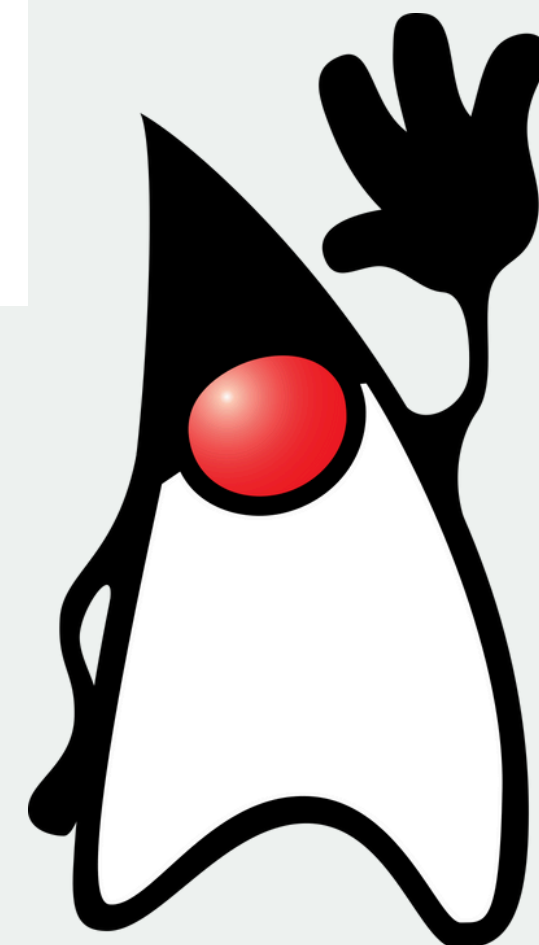
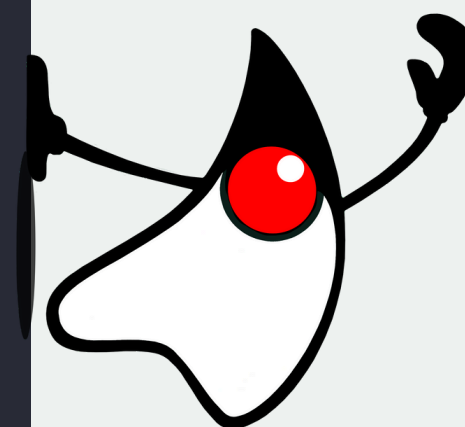


Introdução ao Java e a POO

AEDS II - 2/2024



CONTENT

- 01 POO
- 02 ENCAPSULAMENTO
- 03 HERANÇA
- 04 POLIMORFISMO
- 05 LINGUAGEM JAVA
- 06 EXERCÍCIOS

Programação Orientada a Objetos (POO)

- POO é um paradigma
- Objetos são entidades que encapsulam dados e comportamentos
- Entidades representam abstrações do mundo real ou conceitos específicos de uma problema

```
class Livro {  
    String titulo;  
    String autor;  
    int anoPublicacao;  
  
    void emprestar() {  
        System.out.println("Livro emprestado.");  
    }  
  
    void devolver() {  
        System.out.println("Livro devolvido.");  
    }  
}
```

```
Item item1 = new Item("Pizza", 29.90);  
Item item2 = new Item("Refrigerante", 5.50);
```

```
class Item {  
    String nome;  
    double preco;  
  
    Item(String nome, double preco) {  
        this.nome = nome;  
        this.preco = preco;  
    }  
}
```

class



objects



Encapsulation



Polymorphism



Inheritance




Abstraction



Pilares do
POO

Encapsulamento

- É a prática de "esconder" os detalhes das implementações das classes
- Restringir o acesso direto ao dados
 Proteger a integridade dos dados
- Buscar privar os atributos e publicar os métodos
- Tipos de controladores de acesso

```
public class Pessoa {  
    private String nome; // Atributo encapsulado  
    private int idade;   // Atributo encapsulado  
  
    // Getter para o nome  
    public String getNome() {  
        return nome;  
    }  
  
    // Setter para o nome  
    public void setNome(String nome) {  
        this.nome = nome;  
    }  
  
    // Getter para a idade  
    public int getIdade() {  
        return idade;  
    }  
  
    // Setter para a idade  
    public void setIdade(int idade) {  
        if (idade >= 0) { // Controle adicional para garantir validade  
            this.idade = idade;  
        }  
    }  
}
```

