

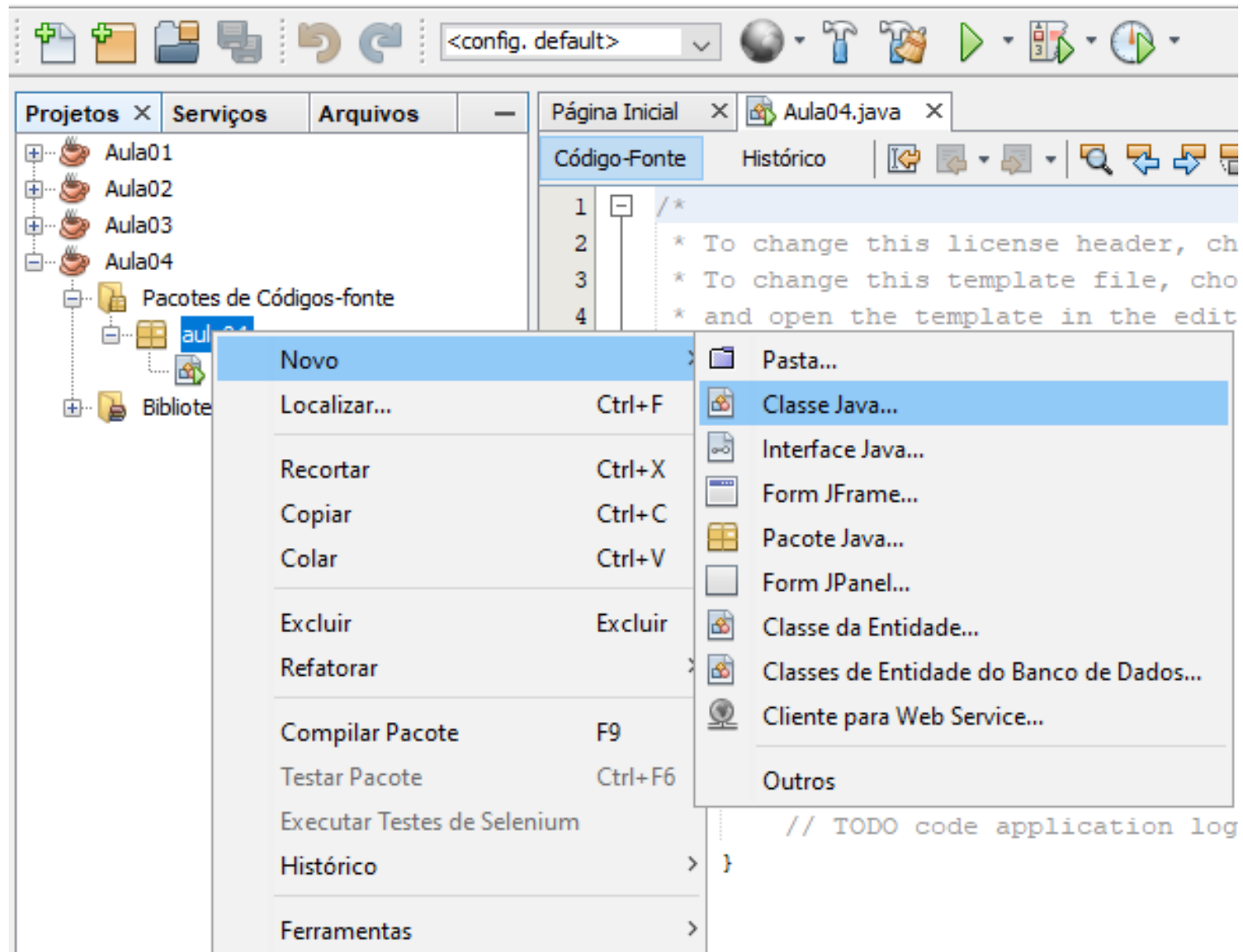
Introdução a Orientação à Objetos

Capitulo III

Revisão

- ✓ Classe: Abstração lógica de uma entidade (reita, forma);
- ✓ Objeto: “materialização” de uma classe (bolo de cenoura, fusca);
- ✓ Instância: processo que permita a existência do objeto na memória;
- ✓ Atributos: características de cada objeto (cor, saldo, idade);
- ✓ Métodos: ações realizadas, funções de cada objeto (andar, calcular área);

Criando nossa própria Classe



Criando nossa própria Classe

Não utilizar espaços;
Primeira letra Maiúscula;

New Classe Java

Etapas

1. Escolher Tipo de Arquivo
2. **Nome e Localização**

Nome e Localização

Nome da Classe:

