Material para
a Formación Profesional inicial

A03. Constantes e método estáticos

Familia profesional	IFC	Informática e comunicacións
Ciclo formativo	CSIFC02 CSIFC03	Desenvolvemento de aplicacións multiplataforma Desenvolvemento de aplicacións web
Grao		Superior
Módulo profesional	MP0485	Programación
Unidade didáctica	UD04	Conceptos avanzados de POO
Actividade	**A02**	**Constantes e métodos estáticos**
Autores		Silvia Framiñán Fondevila Marta Rey López
Nome do arquivo		CSIF02_MP0485_V000402_UD04_A02_Constantes_metodos_estaticos.odt

Índice

1. Ficha técnica

Contexto da actividade

Módulo	Duración	Unidade didáctica.	Sesións 60´	Actividades	Sesións 60´
MP0485. Programación.	240	UD01. Identificación dos elementos dun programa informático.	24	A01. Metodoloxía da programación.	5
				A02. Introdución á linguaxe de programación Java.	7
				A03. Elementos básicos dun programa Java.	12
		UD02. Uso de estruturas de control.	20	A01. Estruturas de control selectivas.	8
				A02. Estruturas de control repetitivas.	12
		UD03. Introdución á POO.	28	A01. Conceptos de POO.	8
				A02. Clases e obxectos.	20
		U04. Conceptos avanzados de POO.	**20**	A01. Obxectos predefinidos e métodos estáticos.	10
				A02. Constantes e métodos estáticos.	**5**
				A03. Visibilidade e empaquetaxe.	5
		UD05. Lectura e escritura de información.	34	A01. Fluxos de entrada e saída.	4
				A02. Uso de ficheiros.	15
				A03. Interfaces gráficas de usuario.	15
		UD06. Aplicación das estruturas de almacenamento.	37	A01. Uso de cadeas.	8
				A02. Uso de arrais.	9
				A03.Uso de coleccións e outras estruturas	12
				A04. Manipulación de documentos XML.	8
		UD07. Xerarquías de clases e excepcións.	37	A01. Introdución á herdanza.	10
				A02. Uso avanzado da herdanza.	10
				A03. Interfaces.	10
				A04. Control de código. Excepcións.	7
		UD08. Mantemento da persistencia dos obxectos.	18	A01. Instalación dun SXBDOO e almacenamento básico de obxectos.	8
				A02. Almacenamento e recuperación de obxectos nun SXBDOO e operacións con datos complexos.	10
		UD09. Xestión dos datos almacenados nas bases de datos relacionais.	22	A01. Conexión con sistemas xestores de bases de datos relacionais.	6
				A02. Operacións de lectura nunha base de datos relacional.	7
				A03. Operacións de escritura nunha base de datos relacional e uso de transaccións.	9

NOTA: Esta actividade está vinculada á programación recollida no arquivo CSIFC02_MP0485_V000400_UD04_Conceptos_avanzados_POO.pdf

Título da actividade

Nº	Título	Descrición	Duración

A02	Constantes e métodos estáticos	Nesta actividade introducirase o concepto de método estático e a súa utilización no funcionamento das clases. Ademais, definiranse as constantes de clase, facendo fincapé nas particularidades das constantes estáticas.	5

Resultados de aprendizaxe do currículo

Resultados de aprendizaxe do currículo	Completo
▪ RA4. Desenvolve programas organizados en clases, para o que analiza e aplica os principios da programación orientada a obxectos.	**NON**

Obxectivos didácticos e título e descrición da actividade

Obxectivos específicos		Actividade		Descrición básica	Duración
O2.1	Definír e utilizar métodos estáticos.	A03	Constantes e métodos estáticos.	Nesta actividade introducirase o concepto de método estático e a súa utilización no funcionamento das clases. Ademais, definiranse as constantes de clase, facendo fincapé nas particularidades das constantes estáticas.	5
O2.2	Definir e utilizar atributos estáticos.				

Criterios de avaliación

Criterios de avaliación
▪ CA4.8. Defínironse e utilizáronse métodos estáticos. ▪ CA4.11. Defínironse e utilizáronse atributos estáticos.

Contidos

Contidos
▪ BC4. Desenvolvemento de clases: – Atributos e métodos estáticos.

