

## NOVICE PROGRAMMER



## EXPERIENCED PROGRAMMER



# Informações diversas

- Têm vindo a ser publicadas/atualizadas as notas da avaliação contínua no Moodle
- Enunciado do projecto (parte 1) será publicado em breve

# Frequência intermédia

**14 Novembro - 10h00**

Salas serão anunciadas via Moodle

**Importante: Quem chegar atrasado  
não poderá fazer o teste**

Matéria que sai:  
aula teórica 1 até à aula teórica 7  
(hoje)

# Frequência intermédia

A frequência terá um enunciado com 5 secções:

- **Conceitos** - Perguntas de resposta livre ou escolha múltipla sobre os conceitos teóricos que demos nas aulas (ex: Quais os 3 paradigmas de programação de que falámos nas aulas?)
- **Análise de código** - São apresentados excertos de código Java para analisar (ex: qual o output do programa, identifique erros, etc.)
- **Testes unitários** - Preencher a tabela com cenários de teste
- **Estrutura de memória** - Com base num excerto de código Java, desenhar a estrutura de memória correspondente (“caixinhas” com valores e setas)
- **Diagrama de classes UML** - É apresentado um problema cuja solução deve ser modelada através de UML, similar aos problemas do autocarro, elevador, carrinho de compras, etc. Poderão ter que usar o static (vai ser dado nesta aula). Será igualmente pedido para implementarem o método principal (ex: entrar())

# TPC teórico 2 - Jornal online



ALTERAÇÕES CLIMÁTICAS

## Depósitos de metano no Ártico estão a libertar-se

→ 760

Expedição internacional indica que há depósitos de metano congelados no Ártico que estarão a ser libertados para a atmosfera. Efeito de estufa do metano é 80 vezes superior ao dióxido de carbono.

REDES SOCIAIS

## Portugueses passam mais de 2h nas redes sociais

No que toca ao peso das redes sociais na utilização digital dos cidadãos da UE, Portugal ocupava em janeiro de 2020 o quinto lugar entre os Estados-membros que mais utilizavam estas plataformas.

(...)



ALTERAÇÕES CLIMÁTICAS

## Ártico. Cientistas descobrem depósitos de metano que se estão a libertar e podem acelerar aquecimento global

Expedição internacional indica que há depósitos de metano congelados no Ártico que estarão a ser libertados para a atmosfera. Efeito de estufa do metano é 80 vezes superior ao dióxido de carbono.

27 out 2020, 17:38

5

→ 760



Ana Catarina Peixoto  
Texto

É conhecido como um “gigante adormecido” do ciclo de carbono, mas a história pode estar a mudar na costa leste da Sibéria. Os resultados preliminares de um estudo feito por cientistas internacionais indicam que **há depósitos de metano congelados no Ártico que estarão a ser libertados para a atmosfera**, um fenómeno que pode acelerar o ritmo do aquecimento global. A notícia é avançada pelo [The Guardian](#), que refere que foram detetados altos níveis deste gás com efeito de estufa até uma profundidade de 350 metros no Mar

Comentários

Ártico. Cientistas descobrem depósitos de metano que se estão a libertar e podem acelerar aquecimento global

Comente e partilhe as suas ideias...

Mais votados (1)

Todos (5)

Marina Molares

2 h

as versões religiosas dos fim de mundo (Apocalipses) são muito mais giras. que as científicas. giro ver que só mudam as roupagens, a estrutura mistificadora do homem está lá toda.

3 · Responder



Herr Mando

12 h

O Observador agora anda copiar artigos do Guardian, esse grande referencia do jornalixo woke britânico, tenho duvidas sobre esses 80 vezes pior, mas e ciencia, logo tem de ser verdade.

2 · Responder



T R

18 h

A culpa é do Trump, claro.

1 · Responder



Paulo Alexandre : Ateu → T R

17 h

A culpa não é do Trump exceptuando aquela que decorre da negligência e da negação das alterações climáticas. Dele e de todos os que tudo têm feito para impedir a adopção de medidas que visem a reconversão industrial e económica.

5 · Responder



# TPC teórico 2 - Jornal online

Pretende-se desenvolver uma aplicação para gerir as notícias de jornais online, como o Observador, mostrado no slide anterior.

Deverá ser apresentada uma lista de notícias e clicando numa delas deverão ser apresentados os respetivos detalhes dessa notícia. A aplicação deve suportar a informação dos écrans apresentados no slide anterior (mas apenas esta).

Cada jornal (ex: observador, eco online, etc.) define um máximo de caracteres que pode ter um título e outro para as sinopses. Além disso, existe um controlo de “fake news” que faz com que notícias publicadas deixem de estar visíveis.

# TPC teórico 2 - Jornal online

ALTERAÇÕES CLIMÁTICAS

tópico

## Ártico. Cientistas descobrem depósitos de metano que se estão a libertar e podem acelerar aquecimento global

título

Expedição internacional indica que há depósitos de metano congelados no Ártico que estarão a ser libertados para a atmosfera. Efeito de estufa do metano é 80 vezes superior ao dióxido de carbono.

