0 && ++i < elementos.length) {
                indices.add(i);
            }
        }
    }
    return indices;
}
```
- ☐ c.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
    T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (criterio.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
        }
    }
    return indices;
}
```
- ☐ d.

```
public LinkedList<Integer> pesquisar(EstatisticaDeComparacoes estatistica,
    T elemento, T... elementos) {
    LinkedList<Integer> indices = new LinkedList<>();
    for (int i = 0; i < elementos.length; i++) {
        estatistica.incrementarComparacoes();
        if (criterio.comparar(elemento, elementos[i]) == 0) {
            indices.add(i);
            while (++i < elementos.length && criterio.comparar(elemento, elementos[i]) == 0) {
                indices.add(i);
            }
        }
    }
    return indices;
}
```

PERGUNTA DO AUTOMÓVEL!!

Pergunta 3

Correto

Nota: 5.00 em
5.00

🚩 Marcar
pergunta

Considere a classe Automovel e o critério de comparação descritos de seguida.

```
public class Automovel {
    private String marca;
    private int cilindrada;
    private float consumo;

    public Automovel(String marca, int cilindrada, float consumo) {
        this.marca = marca;
        this.cilindrada = cilindrada;
        this.consumo = consumo;
    }

    public String getMarca() {
        return marca;
    }

    public int getCilindrada() {
        return cilindrada;
    }

    public float getConsumo() {
        return consumo;
    }
}

public enum ComparacaoAutomoveis implements Comparacao<Automovel> {
    CRITERIO;

    @Override
    public int comparar(Automovel o1, Automovel o2) {
        int comparacao = o1.getMarca().compareTo(o2.getMarca());
        if (comparacao != 0) {
            return comparacao;
        }
        comparacao = -Long.compare(o1.getCilindrada(), o2.getCilindrada());
        if (comparacao != 0) {
            return comparacao;
        }
        return Float.compare(o2.getConsumo(), o1.getConsumo());
    }
}
```

O critério de comparação compara automóveis:

Selecione uma opção de resposta:

- ☐ por marca descendente seguido por cilindrada descendente seguido por consumo ascendente
- ☒ por marca ascendente seguido por cilindrada descendente seguido por consumo descendente ✓
- ☐ por marca ascendente seguido por cilindrada ascendente seguido por consumo ascendente
- ☐ por marca ascendente seguido por cilindrada descendente seguido por consumo ascendente

Pergunta 2

Incorreto

Nota: 0.00 em

5.00

🚩 Marcar
pergunta

Considere a classe Automovel e o critério de comparação descritos de seguida.

```
public class Automovel {
    private String marca;
    private int cilindrada;
    private float consumo;

    public Automovel(String marca, int cilindrada, float consumo) {
        this.marca = marca;
        this.cilindrada = cilindrada;
        this.consumo = consumo;
    }

    public String getMarca() {
        return marca;
    }

    public int getCilindrada() {
        return cilindrada;
    }

    public float getConsumo() {
        return consumo;
    }
}

public enum ComparacaoAutomoveis implements Comparacao<Automovel> {
    CRITERIO;

    @Override
    public int comparar(Automovel o1, Automovel o2) {
        int comparacao = o1.getMarca().compareTo(o2.getMarca());
        if (comparacao != 0) {
            return comparacao;
        }
        comparacao = -Long.compare(o2.getCilindrada(), o1.getCilindrada());
        if (comparacao != 0) {
            return comparacao;
        }
        return Float.compare(o1.getConsumo(), o2.getConsumo());
    }
}
```

O critério de comparação compara automóveis:

Selecione uma opção de resposta:

- ☐ por marca descendente seguido por cilindrada descendente seguido por consumo ascendente
- ☐ por marca ascendente seguido por cilindrada descendente seguido por consumo ascendente
- ☐ por marca ascendente seguido por cilindrada descendente seguido por consumo descendente
- ☒ por marca ascendente seguido por cilindrada ascendente seguido por consumo ascendente ✓

Pergunta 2

Correto

Nota: 5,00 em 5,00

Marcar pergunta

Considere a classe Automovel e o critério de comparação descritos de seguida.

```
public class Automovel {
    private String marca;
    private int cilindrada;
    private float consumo;

    public Automovel(String marca, int cilindrada, float consumo) {
        this.marca = marca;
        this.cilindrada = cilindrada;
        this.consumo = consumo;
    }

    public String getMarca() {
        return marca;
    }

    public int getCilindrada() {
        return cilindrada;
    }

    public float getConsumo() {
        return consumo;
    }
}

public enum ComparacaoAutomoveis implements Comparacao<Automovel> {
    CRITERIO;

    @Override
    public int comparar(Automovel o1, Automovel o2) {
        int comparacao = o1.getMarca().compareTo(o2.getMarca());
        if (comparacao != 0) {
            return comparacao;
        }
        comparacao = -Long.compare(o1.getCilindrada(), o2.getCilindrada());
        if (comparacao != 0) {
            return comparacao;
        }
        return Float.compare(o1.getConsumo(), o2.getConsumo());
    }
}
```

O critério de comparação compara automóveis;

Selecione uma opção de resposta:

- ☐ por marca ascendente seguido por cilindrada descendente seguido por consumo descendente
- ☒ por marca ascendente seguido por cilindrada descendente seguido por consumo ascendente ✓
- ☐ por marca descendente seguido por cilindrada descendente seguido por consumo ascendente
- ☐ por marca ascendente seguido por cilindrada ascendente seguido por consumo ascendente

Pergunta 3

Incorreto

Nota: 0.00 em
5.00

Remover
marcação

Considere a classe Automovel e o critério de comparação descritos de seguida.

```
public class Automovel {
    private String marca;
    private int cilindrada;
    private float consumo;

    public Automovel(String marca, int cilindrada, float consumo) {
        this.marca = marca;
        this.cilindrada = cilindrada;
        this.consumo = consumo;
    }

    public String getMarca() {
        return marca;
    }

    public int getCilindrada() {
        return cilindrada;
    }

    public float getConsumo() {
        return consumo;
    }
}

public enum ComparacaoAutomoveis implements Comparacao<Automovel> {
    CRITERIO;

    @Override
    public int comparar(Automovel o1, Automovel o2) {
        int comparacao = o2.getMarca().compareTo(o1.getMarca());
        if (comparacao != 0) {
            return comparacao;
        }
        comparacao = -Long.compare(o1.getCilindrada(), o2.getCilindrada());
        if (comparacao != 0) {
            return comparacao;
        }
        return Float.compare(o1.getConsumo(), o2.getConsumo());
    }
}
```

O critério de comparação compara automóveis:

Selecione uma opção de resposta:

- ☐ por marca ascendente seguido por cilindrada ascendente seguido por consumo ascendente
- ☐ por marca descendente seguido por cilindrada descendente seguido por consumo ascendente
- ☐ por marca descendente seguido por cilindrada descendente seguido por consumo descendente
- ☒ por marca ascendente seguido por cilindrada descendente seguido por consumo ascendente ✗

A opção correta é a B!