Actividades de ensino e aprendizaxe e de avaliación, métodos, recursos e instrumentos de avaliación (exemplo)

Que e para que	Como			Con que	Como e con que se valora	
Actividade (título e descrición)	**Profesorado (en termos de tarefas)**	**Alumnado (tarefas)**	**Resultados ou produtos**	**Recursos**	**Instrumentos e procedementos de avaliación**	
A02. Constantes e métodos estáticos. ▪ Nesta actividade introducirase o concepto de método estático e a súa utilización no funcionamento das clases. Ademais, definiranse as constantes de clase, facendo fincapé nas particularidades das constantes estáticas.	▪ **Tp2.1** - Explicación dos métodos e atributos estáticos. ▪ **Tp2.2** - Descrición das constantes de clase. ▪ **Tp2.3** - Exposición do uso dos métodos, atributos e constantes estáticas.	▪ **Ta2.1**. Creación dunha clase con métodos estáticos. ▪ **Ta2.2**. Emprego de atributos estáticos.	▪ Proxecto Java.	▪ Ordenador persoal con conexión a Internet. ▪ Contorno de desenvolvemento NetBeans. ▪ Java SE Development Kit. ▪ Apuntamentos da profesora. ▪ Proxector.		3
		▪ **Ta2.3** - Tarefa de avaliación: – Táboa de indicadores sobre un proxecto Java no que se definan e empreguen métodos estáticos. – Táboa de indicadores sobre un proxecto Java no que se definan e empreguen atributos estáticos.	▪ Proxecto Java.		▪ OU 5. Táboa de indicadores sobre un proxecto Java no que se definan e empreguen métodos estáticos. ▪ OU 7. Táboa de indicadores sobre un proxecto Java no que se definan e empreguen atributos estáticos.	2

2. A02. Constantes e métodos estáticos.

2.1 Introdución

Na actividade que nos ocupa aprenderanse os seguintes conceptos e manexo de destrezas sobre a Programación Orientada a Obxectos:

- Definición de métodos e atributos estáticos.
- Uso de atributos e métodos estáticos.

2.2 Actividade

2.2.1 Métodos e atributos estáticos ou de clase

Un atributo estático ou de clase é aquel que non é específico de cada obxecto, senón que só hai unha copia do mesmo e o seu valor é compartido por todos os obxectos da clase. Existe e pode empregarse aínda que non existan obxectos desa clase. Pódese empregar, por exemplo, nun caso de produtos á venda, para aplicar un desconto común a todos os produtos desa clase. Declárase antepoñendo a palabra `static` ao tipo, na sinatura do método:

```
public class Coche {
    private String marca;
    private String modelo;
    static double descuento = 0.2;
}
```

Un método de clase é aquel que se invoca sen instanciar un obxecto desa clase. Para declarar un método de clase, introduciremos antes do tipo de dato de retorno a palabra reservada `static`. Estes métodos só poden facer uso dos parámetros que se lles pasan ou de atributos definidos tamén de forma estática.

```
public class Áreas {
    public static double cadrado(double lado) {
        return lado * lado;
    }

    public static double rectángulo(double base, double altura) {
        return base * altura;
    }
}
```

Podes atopar información sobre como empregar métodos estáticos correctamente na seguinte ligazón: https://www.arquitecturajava.com/uso-de-java-static-method/

2.2.2 Constantes de clase

As constantes de clase decláranse coma os atributos de clase pero deben ir acompañados das palabras clave `static` (que indica que é un atributo que pertence á clase e non á instancia) e `final` (para indicar que non pode modificarse e é constante):

```
public/private static final TipoDaConstante = valorDaConstante;
```

Por exemplo:

```
public static final double PI = 3.1416;
```

2.2.3 Uso de métodos, atributos e constantes estáticas

A súa chamada pode facerse directamente a partir do nome da clase ou desde o nome do obxecto (se a clase non é estática e pode instanciarse):

```
public class Xeometría {
    public static void main(String[] args) {
        System.out.println("Área cadrado:" + Áreas.cadrado(2));
        System.out.println("Área rectángulo:" + Áreas.rectángulo(3,2));
        System.out.println("PI:" + Áreas.PI);
    }
}
```

Ou:

```
public class Xeometría {
    public static void main(String[] args) {
        Áreas oMeuObxectoAreas = new Áreas();
        System.out.println("Área cuadrado:" + oMeuObxectoAreas.cadrado(4));
        System.out.println("Área rectángulo:" + oMeuObxectoAreas.rectángu-
lo(3,4));
        System.out.println("PI:" + oMeuObxectoAreas.PI);
    }
}
```

Realiza a Tarefa 3.1. “Creación dunha clase con métodos estáticos”, na que se implementará unha clase Cilindro que permitirá calcular a súa área e volume.