Projeto:

Localização:

Pacote:

Arquivo Criado:

< Voltar Próximo > **Finalizar** Cancelar Ajuda

Criando nossa própria Classe

```
public class Funcionario {  
    // Atributos  
    int idade;  
    int cpf;  
    float salario;  
    String nome;  
  
    // Métodos  
    void tiraFerias(String mes){  
        System.out.println( nome + " vai tirar férias no mês de " + mes);  
    }  
  
    float calculaSalarioAnual(){  
        float salarioAnual;  
        salarioAnual = salario * 12;  
        return salarioAnual;  
    }  
}
```

→ Início da classe

Os atributos da Classe são suas características.
O que tem um funcionário?

Relacionamento entre classes

Os métodos da Classe são ações ou Comportamento.
O que faz um funcionário?

→ Fim da classe

Criando nossa própria Classe

Tipo de retorno do método

Parâmetro do método

```
// Métodos
void tiraFerias(String mes) {
    System.out.println( nome + " vai tirar férias no mês de " + mes);
}

float calculaSalarioAnual() {
    float salarioAnual;
    salarioAnual = salario * 12;
    return salarioAnual;
}
```

Variável local

Instanciando Objetos

```
public static void main(String[] args) {  
    // Declarando dois "Objetos"  
    Funcionario f1;  
    Funcionario f2;  
    // Instanciando os dois "Objetos"  
    f1 = new Funcionario();  
    f2 = new Funcionario();  
    // Atribuindo valores aos Objetos  
    f1.cpf = 123_456_789;  
    f1.nome = "Samuel Souza";  
    f1.salario = 400f;  
  
    f2.cpf = 987_654_321;  
    f2.nome = "Renzo Mesquita";  
    f2.salario = 1000f;  
  
    //Exibindo informações dos objetos  
    System.out.printf("Funcionario: %s ; CPF: %d %n", f1.nome, f1.cpf);  
    System.out.printf("Funcionario: %s ; CPF: %d %n", f2.nome, f2.cpf);  
}
```

Acessando um atributo da classe através do operador de seção "."

Invocando (ou chamando) métodos

```
// Criando variáveis auxiliares
float salAnual1; //Salário anual do funcionário1
float salAnual2; //Salário anual do funcionário2
```

```
salAnual1 = f1.calculaSalarioAnual();
salAnual2 = f2.calculaSalarioAnual();
```

Métodos que retornam valores
precisam ser armazenados em uma
variável

```
f1.tiraFerias("Janeiro");
f2.tiraFerias("Dezembro");
```

Métodos sem retorno

```
//Exibindo informações dos objetos
```

```
System.out.printf("Salario: %f ; Salario Anual: %f %n", f1.salarario, salAnual1);
System.out.printf("Salario: %f ; Salario Anual: %f %n", f2.salarario, salAnual2);
```

run:

Samuel Souza vai tirar ferias em Janeiro

Renzo Mesquita vai tirar ferias em Dezembro

Salario: 400,000000 ; Salario Anual: 4800,000000

Salario: 1000,000000 ; Salario Anual: 12000,000000

CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)

Também
utilizamos do "."
para acessar
métodos

Inicializando Objetos

Atalho:
ALT + INSERT

Chamada ao Construtor

```
f1 = new Funcionario();  
f2 = new Funcionario();
```

O Construtor é um método que é chamado sempre que instanciamos uma Classe.

O Construtor sempre terá o mesmo nome da Classe.

Por padrão o Construtor vem vazio, porém podemos editá-lo

```
public class Funcionario {  
  
    int idade;  
    int cpf;  
    float salario;  
    String nome;  
  
    public Funcionario() {  
        System.out.println( "Funcionario efetivado!" );  
    }  
  
    void tiraFerias(String mes) {  
        System.out.println(nome + " vai tirar ferias em " + mes);  
    }  
  
    float calculaSalarioAnual() {  
        float salarioAnual = salario * 12;  
        return salarioAnual;  
    }  
}
```

Sobrecarga de métodos

```
public class Funcionario {  
  
    int idade;  
    int cpf;  
    float salario;  
    String nome;  
  
    public Funcionario() {  
        System.out.println("Funcionario efetivado!");  
    }  
  
    float calculaSalarioAnual() {  
        float salarioAnual = salario * 12;  
        return salarioAnual;  
    }  
  
    float calculaSalarioAnual(float decimoTerceiro) {  
        float salarioAnual = (salario * 12) + decimoTerceiro;  
        return salarioAnual;  
    }  
  
    void tiraFerias(String mes) {  
        System.out.println(nome + " vai tirar ferias em " + mes);  
    }  
}
```