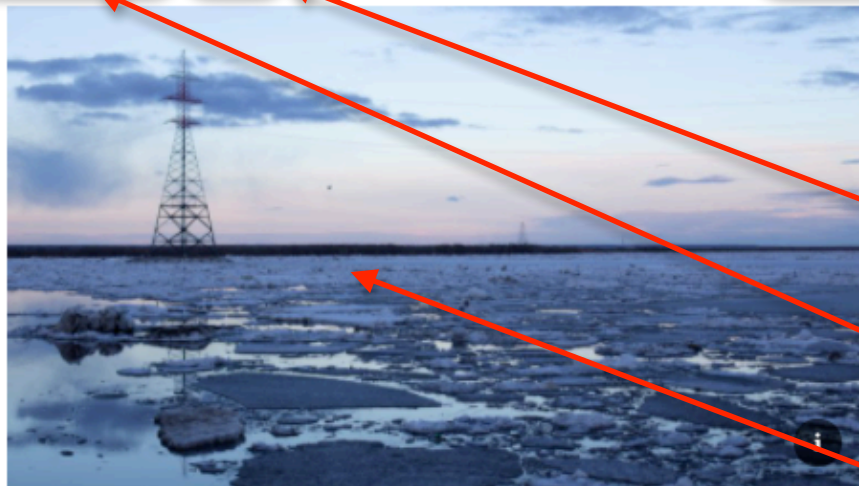
sinopse

27 out 2020, 17:38

5

760

número de partilhas



número de comentários

data de publicação

foto



Ana Catarina Peixoto  
Texto

autor (foto e nome)

É conhecido como um “gigante adormecido” do ciclo de carbono, mas a história pode estar a mudar na costa leste da Sibéria. Os resultados preliminares de um estudo feito por cientistas internacionais indicam que **há depósitos de metano congelados no Ártico que estarão a ser libertados para a atmosfera**, um fenómeno que pode acelerar o ritmo do aquecimento global. A notícia é avançada pelo [The Guardian](#), que refere que foram detetados altos níveis deste gás com efeito de estufa até uma profundidade de 350 metros no Mar

texto

# TPC teórico 2 - Jornal online

## Comentários

Artigo. Cientistas descobrem depósitos de metano que se estão a libertar e podem acelerar aquecimento global

Comente e partilhe as suas ideias...

Mais votados (1)

Todos (5)

Marina Molares

nome do autor

2 h

data/hora

as versões religiosas dos fim de mundo (Apocalipses) são muito mais giras. que as científicas. giro ver que só mudam as roupagens , a estrutura mistificadora do homem está lá toda.

3

número de votos

Herr Mando

O Observador agora anda copiar artigos do Guardian, esse grande referencia do jornalixo woke britanico, tenho duvidas sobre esses 80 vezes pior, mas e ciencia, logo tem de ser verdade.

texto

T R

A culpa é do Trump, claro.

