VAMOS RELEMBRAR!!!!

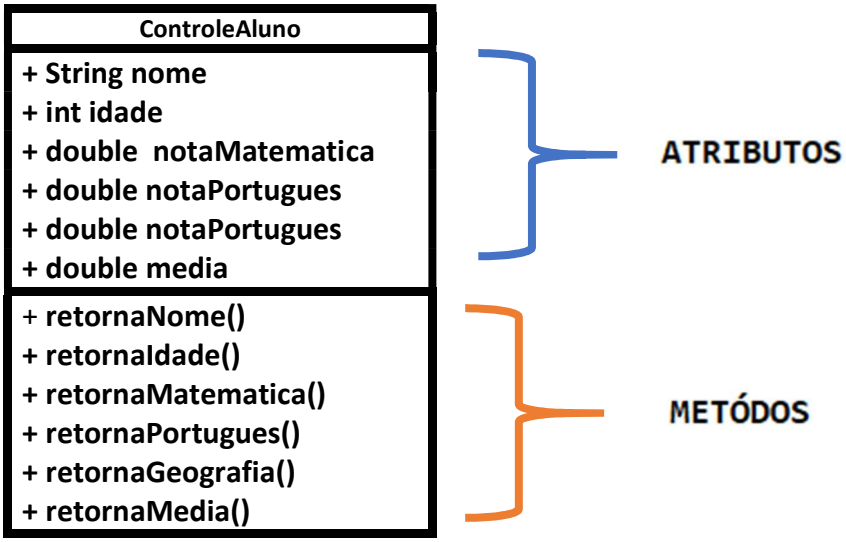
A orientação a objetos (OO) é um **paradigma** de programação que organiza o software em torno de "objetos", que são **instâncias de "classes".** Esses objetos **encapsulam dados (atributos) e comportamentos (métodos),** permitindo uma modelagem mais próxima do mundo real.

Paradigma é descrito nos dicionários como um substantivo masculino que significa exemplo geral, conjunto de formas ou modelo de algo. Na linguagem coloquial é constantemente utilizada como sinônimo de padrão, exemplo, modelo, norma, protótipo e regra.

Instanciar uma classe é criar um novo objeto do mesmo tipo dessa classe. Uma classe somente poderá ser utilizada após ser instanciada.

Vamos meter a mão na massa e vamos criar uma classe de controle de aluno que vai ter como dados o nome, idade, nota de Matemática, Português e Geografia, deverá também calcular a média geral do aluno. Vamos dar uma olhada no Diagrama UML Java abaixo. A classe que iremos ciar se chamará ”ControleAluno”.

Um diagrama **UML (Unified Modeling Language -- Linguagem de Modelagem Unificada)** em Java é uma representação visual da estrutura, comportamento e interações de um sistema orientado a objetos. Ele é amplamente utilizado para modelar sistemas de software, incluindo aqueles desenvolvidos em Java, ajudando a planejar, documentar e comunicar o design do sistema de forma clara e padronizada.



VAMOS CODAR !!!!

Classe ControleAluno

```
public class ControleAluno {
    String nome;
    int idade;
    double notaMatematica;
    double notaPortugues;
    double notaGeografia;
    double media;

    public void  retornaNome(){
        System.out.println("Nome do Aluno : " + nome);
    }

    public void retornaIdade(){
        System.out.println("Idade Aluno: "+ idade);
    }

    public void retornaMatematica(){
        System.out.println("Nota de Matemática: "+ notaMatematica);
    }
    public void retornaPortugues(){
        System.out.println("Nota de Português: "+ notaPortugues);
    }
    public void retornaGeografia(){
        System.out.println("Nota de Geografia: "+ notaGeografia);
    }

    public void retornaMedia(){
        media = (notaMatematica + notaPortugues + notaGeografia)/3;
        System.out.println("Média Geral: "+ media);
    }
}
```

