A partir da classe **EGrau2** e do código **Main** que você forneceu anteriormente, podemos destacar os principais conceitos e estruturas da **Programação Orientada a Objetos (POO)**:

**1. Classe:**

* **Classe** é uma **estrutura básica** da POO que define um **molde** ou um **modelo** para criar objetos. A classe encapsula dados e comportamentos (atributos e métodos).
* No caso, a classe **EGrau2** representa uma equação do segundo grau e contém os atributos a, b e c (coeficientes da equação), além dos métodos para calcular o delta e as raízes.

|  |
| --- |
| public class EGrau2 {  public double a, b, c;  ...  } |

**2. Instância e Objeto:**

* Um **objeto** é uma **instância** de uma classe, ou seja, uma **representação concreta** de uma classe em memória.
* No código Main, quando criamos um novo objeto da classe **EGrau2**, isso é feito por meio do comando:

|  |
| --- |
| EGrau2 resultado = new EGrau2(a, b, c); |

* Aqui, **resultado** é a **instância** ou **objeto** da classe EGrau2, que contém os coeficientes da equação do segundo grau e pode acessar seus métodos e atributos.

**3. Atributos:**

* Os **atributos** (ou **variáveis de instância**) são as **propriedades** de um objeto. Cada objeto criado a partir da classe tem suas próprias cópias desses atributos.
* Em **EGrau2**, os atributos são os coeficientes a, b, e c da equação:

|  |
| --- |
| public double a, b, c; |

**4. Construtor:**

* O **construtor** é um método especial que tem o mesmo nome da classe e é utilizado para **inicializar os atributos** de um objeto quando ele é criado.
* No código, o construtor recebe os coeficientes a, b, e c como parâmetros e os **atribui** aos atributos da classe:

|  |
| --- |
| public EGrau2(double a, double b, double c){  this.a = a;  this.b = b;  this.c = c;  } |

* O uso de this.a = a diferencia o **atributo** a da classe do **parâmetro** a do método.

**5. Métodos:**

* **Métodos** são as **funções** ou **comportamentos** que os objetos da classe podem realizar. Eles podem acessar e modificar os atributos do objeto e realizar ações.
* Na classe **EGrau2**, temos os seguintes métodos:
  + **delta()**: Calcula o valor do delta (discriminante) da equação do segundo grau.

|  |
| --- |
| public double delta(){  return (b \* b) - (4 \* a \* c);  } |

* + **calculax1(double delta)**: Calcula a primeira raiz x1 da equação usando a fórmula de Bhaskara.

|  |
| --- |
| public double calculax1(double delta){  return (-b + Math.sqrt(delta)) / (2 \* a);  } |

* + **calculax2(double delta)**: Calcula a segunda raiz x2 da equação usando a fórmula de Bhaskara.

|  |
| --- |
| public double calculax2(double delta){  return (-b - Math.sqrt(delta)) / (2 \* a);  } |

**6. Encapsulamento:**

* **Encapsulamento** é o conceito de **esconder os detalhes internos** de implementação de uma classe e fornecer uma interface clara para interagir com os objetos.
* Os atributos **a**, **b**, e **c** são **diretamente acessíveis** porque foram declarados como **public**, mas o cálculo das raízes e do delta está **encapsulado** dentro dos métodos **delta()**, **calculax1()**, e **calculax2()**. O programa Main só precisa chamar esses métodos para obter o resultado, sem saber como exatamente o cálculo é feito.

**7. Parâmetros:**

* Parâmetros são os **valores** que você passa para os métodos ou construtores para **personalizar o comportamento** de um objeto ou função.
* O construtor **EGrau2(double a, double b, double c)** usa os parâmetros a, b, e c para inicializar os coeficientes do objeto.
* Os métodos **calculax1(double delta)** e **calculax2(double delta)** usam o parâmetro delta (calculado previamente pelo método delta()) para calcular as raízes.

**8. Reuso de Código:**

* O método **delta()** calcula o valor de delta apenas uma vez e o resultado é reutilizado pelos métodos **calculax1()** e **calculax2()**, evitando duplicação de código.

**Exemplo de Aplicação no Main:**

Quando o programa principal (**Main**) é executado:

1. **Instancia-se** um objeto da classe **EGrau2**:

|  |
| --- |
| EGrau2 resultado = new EGrau2(a, b, c); |

* + O objeto resultado agora contém os coeficientes a, b, e c da equação que foi fornecida pelo usuário.

1. O método **delta()** é chamado para calcular o delta:

|  |
| --- |
| double deltaValido = resultado.delta(); |

1. Dependendo do valor de delta, as raízes são calculadas:

|  |
| --- |
| double raiz1 = resultado.calculax1(deltaValido);  double raiz2 = resultado.calculax2(deltaValido); |

**Conclusão:**

Este exemplo exemplifica claramente os principais conceitos da **Programação Orientada a Objetos (POO)**:

* **Classes** modelam entidades do mundo real (neste caso, uma equação do segundo grau).
* **Objetos** são instâncias dessas classes que possuem estado (dados) e comportamentos (métodos).
* O **encapsulamento** protege os detalhes de implementação.
* O **construtor** inicializa objetos.
* **Métodos** são usados para realizar ações específicas, utilizando e manipulando os dados encapsulados no objeto.
* O uso de **parâmetros** permite criar objetos personalizados e reusar métodos com diferentes valores.