18 h

1 Responder

resposta



Paulo Alexandre : Ateu → T R

17 h

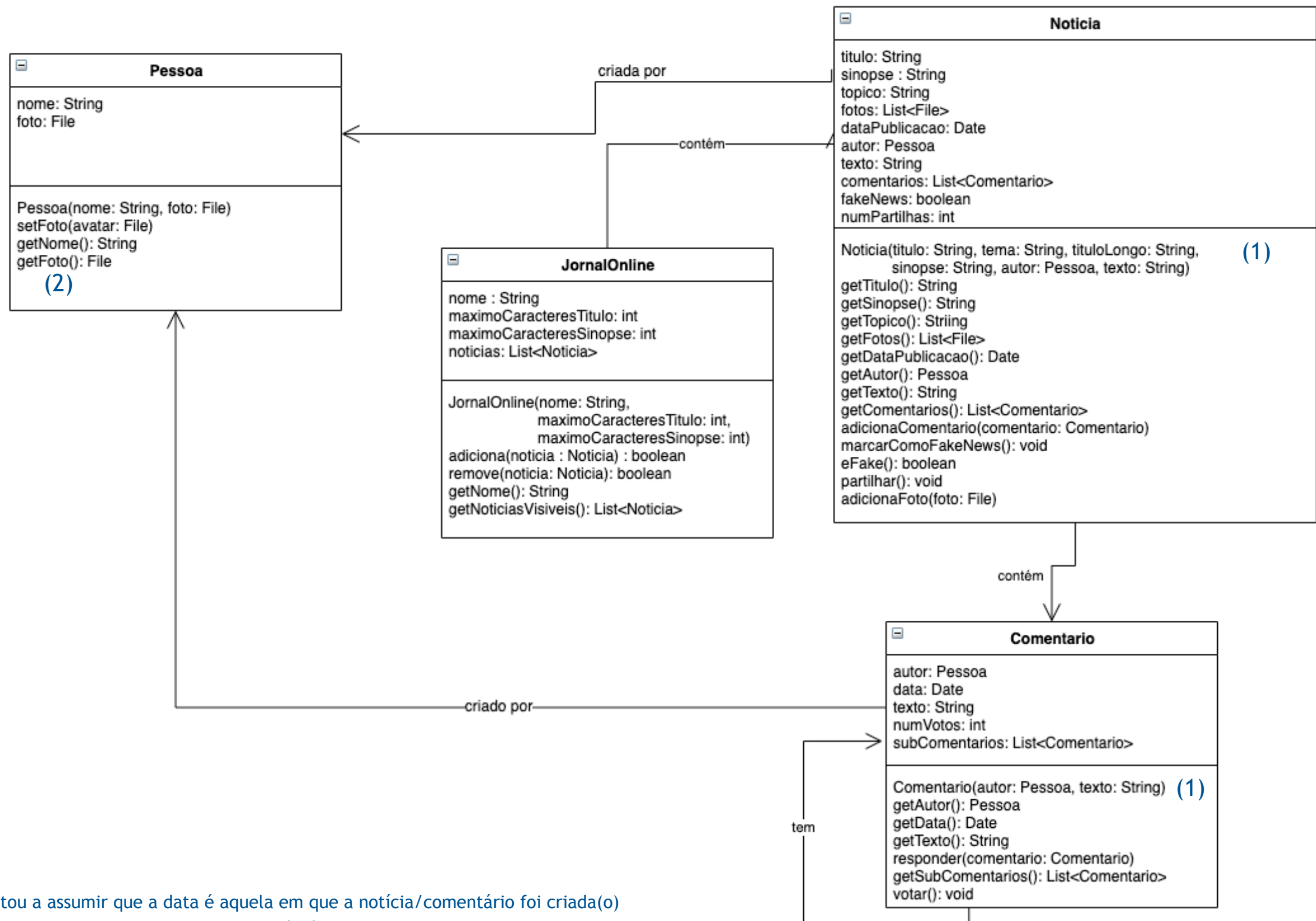
A culpa não é do Trump exceptuando aquela que decorre da negligência e da negação das alterações climáticas. Dele e de todos os que tudo têm feito para impedir a adopção de medidas que visem a reconversão industrial e económica.

5 Responder

foto do autor



# Resolução



(1) - Estou a assumir que a data é aquela em que a notícia/comentário foi criada(o)

(2) - Estou a assumir que a pessoa não muda de nome

# Resolução

```
boolean adiciona(Noticia noticia) {  
  
    // verifica se a notícia já existe  
    for (Noticia noticiaJaExistente: noticias) {  
        if (noticiaJaExistente.equals(noticia)) {  
            return false;  
        }  
    }  
  
    if (noticia.getTitulo().length() > maximoCaracteresTitulo) {  
        return false;  
    }  
  
    if (noticia.getSinopse().length() > maximoCaracteresSinopse) {  
        return false;  
    }  
  
    noticias.add(noticia);  
    return true;  
}
```

# Paradigmas de Programação

## Imperativo

```
class Calculadora {  
  
    static int soma(int a, int b) {  
        int resultado = a + b;  
        return resultado;  
    }  
}
```

- funções static
- variáveis são sempre locais à função
- funções sem efeitos secundários
- útil para implementar algoritmos

## Orientado a Objectos

```
class Calculadora {  
  
    private int resultado;  
  
    void soma(int a, int b) {  
        this.resultado = a + b;  
    }  
}
```

- funções sempre associadas a objectos
- variáveis de objectos
- funções com efeitos secundários (sem return)
- útil para implementar sistemas complexos

# Paradigmas de Programação

E se juntarmos o paradigma imperativo com o paradigma orientado a objectos??

# Classes vs Objetos

## (Revisão)

### Classe Utilizador

The form is titled 'Classe Utilizador' and contains the following fields:

- Name \***: Two text boxes for 'First' and 'Last' names.
- Time \***: Two text boxes for 'HH' and 'MM', followed by a dropdown menu for 'AM/PM' (currently showing 'AM').
- Email \***: A single text box with the instruction 'Please use your office email address.' below it.
- Date \***: Three text boxes for 'MM', 'DD', and 'YYYY', separated by slashes, with a calendar icon to the right.
- Address \***: Three text boxes for 'Street Address', 'Street Address Line 2', and 'City'.
- Region**: A text box for the 'Region'.

Classes são “formulários” em branco.

- Só existe um por aplicação
- Tem que se instanciar antes de utilizar (fazer uma “fotocópia”)

```
utilizador = new Utilizador()
```

- Após instanciar ficamos com um objeto dessa classe - uma cópia do formulário que podemos preencher



# Classes vs Objetos

## (Revisão)

### Objetos da classe Utilizador

The image shows three overlapping user registration forms, each representing an instance (object) of the 'Utilizador' class. The forms are identical in structure but contain different data, demonstrating how objects are created from a single class template.

**Name \***

First:  Last:

**Time \***

:

**Email \***

Please use your office email address.