Assinatura de um método

✓ Nome

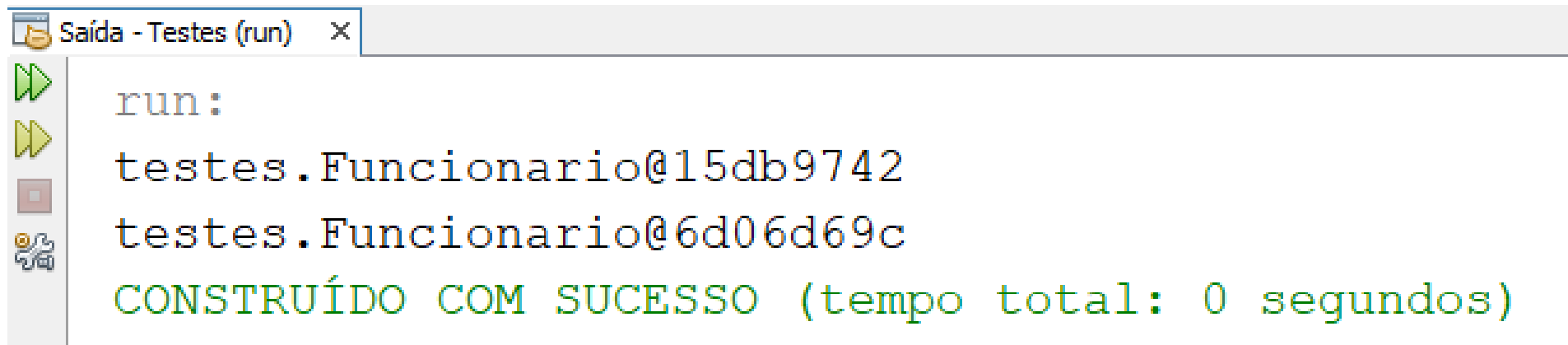
✓ Parâmetro

Sobrecarga:
Diferentes parâmetros

O método toString

O método *toString()* “transforma” de alguma forma a sua classe em uma String, ou seja, ele retorna em uma frase ou palavra o conteúdo da sua classe.

```
System.out.println( f1.toString() );  
System.out.println( f2.toString() );
```

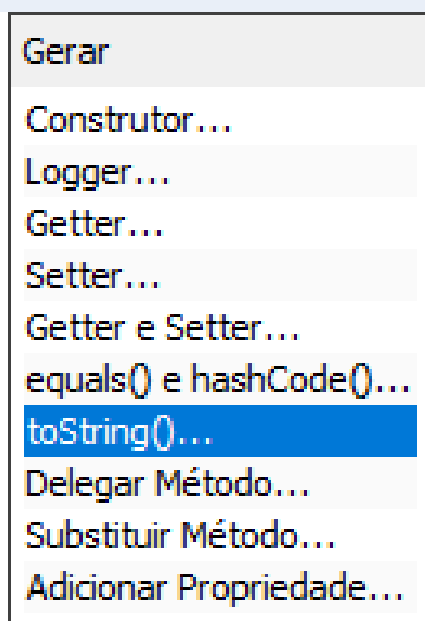


The screenshot shows a window titled "Saída - Testes (run)" with a close button. On the left side of the window, there are four icons: a green play button, a yellow play button, a red square, and a gear icon. The main area of the window displays the following text:

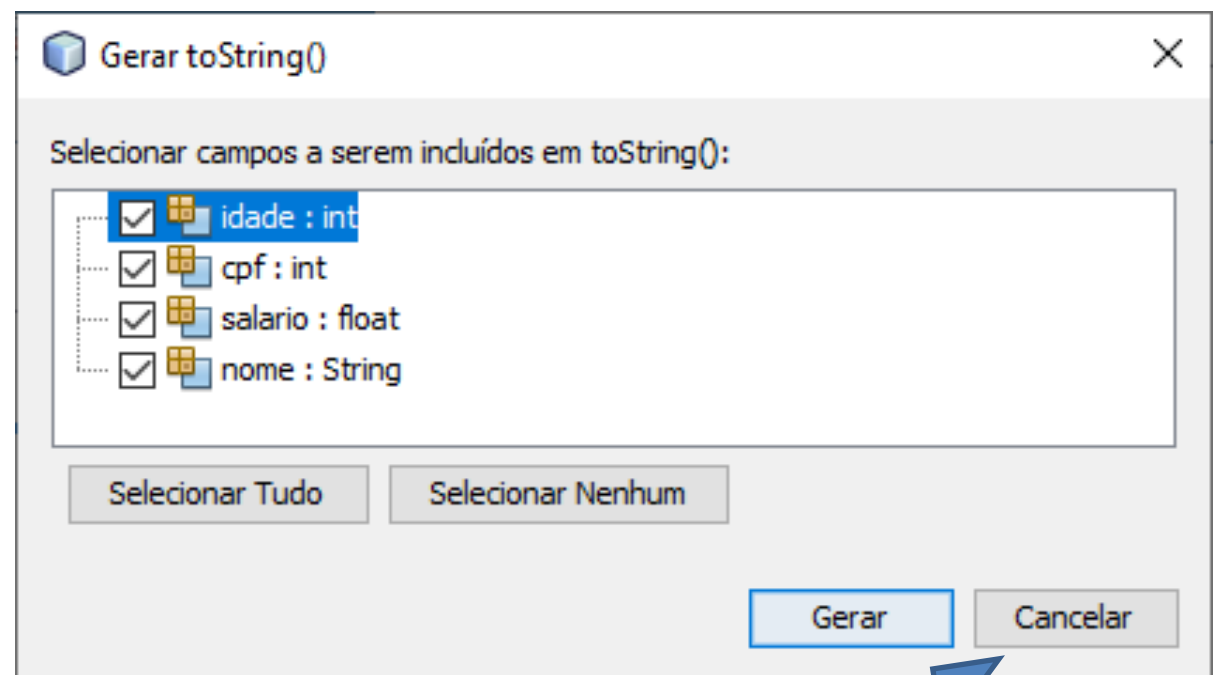
```
run:  
testes.Funcionario@15db9742  
testes.Funcionario@6d06d69c  
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

O método toString

Ao reescrever o método, podemos usá-lo para mostrar os atributos de nossa classe.



atalho ALT + INSERT
(dentro da classe)



Selecionar atributos

O método toString

Por padrão o próprio Java te dá uma sugestão para mostrar os atributos.

```
@Override
public String toString() {
    return "Funcionario{" + "idade=" + idade + ", cpf=" + cpf +
        ", salario=" + salario + ", nome=" + nome + '}';
}
```

Entendendo melhor os tipos de referência (Objetos)

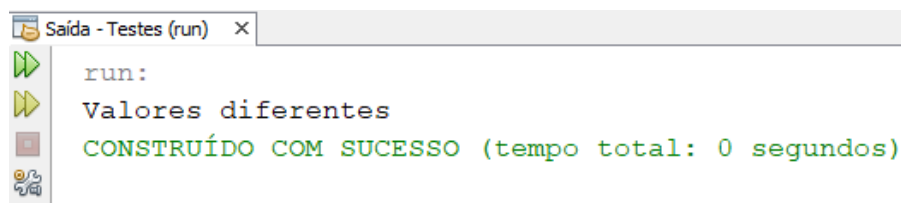
```
// Declarando dois "Objetos"
Funcionario f1;
Funcionario f2;

// Instanciando os dois "Objetos"
f1 = new Funcionario();
f2 = new Funcionario();

// Atribuindo valores aos Objetos
f1.cpf = 123_456_789;
f1.nome = "Samuel Souza";
f1.salario = 400f;

f2.cpf = 987_654_321;
f2.nome = "RenZo Mesquita";
f2.salario = 1000f;

if ( f1 == f2 ) {
    System.out.println("Valores iguais");
} else {
    System.out.println("Valores diferentes");
}
```



```
Saída - Testes (run) X
run:
Valores diferentes
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

```
// Declarando dois "Objetos"
Funcionario f1;
Funcionario f2;

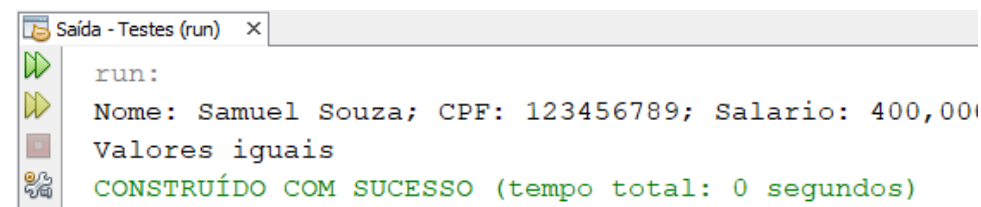
// Instanciando os dois "Objetos"
f1 = new Funcionario();
f2 = new Funcionario();

// Atribuindo valores aos Objetos
f1.cpf = 123_456_789;
f1.nome = "Samuel Souza";
f1.salario = 400f;

f2.cpf = 987_654_321;
f2.nome = "RenZo Mesquita";
f2.salario = 1000f;

f2 = f1;
System.out.printf("Nome: %s; CPF: %d; Salario: %f %n",
    f2.nome, f2.cpf, f2.salario);

if ( f1 == f2 ) {
    System.out.println("Valores iguais");
} else {
    System.out.println("Valores diferentes");
}
```



```
Saída - Testes (run) X
run:
Nome: Samuel Souza; CPF: 123456789; Salario: 400,00
Valores iguais
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

Entendendo melhor os tipos de referência (Objetos)