Realiza a Tarefa 3.2. “Emprego de atributos estáticos”, na que se implementará parte dunha clase na que se utilizarán atributos estáticos.

2.3 Tarefas

As tarefas propostas para esta actividade son as seguintes:

- Tarefa 3.1. **Creación dunha clase con métodos estáticos**, na que se implementará unha clase Cilindro que permitirá calcular a súa área e volume.
- Tarefa 3.2. **Emprego de atributos estáticos**, na que se implementará parte dunha clase na que se utilizarán atributos estáticos.

2.3.1 Tarefa 3.1. Creación dunha clase con métodos estáticos

Nesta tarefa crearase unha clase Cilindro da que se poderán calcular a súa área e volume.

Enunciado

Implementa en Java unha clase Cilindro con dous parámetros: radio e altura. A clase conterá dous métodos estáticos que recibirán os parámetros radio (r) e altura (h) e farán o cálculo da área e volume do cilindro. Tamén conterá dous métodos non estáticos área e volume que non recibirán parámetros e farán tamén o cálculo da área e volume pero cos valores dos atributos do obxecto (para o que terán que chamar aos métodos estáticos).

Crea un programa principal para probar os métodos da clase.

A área do cilindro pode calcularse como:

$$A = 2 \cdot \pi \cdot r \cdot (r + h)$$

E o seu volume:

$$V = \pi \cdot r^2 \cdot h$$

Solución

- Cilindro.java

```
package poliedros;

public class Cilindro {
    private double radio;
    private double altura;

    public Cilindro(double radio, double altura) {
        this.radio = radio;
        this.altura = altura;
    }

    public double getRadio() {
        return radio;
    }

    public void setRadio(double radio) {
        this.radio = radio;
    }

    public double getAltura() {
        return altura;
    }
```

```
    public void setAltura(double altura) {
        this.altura = altura;
    }

    public static double area(double radio, double altura) {
        return 2 * Math.PI * radio * (radio + altura);
    }

    public static double volume(double radio, double altura) {
        return Math.PI * Math.pow(radio,2) * altura;
    }

    public double area() {
        return this.area(this.radio, this.altura);
    }

    public double volume() {
        return this.volume(this.radio, this.altura);
    }
}
```

- Poliedro.java

```
package poliedros;

public class Poliedros {

    public static void main(String[] args) {
        //Chamada aos métodos estáticos
        System.out.println(Cilindro.area(2,4));
        System.out.println(Cilindro.volume(2,4));

        //Chamada aos métodos das instancias
        Cilindro meuCilindro = new Cilindro(3,5);
        System.out.println(meuCilindro.area());
        System.out.println(meuCilindro.volume());
    }
}
```

2.3.2 Tarefa 3.2. Emprego de atributos estáticos

Nesta tarefa implementarase unha clase Empregado que conterá atributos estáticos e constantes de clase.

Enunciado

Implementa unha clase Empregado que terá como atributos o nome e a antigüidade do mesmo, ademais dun atributo estático que será o salario base dun empregado. Conterá un método estático que permitirá aumentar o salario base na porcentaxe indicada e un método de instancia que fará o cálculo e devolverá o salario do empregado, que será o salario base máis 10€ (constante da clase) por cada ano de antigüidade.

Crea un programa principal para probar a funcionalidade da clase.