**Date \***

/  /

**Address \***

Street Address

Street Address Line 2

City Region

# Objetos

Variáveis e métodos estão sempre associados a um objeto

```
Pessoa pessoa = new Pessoa();  
pessoa.idade = 30;  
pessoa.respira();
```

# Objetos

Cada objeto tem estado (variáveis) próprio

```
Pessoa pessoa1 = new Pessoa();  
pessoa1.idade = 30;
```

```
Pessoa pessoa2 = new Pessoa();  
pessoa2.idade = 32;
```

```
Pessoa pessoa3 = new Pessoa();  
pessoa3.idade = 18;
```

# Objetos

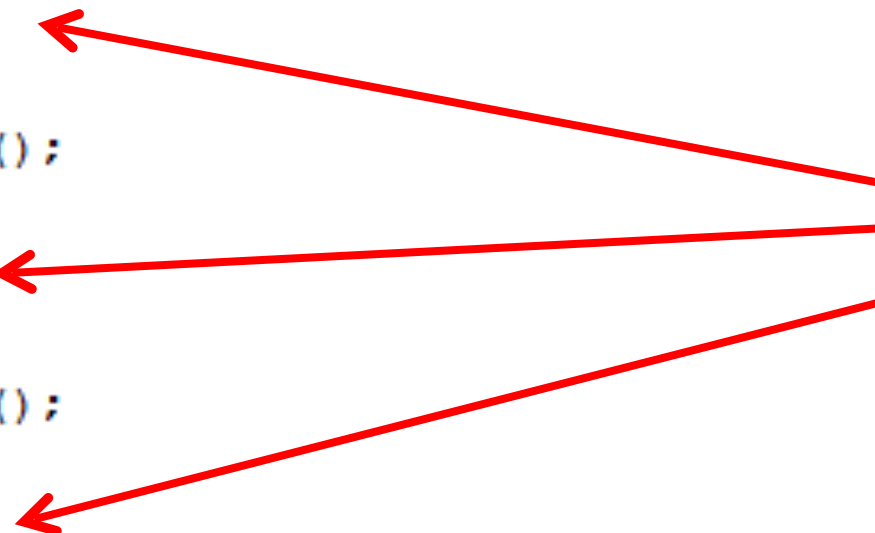
Mas por vezes, há variáveis que têm o mesmo valor para todos os objetos daquela classe

```
Pessoa pessoa1 = new Pessoa();  
pessoa1.idade = 30;  
pessoa1.idadeReforma = 65;
```

```
Pessoa pessoa2 = new Pessoa();  
pessoa2.idade = 32;  
pessoa2.idadeReforma = 65;
```

```
Pessoa pessoa3 = new Pessoa();  
pessoa3.idade = 18;  
pessoa3.idadeReforma = 65;
```

Todas as pessoas  
têm a mesma  
idadeReforma



# static

Variáveis **static** são transversais a todos os objetos da mesma classe  
Ou seja, são variáveis da classe e não do objeto!

```
class Pessoa {  
    static int idadeReforma;  
    int idade;  
}
```

```
Pessoa.idadeReforma = 65;
```

```
Pessoa pessoa1 = new Pessoa();  
pessoa1.idade = 30;
```

```
Pessoa pessoa2 = new Pessoa();  
pessoa2.idade = 32;
```

```
Pessoa pessoa3 = new Pessoa();  
pessoa3.idade = 18;
```

