ENCAPSULAMENTO

PROF. ME.
HÉLIO
ESPERIDIÃO

REVIEW – CLASSES E INSTÂNCIAS.

```
public class Pessoa {
    public int idade;
    public String nome;
    public void fazerAniversario(){
        this.idade++;
    }
    public static void main(String args[]){
        Pessoa p =new Pessoa();
        p.nome="Hélio";
        p.idade=31;
        p.fazerAniversario();
        System.out.println(p.idade);


        Pessoa p2 =new Pessoa();
        p2.nome="Patricia";
        p2.idade=17;
        p2.fazerAniversario();
        System.out.println(p2.idade);
    }

}
```

REVIEW - O OPERADOR THIS

This faz referencia a métodos e atributos da própria classe

```
public class Pessoa {
    public int idade;
    public String nome;
    public void fazerAniversario(){
        this.idade++;
    }
    public static void main(String args[]){
        Pessoa p =new Pessoa();
        p.nome="Hélio";
        p.idade=31;
        p.fazerAniversario();
        System.out.println(p.idade);
    }

}
```

REVIEW - EXEMPLO DE CONSTRUTOR

Método Construtor

```
public class Pessoa {
    public int idade;
    public String nome;

    public void fazerAniversario(){
        this.idade++;
    }

    public Pessoa(String nome,int idade){
        this.nome=nome;
    }
    public static void main(String args[]){
        Pessoa p =new Pessoa("Hélio",31);
        System.out.println(p.nome);
        System.out.println(p.idade);

    }

}
```

Parâmetros do método construtor.

O operador this
faz referência a própria classe ou seja ao atributo nome e não ao parâmetro nome

Instância da classe pessoa, passa como parâmetros para o construtor o nome e a idade

O CONSTRUTOR OBRIGA

- Veja que existe apenas um construtor na classe e ele exige que sejam passados dois parâmetros(nome e idade). Caso os parâmetros não sejam passados é gerado erro de compilação

```
public class Pessoa {
    public int idade;
    public String nome;

    public void fazerAniversario(){
        this.idade++;
    }

    public Pessoa(String nome,int idade){
        this.nome=nome;
    }
    public static void main(String args[]){
        Pessoa p =new Pessoa();
        System.out.println(p.nome);
        System.out.println(p.idade);

    }

}
```

OBJETO/CLASSE COMO ATRIBUTO

```java
public class Pessoa {
    public int idade;
    public String nome;
    public Pessoa mae;
    public void fazerAniversario(){
        this.idade++;
    }

    public Pessoa(String nome,int idade){
        this.nome=nome;
        this.idade=idade;
    }
    public static void main(String args[]){
        Pessoa p =new Pessoa("Ana",31);
        p.mae = new Pessoa("Mariana",65);
        p.mae.mae=new Pessoa("Geovana",90);

        System.out.println(p.nome);
        System.out.println(p.idade);

        System.out.println(p.mae.nome);
        System.out.println(p.mae.idade);

         System.out.println(p.mae.mae.nome);
    }
}
```

ENCAPSULAMENTO

Encapsulamento que em programação orientada a objetos significa separar o programa em partes, o mais isoladas possível.

A ideia é tornar o software mais flexível, fácil de modificar e de criar novas implementações.

Objetos restringem a visibilidade de seus recursos (atributos e métodos) aos outros usuários.

Todo objeto tem uma interface, que determina como os outros objetos podem interagir com ele.

A implementação do objeto é encapsulada, isso é, invisível ou visível fora do próprio objeto

POR QUE CONTROLAR O ACESSO?

- Proteger informação privada.
- Esclarecer como outros programadores devem usar sua classe.
- Manter a implementação separado da interface.

MODIFICADORES

- Usado para modificar o modo como são declaradas as classes, métodos e variáveis.
- Existem três modificadores de acesso e um padrão.
- Toda classe, método e variáveis de instância que declara tem um controle de acesso.

MEDIADORES DE ACESSO

01

Public – deixa visível a classe ou membro para todas as outras classes, subclasses e pacotes do projeto .