ATRIBUTOS

METÓDOS

Agora vamos coda a classe **CadastraAluno.**

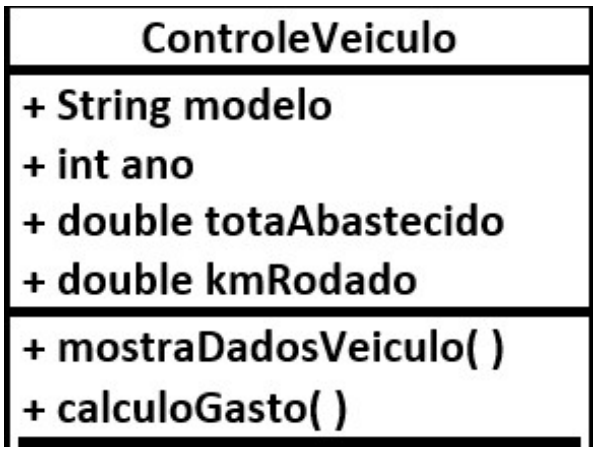
```
public class CadastraAluno {
    public static void main(String[] args) {
        Scanner leia = new Scanner(System.in);
        ControleAluno aluno1 = new ControleAluno();
        System.out.print("Digite nome do Aluno: ");
        aluno1.nome = leia.nextLine();
        System.out.print("Digite idade do aluno: ");
        aluno1.idade = leia.nextInt();
        System.out.print("Digite nota de Matemática: ");
        aluno1.notaMatematica = leia.nextDouble();
        System.out.print("Digite nota de Português: ");
        aluno1.notaPortugues = leia.nextDouble();
        System.out.print("Digite nota de Geografia: ");
        aluno1.notaGeografia = leia.nextDouble();
        aluno1.retornaNome();
        aluno1.retornaIdade();
        aluno1.retornaMatematica();
        aluno1.retornaPortugues();
        aluno1.retornaGeografia();
        aluno1.retornaMedia();
    }
}
```

Se tudo ocorrer certinho é a tela abaixo que deve aparecer.

```
Digite nome do Aluno: Joao da Silva
Digite idade do aluno: 16
Digite nota de Matemática: 5,5
Digite nota de Português: 6,5
Digite nota de Geografia: 7,5
Nome do Aluno : Joao da Silva
Idade Aluno: 16
Nota de Matemática: 5.5
Nota de Português: 6.5
Nota de Geografia: 7.5
Média Geral: 6.5
```

Exercicio de Fixação

Veja a imagem abaixo do modelo UML Java e o programa funcionando e crie os códigos necessários para execução do foi solicitado.



Atenção

O mostraDadosVeiculo( ) → Deverá mostrar os dados do veículo, modelo ,ano do veículo, total abastecido, total.

calculoGasto( ) → O resultado do gasto de combustível .

Agora você deve criar a classe **CadastraVeiculo,** onde vai entrar com os dados e fazer o funcionamento do programa. Veja a imagem do programa funcionando.

```
Digite Modelo do veiculo: Fiat Uno
Digite Ano do veiculo: 2001
Total Qtde de combustivel: 45
Total rodado : 855
Modelo do veiculo: Fiat Uno
Ano do veiculo: 2001
Total Abastecimento: 45.0
Kilometro Rodado: 855.0
O veículo faz  19.0 Km/Litro
```

O encapsulamento é um dos pilares da Programação Orientada a Objetos (POO) e refere-se ao conceito de esconder os detalhes internos de um objeto e expor apenas uma interface controlada para interação com ele. Em outras palavras, o encapsulamento protege os dados de um objeto, garantindo que eles só possam ser acessados ou modificados por meio de métodos específicos.

Objetivo do Encapsulamento:

Proteção dos Dados:

Impede que os dados internos de um objeto sejam acessados ou modificados diretamente, evitando inconsistências ou corrupção do estado do objeto.

Controle de Acesso:

Permite definir regras claras sobre como os dados podem ser acessados ou alterados, garantindo que todas as operações sejam validadas.

Facilidade de Manutenção:

Ao esconder os detalhes internos, o encapsulamento permite que você altere a implementação de uma classe sem afetar outras partes do código que dependem dela.

Abstração:

Expõe apenas o que é necessário para o uso do objeto, simplificando a interação com ele.

Como o Encapsulamento é Implementado:

Em linguagens como Java, C#, Python e outras que suportam POO, o encapsulamento é implementado usando:

Modificadores de Acesso:

private: O atributo ou método só pode ser acessado dentro da própria classe.