Solución

- Empregado.java

```
package empresa;

public class Empregado {
    private String nome;
    private int antigüidade;
    private static double salarioBase = 1000;
```

```
    private final static double incrementoAno = 10;

    public Empregado(String nome, int antigüidade) {
        this.nome = nome;
        this.antigüidade = antigüidade;
    }

    public String getNome() {
        return nome;
    }

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

    public int getAntigüidade() {
        return antigüidade;
    }

    public void setAntigüidade(int antigüidade) {
        this.antigüidade = antigüidade;
    }

    public static double getSalarioBase() {
        return salarioBase;
    }

    public static void setSalarioBase(double salarioBase) {
        Empregado.salarioBase = salarioBase;
    }

    public static void incrementarSalarioBase(double porcentaxe) {
        salarioBase = salarioBase * (100 + porcentaxe) / 100;
    }

    public double salario() {
        return salarioBase + incrementoAno * antigüidade;
    }
}
```

- Empresa.java

```
package empresa;

public class Empresa {

    public static void main(String[] args) {
        System.out.println("O salario base é de " + Empregado.getSalarioBase() +
"€");
        Empregado manolo = new Empregado("Manolo", 10);
        System.out.println("O empregado " + manolo.getNome() + " cobra " + mano-
lo.salario() + "€");
        Empregado.incrementarSalarioBase(2); //Increméntase o salarioBase para
todos os empregados nun 2%
        System.out.println("O salario base é de " + Empregado.getSalarioBase() +
"€");
        System.out.println("O empregado " + manolo.getNome() + " cobra despois
do aumento " + manolo.salario() + "€");
    }
 }
```

- Resultado

```
O salario base é de 1000.0€
O empregado Manolo cobra 1100.0€
O salario base é de 1020.0€
O empregado Manolo cobra despois do aumento 1120.0€
```

3. Materiais

3.1 Documentos de apoio ou referencia

- Especificación da linguaxe Java: http://docs.oracle.com/javase/specs/
- Java SE 8 API Documentation: https://docs.oracle.com/javase/8/docs/api/index.html
- The Java Tutorials: https://docs.oracle.com/javase/tutorial/
- V. RESÚA EIRAS. POOJava. http://iespazodamerce.es/wiki/index.php?title=POOJava
- Java static. Atributos y métodos estáticos o de clase. http://puntocomnoesunlenguaje.blogspot.com.es/2013/02/java-static.html
- J. SÁNCHEZ. Manual de Java. http://jorgesanchez.net/java
- P.A. SZNAJDLEDER. Java a fondo. Alfaomega. 2ª edición.
- Ejercicios propuestos y resueltos programación orientado a objetos Java. https://www.discoduroderoer.es/ejercicios-propuestos-y-resueltos-programacion-orientado-a-objetos-java/
- J. BOBADILLA SANCHO. Java a través de ejemplos. Ra-Ma. Ed. 2003.
- Clases básicas predefinidas. Depto. Lenguajes y Ciencias de la Computación. E.T.S.I. Informática. Universidad de Málaga. http://www.lcc.uma.es/~vicente/docencia/poo/teoria/poo_4_cbasic.pdf
- Wikipedia. https://www.wikipedia.org

3.2 Recursos didácticos

- Ordenador persoal con conexión a Internet.
- Contorno de desenvolvemento NetBeans.
- Java SE Development Kit.
- Apuntamentos da profesora.
- Proxector.

4. Avaliación

Criterios de avaliación seleccionados para esta actividade (exemplo)	Evidencia de aprendizaxe	Instrumento de avaliación (tomados da aplicación de programación)	Peso na cualificación da UD
▪ CA4.8. Defiríronse e utilizáronse métodos estáticos.	▪ Proxecto Java	▪ OU 5. Táboa de indicadores sobre un proxecto Java no que se definan e empreguen métodos estáticos.	20%
▪ CA4.11. Defiríronse e utilizáronse atributos estáticos.	▪ Proxecto Java	▪ OU 7. Táboa de indicadores sobre un proxecto Java no que se definan e empreguen atributos estáticos.	20%

4.1 Modelo de proba

Enunciado

Implementa unha clase Coche que conteña os atributos marca, modelo, cabalos fiscais e matrícula. Aos tres primeiros daráselles valor no construtor, á matrícula, no momento da venda. Existirá un atributo de clase co número de coches vendidos ata o momento. A clase terá un método venda que recibirá a matrícula do coche e o escribirá no seu atributo, ademais de incrementar en 1 o número de coches vendidos.

Ademais, a clase conterá un método estático para calcular o Imposto sobre Vehículos de Tracción Mecánica (IVTM), ao que se lle pasarán o número de cabalos fiscais, así como outro método de instancia que calcula o IVTM pasándolle ao devandito método estático os cabalos fiscais almacenados no atributo do obxecto.