Herança

- Relação “é um”
- Uma classe consegue herdar **todos** os dados e comportamentos de outra

- Vantagens:
 - Reuso
 - Polimorfismo

- Superclasse
(+ genérica)

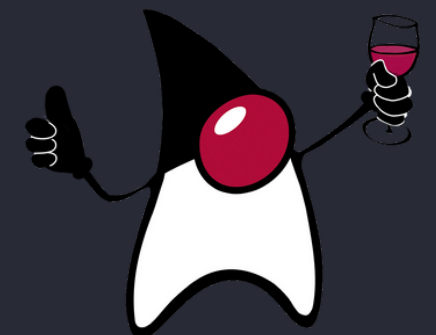
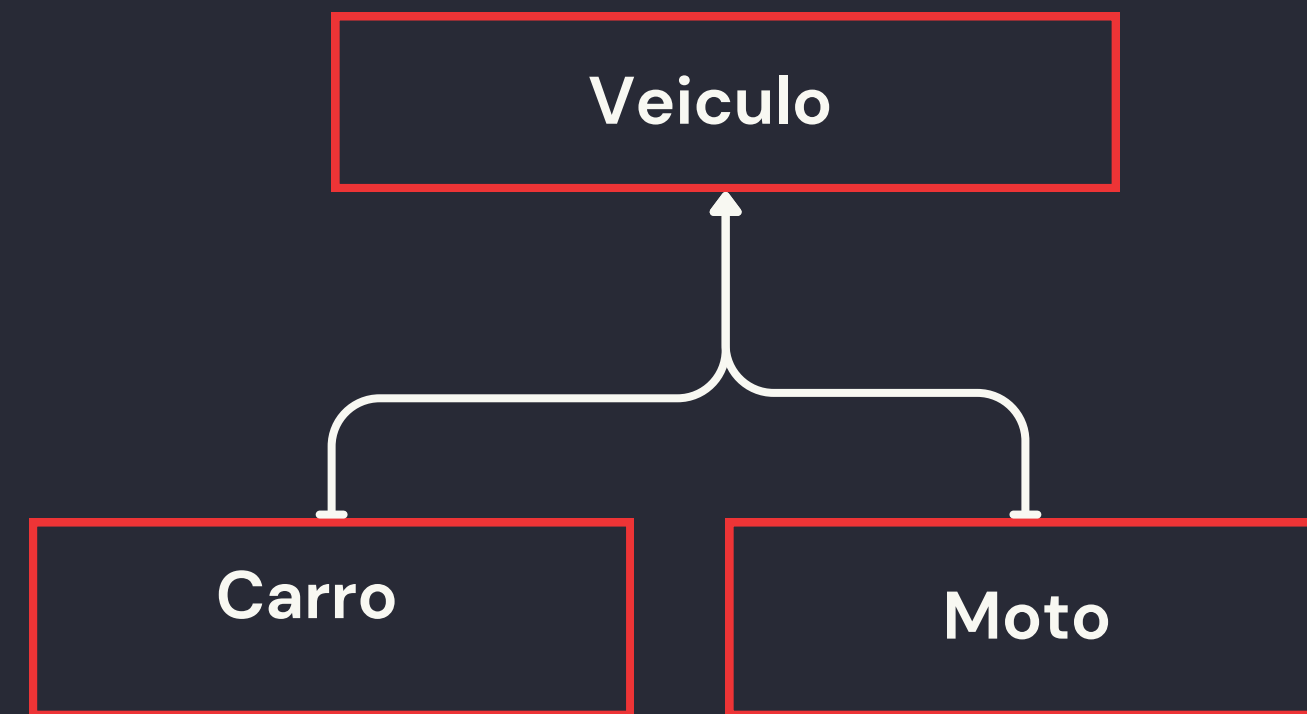
- Subclasse
(+ específica)

```
// Classe base ou superclasse
public class Veiculo {
    private String marca;
    private int ano;

    public Veiculo(String marca, int ano) {
        this.marca = marca;
        this.ano = ano;
    }

    public void buzinar() {
        System.out.println("Buzinando!");
    }

    public void exibirInfo() {
        System.out.println("Marca: " + marca + ", Ano: " + ano);
    }
}
```