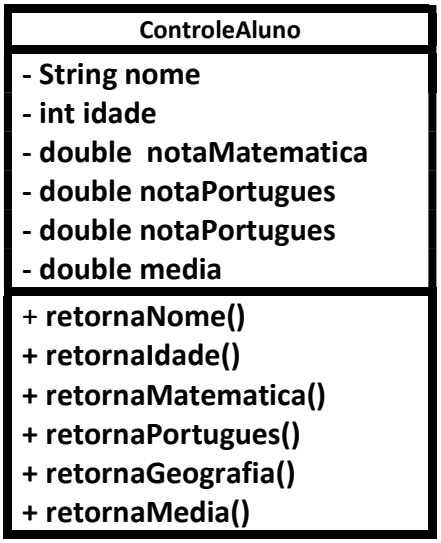
protected: O atributo ou método pode ser acessado dentro da própria classe e por classes filhas (herança).

public: O atributo ou método pode ser acessado de qualquer lugar.

Getters e Setters:

Getters: Métodos públicos que permitem ler o valor de um atributo privado.

Setters: Métodos públicos que permitem modificar o valor de um atributo privado, geralmente com validações.



Observe o modelo UML note que agora nos atributos temos o sinal negativo, isto significa que os atributos são qualificadas como privadas (private) , temos que resolver como acessar este métodos.

```
public class ControleAluno {
    String nome;
    private int idade;
    private double notaMatematica;
    private double notaPortugues;
    private double notaGeografia;
    private double media;

    public String getNome() {
        return nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }

    public int getIdade() {
        return idade;
    }

    public void setIdade(int idade) {
        this.idade = idade;
    }

    public double getNotaMatematica() {
        return notaMatematica;
    }

    public void setNotaMatematica(double notaMatematica) {
        this.notaMatematica = notaMatematica;
    }

    public double getNotaPortugues() {
        return notaPortugues;
    }


  public void setNotaGeografia(double notaGeografia) {
      this.notaGeografia = notaGeografia;
  }

  public double getMedia() {
      return media;
  }

  public void setMedia(double media) {
      this.media = media;
  }

  public void  retornaNome(){
      System.out.println("Nome do Aluno : " + nome);
  }

  public void retornaIdade(){
      System.out.println("Idade Aluno: "+idade);
  }

  public void retornaMatematica(){
      System.out.println("Nota de Matemática: "+ notaMatematica);
  }
  public void retornaPortugues(){
      System.out.println("Nota de Português: "+ notaPortugues);
  }
  public void retornaGeografia(){
      System.out.println("Nota de Geografia: "+ notaGeografia);
  }

  public void retornaMedia(){
      media = (notaMatematica + notaPortugues + notaGeografia)/3;
      System.out.println("Média Geral: "+ media);
  }

}
```

```java
public class CadastraAluno {
    public static void main(String[] args) {
        Scanner leia = new Scanner(System.in);
        ControleAluno aluno1 = new ControleAluno();
        System.out.print("Digite nome do Aluno: ");
        String nome = leia.nextLine();
        aluno1.setNome(nome);

        System.out.print("Digite idade do aluno: ");
        int idade = leia.nextInt();
        aluno1.setIdade(idade);

        System.out.print("Digite nota de Matemática: ");
        double notamatematica = leia.nextDouble();
        aluno1.setNotaMatematica(notamatematica);

        System.out.print("Digite nota de Português: ");
        double notaportugues = leia.nextDouble();
        aluno1.setNotaPortugues(notaportugues);

        System.out.print("Digite nota de Geografia: ");
        double notageografia = leia.nextDouble();
        aluno1.setNotaGeografia(notageografia);

        aluno1.retornaNome();
        aluno1.retornaIdade();
        aluno1.retornaMatematica();
        aluno1.retornaPortugues();
        aluno1.retornaGeografia();
        aluno1.retornaMedia();
    }
}
```

Exercicio Fixação

ControleConta
- String nome - int idade - double saldoInicial - double deposito - double saque - double saldoFinal
+ mostrarDados( ) + depositar( ) + sacar( ) + mostrarSaldo ( )

Você deve criar uma classe (MovimentaConta) que receba o nome do cliente, sua idade, um deposito pode ter o valor zerado, um saque pode ter o valor zerado, depois deve mostrar o saldo da conta do cliente.

ATENÇÃO VAMOS PARTIR QUE O SALDO INICIAL SERA DE r$ 1000,00