idadeReforma está associado à classe Pessoa e não a um objeto da classe Pessoa



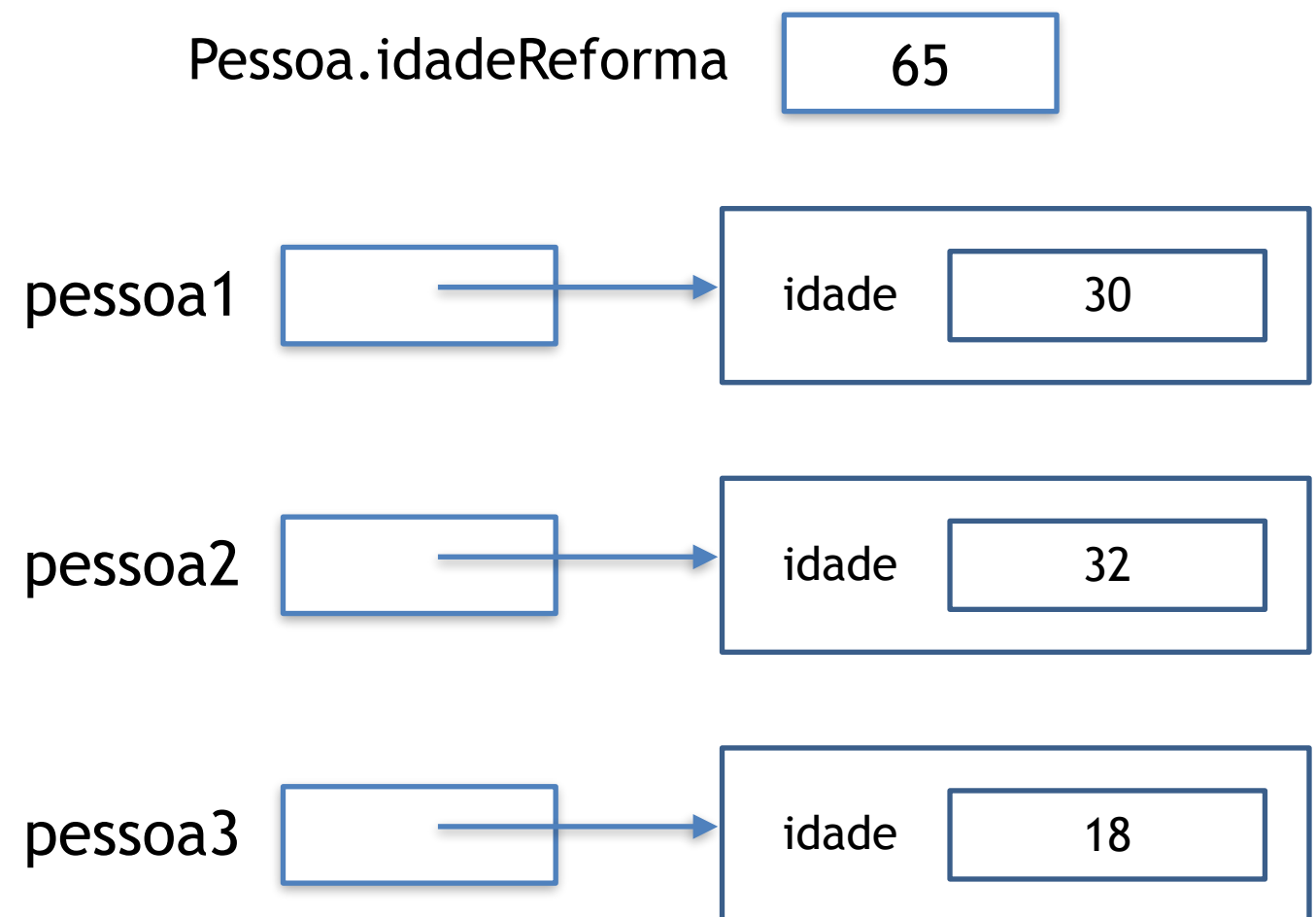
# static - estrutura de memória

```
Pessoa.idadeReforma = 65;
```

```
Pessoa pessoa1 = new Pessoa();  
pessoa1.idade = 30;
```

```
Pessoa pessoa2 = new Pessoa();  
pessoa2.idade = 32;
```

```
Pessoa pessoa3 = new Pessoa();  
pessoa3.idade = 18;
```



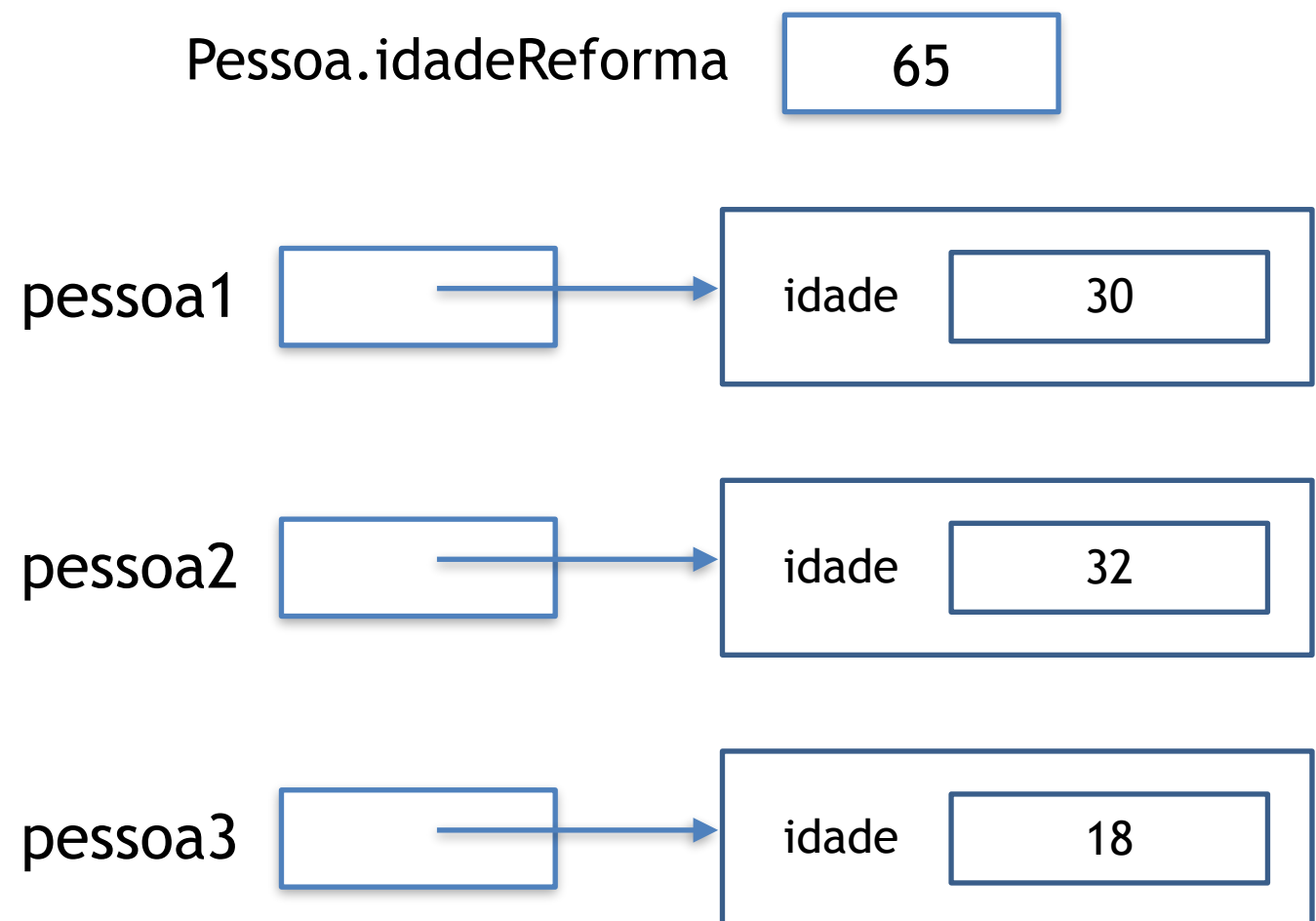
# static - estrutura de memória

```
Pessoa.idadeReforma = 65;
```

```
Pessoa pessoa1 = new Pessoa();  
pessoa1.idade = 30;
```

```
Pessoa pessoa2 = new Pessoa();  
pessoa2.idade = 32;
```

```
Pessoa pessoa3 = new Pessoa();  
pessoa3.idade = 18;
```



