

Exame Especial

Professor: Gustavo Henrique Borges Martins

Aluno: _____

1. (35 pontos) Escreva uma classe que contenha:

- Um atributo privado *String frase*.
- Um método estático que receba uma *String* e retire quaisquer caracteres que não sejam letras do alfabeto português, transforme qualquer letra maiúscula em minúscula, e retorne este valor.
- Um método estático que receba uma *String* e retorne verdadeiro se essa *String* é um palíndromo¹.
- Um construtor que receba e inicialize o atributo com uma *String*, depois de ter sido passado no método que retire os caracteres não letras, apenas se esta *String* for um palíndromo.
- Se a palavra passada ao construtor não for um palíndromo, levante uma exceção do tipo **PalindromeException**.
- Um método que receba uma *String* e retorne o número de ocorrências desta *String* no atributo privado.

Crie a classe de exceção **PalindromeException** como filha da classe **Exception**.

¹Um palíndromo é uma sequência de caracteres que pode ser lida de qualquer sentido e permanece igual. Exemplo de palavra palíndroma: "anilina", exemplo de frase palíndroma: "Roma me tem amor."

2. (35 pontos) Transcreva o diagrama abaixo em linguagem Java.

Projeto Posicionamento

Exame Especial 19/07/2022



Implemente o método **Localizacao** da classe **Coordenadas**, sendo que o retorno deve ser dois valores *float*, os ângulos θ , representando a longitude, e ϕ , representando a latitude:

$$\theta = \frac{\pi}{2} - \arccos \left(\frac{z}{\sqrt{x^2 + y^2 + z^2}} \right) \quad (1)$$

$$\phi = \arctan \left(\frac{y}{x} \right) \quad (2)$$

Implemente o método **Verifica** da classe **Tempo**, sendo que este método deve levantar uma exceção do tipo **DateTimeException** se a data² e hora³ informada não for válida.

Implemente o método **Adicionar** da classe **Tempo**, que adiciona o valor da instância atual com a instância passada, respeitando as regras dos segundos, minutos, horas, dias, meses e anos, e retorne este valor em uma nova instância.

²Uma data válida é uma data que pertence ao calendário Gregoriano. Isso significa que a validação do dia, do mês e do ano devem ser feitas. Exemplo de data inválida: 32/13/-0010.

³Os valores de horas devem estar entre 0h e 23h, os valores de minutos devem estar entre 0min e 59min, e os valores de segundos devem estar entre 0s e 59s

3. (30 pontos) Analise o seguinte trecho de código:

```
1 public interface Comparavel {
2     public int comparacao (Comparavel valor);
3 }
4 public class Ordena {
5     private static long trocas, comparacoes;
6     private static void trocar (Comparavel[] vetor, int a, int b) {
7         trocas++;
8         Comparavel temp = vetor[a];
9         vetor[a] = vetor[b];
10        vetor[b] = temp;
11    }
12    private static int QSparticao(Comparavel[] vetor, int e, int d) {
13        Comparavel valor = vetor[d];
14        int i = (e - 1);
15        for (int j = e; j < d; j++) {
16            comparacoes++;
17            if (valor.comparacao(vetor[j]) == -1)
18                trocar(vetor, ++i, j);
19        }
20        trocar(vetor, ++i, d);
21        return i;
22    }
23    public static void QSIterativo (Comparavel[] vetor) {
24        comparacoes = 0;
25        trocas = 0;
26        int esquerda = 0, direita = vetor.length - 1;
27        int[] pilha = new int[vetor.length];
28        int top = -1;
29        pilha[++top] = esquerda;
30        pilha[++top] = direita;
31        while (top >= 0) {
32            direita = pilha[top--];
33            esquerda = pilha[top--];
34            int p = QSparticao(vetor, esquerda, direita);
35            if (p - 1 > esquerda) {
36                pilha[++top] = esquerda;
37                pilha[++top] = p - 1;
38            }
39            if (p + 1 < direita) {
40                pilha[++top] = p + 1;
41                pilha[++top] = direita;
42            }
43        }
44    }
45    public static void QSRecursoivo (Comparavel[] vetor) {
46        comparacoes = 0;
47        trocas = 0;
48        QSRecursoivo(vetor, 0, vetor.length);
49    }
50    private static void QSRecursoivo (Comparavel[] vetor, int esquerda, int direita)
51    {
52        if (esquerda >= direita)
53            return;
```

```

53         int p = QSParticao(vetor, esquerda, direita);
54         QSRecurso(vetor, esquerda, p-1);
55         QSRecurso(vetor, p+1, direita);
56     }
57     public static void contagem () {
58         System.out.println ("Última ordenação: Trocas: " + trocas + " Comparações: "
59         + comparacoes);
60     }
61 }
62 public class Real implements Comparavel {
63     public double r;
64     public Real (double a) {
65         r = a;
66     }
67     public int comparacao (Comparavel valor) {
68         Real comparado = (Real) valor;
69         if (r == comparado.r)
70             return 0;
71         else if (r < comparado.r)
72             return 1;
73         else
74             return -1;
75     }
76 }

```

Considerando um vetor da classe **Real**, em que o conteúdo dos atributos *r* tenha os seguintes elementos, na ordem: [9, 4, 10, 7, 5, 3, 2]. Usando este vetor como parâmetro para os métodos **QSIterativo** e **QSRecurso**, responda:

- Escreva o valor das variáveis nas iterações do método **QSIterativo**.
- Quantas comparações o método **QSIterativo** realiza?
- Quantas trocas o método **QSIterativo** realiza?
- Escreva o valor das variáveis nas iterações do método **QSRecurso**.
- Quantas comparações o método **QSRecurso** realiza?
- Quantas trocas o método **QSRecurso** realiza?

Escreva uma classe filha de **Comparavel** que contenha um atributo público *String*, e que seja capaz de organizar alfabeticamente um vetor desta classe usando os métodos da classe **Ordena**.

Questões	1	2	3	Total
Total de pontos	35	35	30	100
Pontos obtidos				