```
// Declarando dois "Objetos"
Funcionario f1;
Funcionario f2;

// Instanciando os dois "Objetos"
f1 = new Funcionario();
f2 = new Funcionario();

// Atribuindo valores aos Objetos
f1.cpf = 123_456_789;
f1.nome = "Samuel Souza";
f1.salario = 400f;

f2.cpf = 987_654_321;
f2.nome = "RenZo Mesquita";
f2.salario = 1000f;

f1.consulta(f2);

System.out.printf("Nome: %s; CPF: %d; Salario: %f %n",
    f2.nome, f2.cpf, f2.salario);

if ( f1 == f2 ) {
    System.out.println("Valores iguais");
} else {
    System.out.println("Valores diferentes");
}
```

```
public class Funcionario {

    int idade;
    int cpf;
    float salario;
    String nome;

    void consulta(Funcionario f) {
        f.cpf = cpf;
        f.nome = nome;
        f.salario = salario;
        f.idade = idade;
    }

    float calculaSalarioAnual() {
        float salarioAnual = salario * 12;
        return salarioAnual;
    }
}
```

Qual a saída?

Entendendo melhor os tipos de referência (Objetos)

```
Saída - Testes (run) x
run:
Nome: Samuel Souza; CPF: 123456789; Salario: 400,00
Valores diferentes
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

```
// Atribuindo valores aos Objetos
```

```
f1.cpf = 123_456_789;
f1.nome = "Samuel Souza";
f1.salario = 400f;
```

```
f2.cpf = 987_654_321;
f2.nome = "RenZo Mesquita";
f2.salario = 1000f;
```

```
f1.consulta(f2);
```

Passagem de f2 como referência

```
System.out.printf("Nome: %s; CPF: %d; Salario: %f %n",
    f2.nome, f2.cpf, f2.salario);
```

```
if ( f1 == f2 ) {
    System.out.println("Valores iguais");
} else {
    System.out.println("Valores diferentes");
}
```

```
void consulta(Funcionario f) {
    f.cpf = cpf;
    f.nome = nome;
    f.salario = salario;
    f.idade = idade;
}
```


Exercícios

1. Implemente uma classe Calculadora e crie métodos que realizem as seguintes operações: Soma, Subtração, Divisão e Multiplicação.
2. Você foi contratado para implementar o controle de estoque em uma pequena fábrica de peças de automóveis, que é responsável pela montagem dos motores de diversos modelos de carros. Neste primeiro momento o projeto irá entregar apenas o cadastro dos materiais que contém as seguintes informações: código de série, código do material, nome do material, categoria do material e quantidade. Crie no mínimo três objetos diferentes, preencha seus atributos e mostre na saída os valores atribuídos.

Exercícios

3. Crie uma aplicação em Java para gerenciar carros em geral.
Cada carro **contém** *uma cor, marca, modelo, velocidade máxima e velocidade atual*. Os carros também **possuem** *motor*, que por sua vez tem como **atributo** *potência e tipo*.
Os carros podem ser *ligados* e também podem *acelerar*.
Sobrescreva o método *toString()* para mostrar as informações atuais do carro.
- Crie no mínimo três objetos diferentes e preencha seus atributos.
- Chame todos os métodos para todos os objetos criados.

Exercícios

4. O departamento de Marketing gostaria de fazer um controle das campanhas que são vinculadas pela empresa, para isso foi solicitada a criação de um software que faça o cadastro de todas as propagandas lançadas.

As informações para controle são: código de identificação, nome do projeto, título da campanha, custo e autor.

Geralmente os autores das campanhas são empresas contratadas e para estes casos devemos armazenar o nome da empresa, CNPJ, contato e telefone.

Obrigado!