É como se misturássemos Programação Imperativa e Programação por Objectos no mesmo programa

# static

Podemos referenciar as variáveis através do objeto mas é de evitar!

```
Pessoa pessoa1 = new Pessoa();  
pessoa1.idade = 30;  
pessoa1.idadeReforma = 65;    // o compilador aceita mas é de evitar!!!
```

# static

As variáveis da classe (static) podem ser usadas em conjunto com as variáveis do objeto

```
class Pessoa {  
  
    static int idadeReforma;  
    int idade;  
  
    boolean estaReformada() {  
        return idade >= idadeReforma;  
    }  
}
```

```
Pessoa.idadeReforma = 65;  
  
Pessoa pessoa1 = new Pessoa();  
pessoa1.idade = 30;  
pessoa1.estaReformada(); // false  
  
Pessoa pessoa2 = new Pessoa();  
pessoa2.idade = 70;  
pessoa2.estaReformada(); // true
```

# static

As variáveis static são compartilhadas por todos os objetos dessa classe

```
class Pessoa {  
  
    static final int IDADE_REFORMA = 65;  
    static int numPessoas = 0;  
    int idade;  
  
    public Pessoa(int idade) {  
        this.idade = idade;  
        numPessoas++; // incrementa o número total de pessoas  
    }  
  
    boolean estaReformada() {  
        return idade >= IDADE_REFORMA;  
    }  
}
```

Equivalente a Pessoa.numPessoas++

```
Pessoa pessoa1 = new Pessoa(30);  
Pessoa pessoa2 = new Pessoa(35);  
  
System.out.println(Pessoa.numPessoas); // escreve "2"
```




# static

Muitas vezes, as variáveis da classe (static) não mudam (são constantes).  
Para isso, usamos o “**final**”

```
class Pessoa {  
  
    final static int IDADE_REFORMA = 65;  
    int idade;  
  
    boolean estaReformada() {  
        return idade >= IDADE_REFORMA;  
    }  
}
```

As constantes são variáveis static e final e costumam ser escritas em maiúsculas com underscores a separar as palavras



# Métodos

Tipicamente, os métodos manipulam variáveis do objeto

```
class Empregado {  
    int salarioBase;  
    int subsidioAlmoco;  
  
    int getSalario() {  
        return salarioBase + subsidioAlmoco;  
    }  
}
```

# Funções static

Na programação imperativa, as funções não manipulam nada, limitam-se a fazer algo com os seus parâmetros

```
int duplica(int valor) {  
    return valor * 2;  
}
```

← Não usa variáveis do objeto, usa apenas o parâmetro “valor”

Este método tem sempre o mesmo comportamento independentemente do objeto

# Métodos static

Os métodos que não manipulam variáveis do objeto devem ser **static** (métodos da classe)

```
static int duplica(int valor) {  
    return valor * 2;  
}
```