Herança

```
// Subclasse de Veiculo
public class Carro extends Veiculo {
    private int portas;

    public Carro(String marca, int ano, int portas) {
        super(marca, ano); // Chama o construtor da superclasse
        this.portas = portas;
    }

    public void exibirInfo() {
        super.exibirInfo(); // Chama o método da superclasse
        System.out.println("Número de portas: " + portas);
    }
}
```

```
// Subclasse de Veiculo
public class Moto extends Veiculo {
    private boolean temSissyBar;

    public Moto(String marca, int ano, boolean temSissyBar) {
        super(marca, ano); // Chama o construtor da superclasse Veiculo
        this.temSissyBar = temSissyBar;
    }

    public void exibirInfo() {
        super.exibirInfo(); // Chama o método da superclasse
        System.out.println("Tem Sissy Bar: " + (temSissyBar ? "Sim" : "Não"));
    }
}
```

```
public static void main(String[] args) {
    Carro meuCarro = new Carro("Fiat", 2021, 4);
    Moto minhaMoto = new Moto("Harley-Davidson", 2019, true);
}
```


Polimorfismo (Universal de inclusão)

- Variáveis de um **mesmo tipo mais genérico** possam **apontar** para **objetos de tipos específicos diferentes**, tendo **comportamentos diferentes** conforme cada tipo específico.

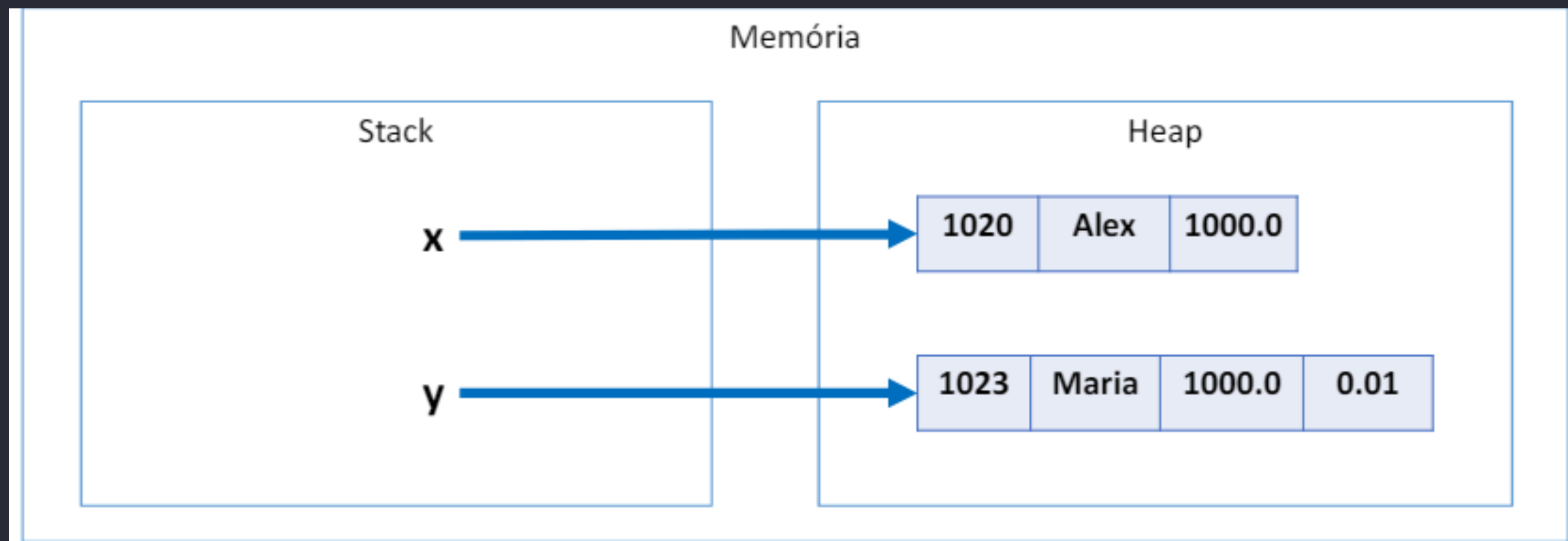
```
Account x = new Account(1020, "Alex", 1000.0);  
Account y = new SavingsAccount(1023, "Maria", 1000.0, 0.01);  
  
x.withdraw(50.0);  
y.withdraw(50.0);
```

Account:

```
public void withdraw(double amount) {  
    balance -= amount + 5.0;  
}
```

SavingsAccount:

```
@Override  
public void withdraw(double amount) {  
    balance -= amount;  
}
```



Principais Funções em Java

1- Equals()

Descrição: Compara o conteúdo de dois objetos para verificar se são iguais, retorna um boolean

Sintaxe: obj1.equals(obj2)

2- split()

Descrição: Divide uma string em partes com base em um delimitador.