- Menos de 8 cabalos fiscais 22,20 €
- De 8 ata 11,99 cabalos fiscais 61,75 €
- De 12 ata 15,99 cabalos fiscais 133,95 €
- De 16 ata 19,99 cabalos fiscais 179,22 €
- De máis de 20 cabalos fiscais 224,00 €

Define os métodos getters e setters para os atributos que proceda.

Crea un programa principal para probar toda a funcionalidade da clase.

Solución

- Coche.java

```
package vehículos;

public class Coche {

    private String marca;
    private String modelo;
    private double cabalosFiscais;
    private String matricula;
    private static int numeroVendidos = 0;

    public Coche(String marca, String modelo, int cabalosFiscais) {
        this.marca = marca;
```

```
        this.modelo = modelo;
        this.cabalosFiscais = cabalosFiscais;
    }

    public String getMarca() {
        return marca;
    }

    public void setMarca(String marca) {
        this.marca = marca;
    }

    public String getModelo() {
        return modelo;
    }

    public void setModelo(String modelo) {
        this.modelo = modelo;
    }

    public double getCabalosFiscais() {
        return cabalosFiscais;
    }

    public void setCabalosFiscais(double cabalosFiscais) {
        this.cabalosFiscais = cabalosFiscais;
    }

    public static int getNumeroVendidos() {
        return numeroVendidos;
    }

    public void vender(String matricula) {
        this.matricula = matricula;
        numeroVendidos++; //Incrementamos o número de coches vendidos
    }

    public static double IVTM(double cabalosFiscais) {
        if (cabalosFiscais < 8) {
            return 22.20;
        } else if (cabalosFiscais < 12) {
            return 61.75;
        } else if (cabalosFiscais < 16) {
            return 133.95;
        } else if (cabalosFiscais < 20) {
            return 179.22;
        } else {
            return 224;
        }
    }

    public double IVTM() {
        return IVTM(this.cabalosFiscais);
    }

}
```

- Vehículos.java

```
package vehículos;

public class Vehículos {

    public static void main(String[] args) {
        Coche meuCoche = new Coche("Renault", "Megane", 10);
        System.out.println("Número de coches vendidos: " + Coche.getNumeroVendi-
dos());
        meuCoche.vender("1234-HRM");
        System.out.println("Número de coches vendidos: " + Coche.getNumeroVendi-
dos());
```

```
        double cabalosFiscais = 17;
        System.out.println("O IVTM dun coche de " + cabalosFiscais + " cabalos
Fiscais é de " + Coche.IVTM(cabalosFiscais) + "€");
        System.out.println("O IVTM do meu coche é de " + meuCoche.IVTM() + "€");
    }

}
```

- Resultado

```
Número de coches vendidos: 0
Número de coches vendidos: 1
O IVTM dun coche de 17.0 cabalos Fiscais é de 179.22€
O IVTM do meu coche é de 61.75€
```

Táboa de indicadores

OU 5. Táboa de indicadores sobre un proxecto Java no que se definan e empreguen métodos estáticos.

Nome:		Data:	
▪ CA4.8. Defiríronse e utilizáronse métodos estáticos.			
Indicadores	**Puntuación base**	**Puntuación obtida**	**Observacións**
▪ Declarouse correctamente o método estático IVTM.	2,5		
▪ Declarouse o método de instancia IVTM diferenciándoo correctamente do método estático.	2		
▪ Declaráronse correctamente os métodos getters e setters diferenciando os estáticos dos de instancia.	2		
▪ Empregáronse correctamente os métodos estáticos.	2		
▪ Empregáronse correctamente os métodos de instancia diferenciándoos dos estáticos.	1,5		

OU 7. Táboa de indicadores sobre un proxecto Java no que se definan e empreguen atributos estáticos.

Nome:		Data:	
▪ CA4.11. Defiríronse e utilizáronse atributos estáticos.			
Indicadores	**Puntuación base**	**Puntuación obtida**	**Observacións**
▪ Definiuse correctamente o atributo estático co número de vehículos vendidos.	3		
▪ Defiríronse correctamente os atributos de instancia diferenciándoos correctamente dos estáticos.	2		
▪ Actualizouse correctamente o atributo estático co número de vehículos.	3		
▪ Empregouse o construtor para dar valor aos atributos que corresponde.	2		