# Métodos static

Métodos static só podem referenciar variáveis ou outros métodos static

```
class Pessoa {  
  
    static final int IDADE_REFORMA = 65;  
    int idade;  
  
    boolean estaReformada() {  
        return idade >= IDADE_REFORMA;  
    }  
  
    public static void main(String[] args) {  
        System.out.println(IDADE_REFORMA); // correto  
        System.out.println(idade); // erro: não compila  
        estaReformada(); // erro: não compila  
    }  
}
```



# Métodos static

A classe Math (incluída no Java) só tem métodos static (exemplo: round())

```
class Math {  
  
    public static int round(float a) {  
        if (a != 0x1.fffffep-2f) // greatest float value less than 0.5  
            return (int)floor(a + 0.5f);  
        else  
            return 0;  
    }  
}
```



Não usa variáveis do objeto,  
usa apenas o parâmetro “a”

# Math

```
Math.random()  
Math.min(int x, int y)  
Math.min(long x, long y)  
Math.max(int x, int y)  
Math.abs(int x)  
...
```

# Métodos static

Não é necessário instanciar a classe Math  
(fazer `new Math()`)

```
int valor = Math.round(34.56799f);
```

# Exercício (breakout rooms)

## Quais as classes que não compilam?



```
class Classe1 {  
    static int x;  
  
    public void go() {  
        System.out.println(x);  
    }  
}
```

```
class Classe4 {  
    static final int x = 12;  
  
    public void go() {  
        System.out.println(x);  
    }  
}
```

```
class Classe2 {  
    int x;  
  
    public static void go() {  
        System.out.println(x);  
    }  
}
```

```
class Classe5 {  
    static final int x = 12;  
  
    public void go(int x) {  
        System.out.println(x);  
    }  
}
```


```
class Classe3 {  
    final int x;  
  
    public void go() {  
        System.out.println(x);  
    }  
}
```

```
class Classe6 {  
    int x = 12;  
  
    public static void go(int x) {  
        System.out.println(x);  
    }  
}
```


Vai ser  
chamado  
um aluno  
de cada  
grupo para  
responder e  
justificar

# Resolução


```
class Classe1 {  
    static int x;  
  
    public void go() {  
        System.out.println(x);  
    }  
}
```




```
class Classe4 {  
    static final int x = 12;  
  
    public void go() {  
        System.out.println(x);  
    }  
}
```




```
class Classe2 {  
    int x;  
  
    public static void go() {  
        System.out.println(x);  
    }  
}
```




```
class Classe5 {  
    static final int x = 12;  
  
    public void go(int x) {  
        System.out.println(x);  
    }  
}
```



```
class Classe3 {  
    final int x;  
  
    public void go() {  
        System.out.println(x);  
    }  
}
```



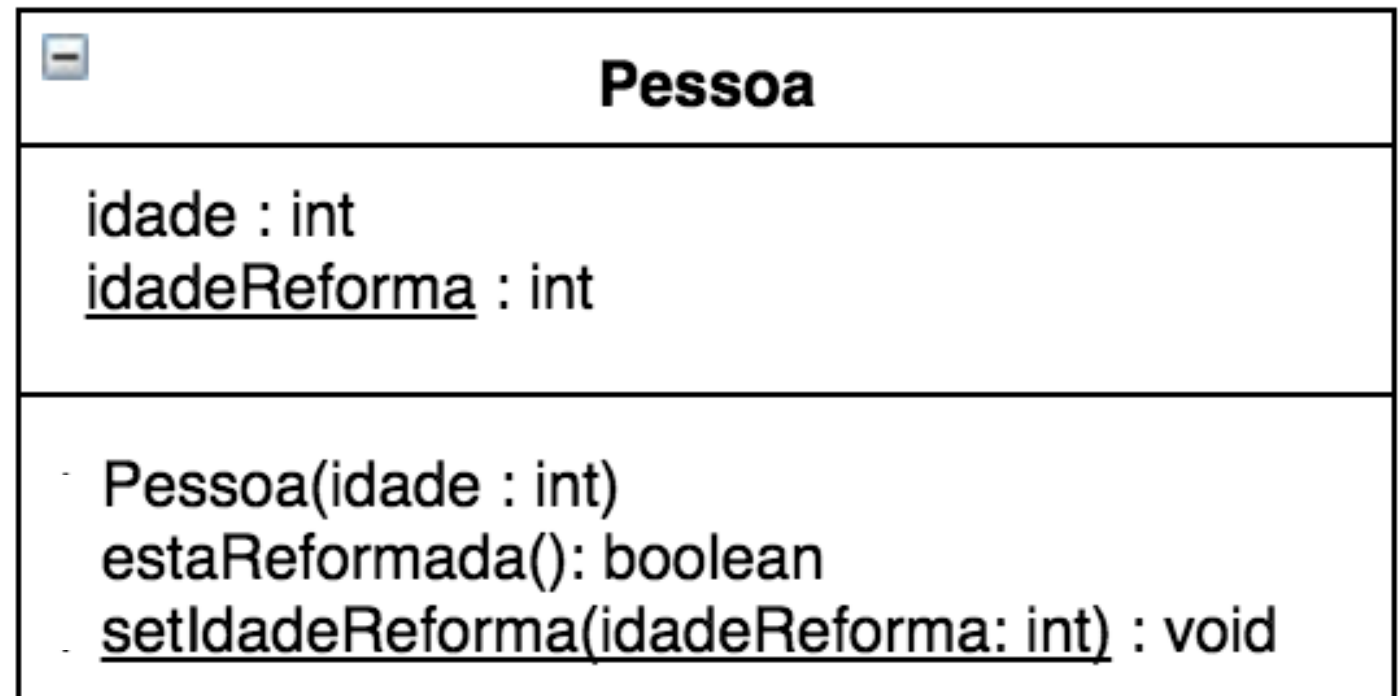
```
class Classe6 {  
    int x = 12;  
  
    public static void go(int x) {  
        System.out.println(x);  
    }  
}
```