Sintaxe: string.split(",")

3- substring()

Descrição: Extrai uma parte da string entre dois índices.

Sintaxe: string.substring(inicio, fim)

4- toUpperCase() e toLowerCase()

Descrição: Converte todos os caracteres da string para maiúsculas ou minúsculas.

Sintaxe:

string.toUpperCase()

string.toLowerCase()

5- equalsIgnoreCase()

Descrição: Compara duas strings, ignorando as diferenças entre maiúsculas e minúsculas.

Sintaxe: string1.equalsIgnoreCase(string2)

6- StringBuilder

Principais Métodos

1. **append()**

- Adiciona uma string/caracter ao final.
- `StringBuilder sb = new StringBuilder();`
- `sb.append("Hello");`
- `sb.append(" World"); // "Hello World"`

2. **insert()**

- Insere uma string/caracter em uma posição específica.
- `StringBuilder sb = new StringBuilder();`
- `sb.append("Hello");`
- `sb.insert(5, " World"); // "Hello World"`

3. **delete()**

- Remove uma subsequência de caracteres.
- `StringBuilder sb = new StringBuilder();`
- `sb.append("Hello World");`
- `sb.delete(5, 11); // "Hello"`

Exercícios

Pomekon Collection

Número questão no Becrownd: 2174



Alien Volwels

Número questão no Becrownd: 2150





Coleção de Pomekon

Por Gabriel Duarte, UNIFESO  Brazil

Timelimit: 1

Desde que foi lançado oficialmente o Pomekon no Brasil, Dabriel está tentando realizar seu maior sonho: Ser um Mestre Pomekon. Sua meta é conquistar os 151 Pomekons disponíveis. Ele já conseguiu capturar muitos monstrixinhos, porém em sua cidade aparecem muitos Pomekons repetidos, fazendo com que ele capture diversas vezes o mesmo Pomekon.

Vendo que sua mochila está bem cheia, Dabriel pediu para que você fizesse um programa de computador que informasse a ele quantos Pomekons faltam para completar a coleção.

Entrada

A primeira linha do caso de teste consiste de um inteiro **N** ($1 \leq N \leq 10^3$), representando a quantidade de Pomekons que Dabriel já capturou. As próximas **N** linhas consistem de uma string **S** ($1 \leq |S| \leq 10^3$), representando o nome de cada Pomekons. O nome de cada Pomekons consiste apenas de letras maiúsculas e minúsculas.

Saída

Você deverá imprimir: "Falta(m) **X** pomekon(s).", onde **X** representa a quantidade Pomekons não capturados.

Exemplos de Entrada	Exemplos de Saída
<pre>7 Charmander Caterpie Pidgeot Rattata Zubat Zubat Zubat</pre>	<pre>Falta(m) 146 pomekon(s) .</pre>

Vogais Alienígenas

Por Ricardo Martins, IFSULDEMINAS  Brazil

Timelimit: 1

Desde o Gerador de Improbabilidade Infinita, muitos nem questionam sobre a vida em outros planetas, e se aprofundam em questionamentos mais improváveis, como, por exemplo, será que seres de outros planetas usam os mesmos caracteres que a gente para escrever? E se isto for verdade, será que usam as mesmas vogais que a gente? Pensando nisto, muitos cientistas projetaram vários tipos de alfabetos alienígenas, usando as letras do nosso alfabeto, além dos dígitos, de 0 a 9, com as suas respectivas vogais. Baseados nisto, estes pedem a sua ajuda para identificar vogais em alfabetos alienígenas e fazer contagens a respeito.

Escreva um programa que, dado uma sequência de vogais, em um determinado alfabeto alienígena, contabilize, em um texto escrito com o mesmo alfabeto, quantas vogais o mesmo possui.

Entrada

Haverá diversos casos de teste. Cada caso de teste é formado por duas linhas. A primeira linha informa uma palavra, formada por todas as vogais alienígenas de um determinado planeta. A segunda linha contém uma frase formada por letras do mesmo alfabeto. A entrada termina com fim de arquivo.

Saída

Para cada caso de teste, imprima a quantidade de vogais alienígenas correspondente.

Exemplo de Entrada

```
aeiou
o rato roeu a roupa do rei de roma
4310
t3st3 p4r4 c0dlflc4r
kwy
the quick brown fox jumps over the lazy dog
```

Exemplo de Saída

```
16
8
3
```