02

Private – deixa visível o atributo apenas para a classe em que este atributo se encontra.

03

Protected – deixa visível o atributo para todas as outras classes e subclasses que pertencem ao mesmo pacote.

PUBLICO VS PRIVATE

- Publico todas as classes podem usar os métodos e campos
- .
- Privado apenas a classe podem usar os métodos e campos

MODIFICADORES

Modificador	Classe	Pacote	Globalmente
Public	sim	sim	sim
Protected	sim	sim	não
Sem Modificador	sim	sim	não
Private	sim	não	não

MODIFICADORES EM CLASSES

- Uma classe pode ser declarada apenas com acesso **public** ou **default**.

- O acesso public, permite que todas as classes tenha acesso a esta classe, para acessar uma classe public que esteja em outro pacote precisamos usar o import.

MODIFICADOR DE ACESSO DE MÉTODOS E VARIÁVEIS

- Modificadores de acesso public.
- Modificador de acesso private
- Modificador de acesso protected e default

EXEMPLO

- Todos os atributos e métodos da classe são públicos.
- Todos os atributos e métodos da classe podem ser utilizados em qualquer lugar

```java
package poo;
public class Pessoa {
    public int idade;
    public String nome;
    public Pessoa mae;
    public void fazerAniversario(){
        this.idade++;
    }

}
```

Instância da classe "Pessoa"
Dentro da classe "NomeClasse"

```java
package poo;

public class NomeClasse {
 public static void main(String[] args){
        Pessoa p =new Pessoa();
        p.idade=18;
        System.out.println(p.idade);
 }
}
```

ATRIBUTOS PRIVADOS

- Verifique que o atributo idade **não pode** ser acessado da classe "NomeClasse"

```
package poo;
public class Pessoa {
    private int idade;
    private String nome;
    private Pessoa mae;
    private void fazerAniversario(){
        this.idade++;
    }

}
```

Instância da classe "Pessoa"
Dentro da classe "NomeClasse"

```
package poo;

public class NomeClasse {
 public static void main(String[] args){
        Pessoa p =new Pessoa();
        p.idade=18;
        System.out.println(p.idade);
 }
}
```

ENCAPSULANDO DADOS

- É comum em programação orientada a objetos esconder o máximo possível da implementação das classes, ou seja, os atributos e métodos tem que ser privadados.
- Para acessar atributos e métodos privados e necessário que exista dois métodos associados a cada atributos.
- Método **get** Recuperar o valor de um atributo.
- Método **set** atribui o valor de um atributo.

MÉTODO SET

Os atributos são privados

O nome do método é iniciado com set seguido do nome do atributo com a primeira letra maiúscula

```
public class Pessoa {
    private int idade;
    private String nome;
    private Pessoa mae;
    public void setIdade(int idade) {
        this.idade = idade;
    }
```

O método set é público

Acessa um atributo privado

O método é void pois não retorna nenhum valor, apenas modifica o valor Do atributo idade

MÉTODO GET

Como o atributo idade é um inteiro
o método vai retornar um inteiro

```
private int idade;
private String nome;
private Pessoa mae;
public void setIdade(int idade) {
    this.idade = idade;
}
public int getIdade() {
    return this.idade;
}
```

O nome do método é iniciado
com get seguido do nome do atributo
com a primeira letra maiúscula

Retorna o valor do atributo

COMPLETO

```java
package poo;

public class NomeClasse {
 public static void main(String[] args){
        Pessoa p =new Pessoa();
        p.setIdade(18);
        System.out.println(p.getIdade());
 }
}
```

```java
public class Pessoa {
    private int idade;
    private String nome;
    private Pessoa mae;
    public void setIdade(int idade) {
        this.idade = idade;
    }
    public int getIdade() {
        return this.idade;
    }
    public String getNome() {
        return this.nome;
    }
    public void setNome(String nome) {
        this.nome = nome;
    }
    public Pessoa getMae() {
        return this.mae;
    }
    public void setMae(Pessoa mae) {
        this.mae = mae;
    }
    public void fazerAniversario(){
        this.idade++;
    }

}
```