# UML

Em UML, as variáveis e métodos static devem estar sublinhados

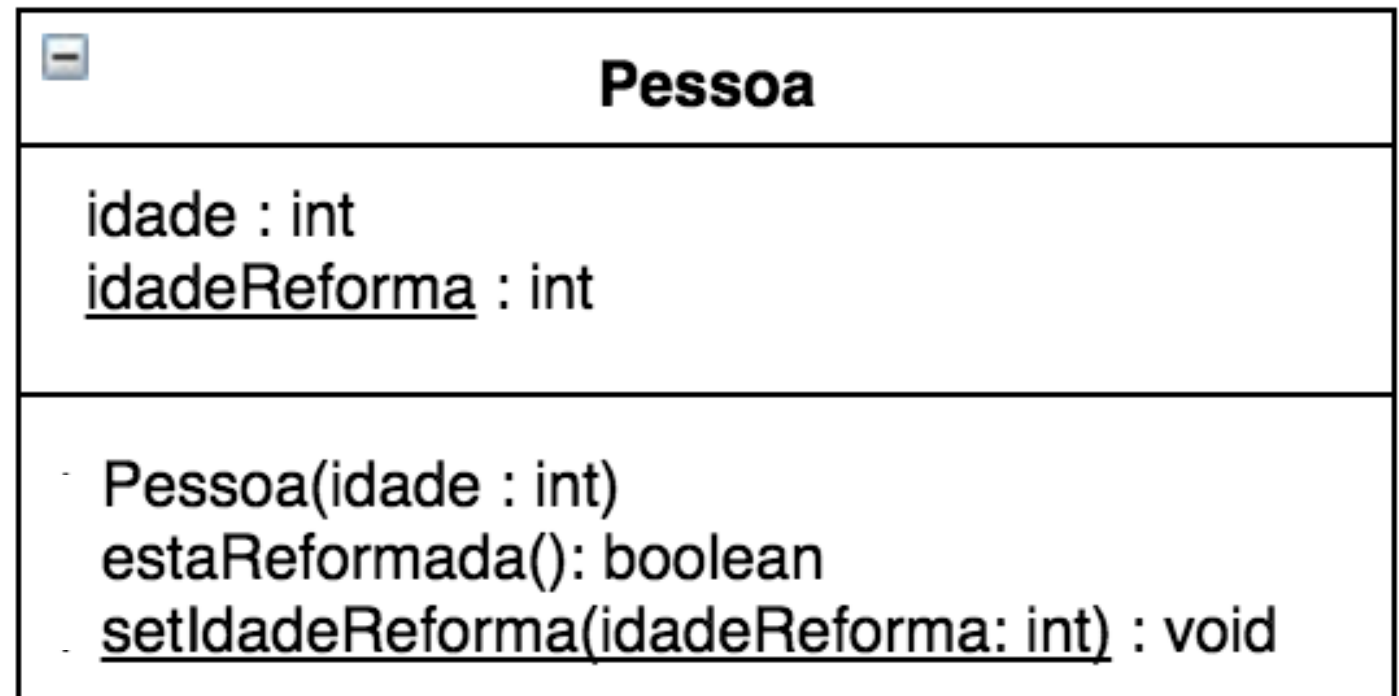
```
class Pessoa {  
    int idade;  
    static int idadeReforma;  
  
    Pessoa(int idade) {  
        this.idade = idade;  
    }  
  
    boolean estaReformada() {  
        return this.idade >= idadeReforma;  
    }  
  
    static void setIdadeReforma(int idadeReforma) {  
        Pessoa.idadeReforma = idadeReforma;  
    }  
}
```



# UML

Em UML, as variáveis e métodos static devem estar sublinhados

```
class Pessoa {  
    int idade;  
    static int idadeReforma;  
  
    Pessoa(int idade) {  
        this.idade = idade;  
    }  
  
    boolean estaReformada() {  
        return this.idade >= idadeReforma;  
    }  
  
    static void setIdadeReforma(int idadeReforma) {  
        Pessoa.idadeReforma = idadeReforma;  
    }  
}
```



Cábula UML será atualizada com esta informação