

BLOATERS

# BLOATERS

Código muito  
grande

Acúmulo ao  
longo do  
tempo

Dificuldade  
de mexer no  
código

# ALGUNS INDICATIVOS

**Métodos grandes**

**Classe grande**

**Obsessão  
primitiva**

**Lista de  
parâmetros longa**

**Grupos de dados**

## Métodos grandes

Sempre  
acrescenta algo

Nunca tira nada

Mais fácil  
escrever do que  
ler

Só percebe o  
problema quando  
virou um monstro

## Métodos grandes

Sempre  
acrescenta algo

Nunca tira nada

Mais fácil  
escrever do que  
ler

Só percebe o  
problema quando  
virou um monstro

Se você precisa  
comentar em algo,  
talvez isso  
deveria estar em  
outro lugar

**Métodos grandes**

**Extraire método**

Métodos grandes

Extrair método

```
def printBill(self):  
    self.printMessage()  
  
    # print details  
    print("name:", self.name)  
    print("amount:", self.getQuantity())
```

## Métodos grandes

## Extrair método

```
def printBill(self):  
    self.printMessage()  
  
    # print details  
    print("name:", self.name)  
    print("amount:", self.getQuantity())
```

```
def printBill(self):  
    self.printMessage()  
    self.printDetails(self.getQuantity())  
  
def printDetails(self, quantity):  
    print("name:", self.name)  
    print("amount:", quantity)
```



## Métodos grandes

## Extrair método

```
def printBill(self):  
    self.printMessage()  
  
    # print details  
    print("name:", self.name)  
    print("amount:", self.getQuantity())
```

```
def printBill(self):  
    self.printMessage()  
    self.printDetails(self.getQuantity())  
  
def printDetails(self, quantity):  
    print("name:", self.name)  
    print("amount:", quantity)
```

Mais legível

Menos duplicado e  
mais reuso

Isola partes  
independentes

**Métodos grandes**

**Extrair método**

**Criar  
novo  
método**

**Métodos grandes**

**Extrair método**

**Criar  
novo  
método**

**Coloca o  
código no  
novo método**

**Métodos grandes**

**Extrair método**

**Criar  
novo  
método**

**Coloca o  
código no  
novo método**

Se as variáveis  
não são usadas  
fora desse pedaço  
de código, serão  
variáveis locais  
dele do método

**Métodos grandes**

**Extrair método**

**Criar  
novo  
método**

**Coloca o  
código no  
novo método**

**Métodos grandes**

**Extrair método**

**Criar  
novo  
método**

**Coloca o  
código no  
novo método**

**Chama o  
método**

**Métodos grandes**

**Extrair método**

**Criar  
novo  
método**

**Coloca o  
código no  
novo método**

**Chama o  
método**

Se as variáveis  
forem usadas  
anteriormente,  
deve, ser passadas  
como parâmetro

**Métodos grandes**

**Extrair método**

**Criar  
novo  
método**

**Coloca o  
código no  
novo método**

**Chama o  
método**



**Métodos grandes**

**Extrair método**

**Criar  
novo  
método**

**Coloca o  
código no  
novo método**

**Chama o  
método**

**Retorna se  
necessário**

**Métodos grandes**

**Extrair método**

**Criar  
novo  
método**

**Coloca o  
código no  
novo método**

**Chama o  
método**

**Retorna se  
necessário**

Se a variável  
assume um novo  
valor, talvez deve  
retornar.  
Verifique

Métodos grandes

Extrair método

Criar  
novo  
método

Coloca o  
código no  
novo método

Chama o  
método

Retorna se  
necessário

Se as variáveis  
não são usadas  
fora desse pedaço  
de código, serão  
variáveis locais  
dele do método

Se as variáveis  
forem usadas  
anteriormente,  
deve, ser passadas  
como parâmetro

Se a variável  
assume um novo  
valor, talvez deve  
retornar.  
Verifique

Métodos grandes

Trocar variável  
temporária por  
consulta

Métodos grandes

Trocar variável  
temporária por  
consulta

```
def calculateTotal():  
    basePrice = quantity * itemPrice  
    if basePrice > 1000:  
        return basePrice * 0.95  
    else:  
        return basePrice * 0.98
```

Métodos grandes

Trocar variável  
temporária por  
consulta

```
def calculateTotal():  
    basePrice = quantity * itemPrice  
    if basePrice > 1000:  
        return basePrice * 0.95  
    else:  
        return basePrice * 0.98
```

```
def calculateTotal():  
    if basePrice() > 1000:  
        return basePrice() * 0.95  
    else:  
        return basePrice() * 0.98  
  
def basePrice():  
    return quantity * itemPrice
```

Métodos grandes

Trocar variável  
temporária por  
consulta

Mais legível

Menos duplicado e  
mais reuso

Cuidado com a  
performance

```
def calculateTotal():  
    basePrice = quantity * itemPrice  
    if basePrice > 1000:  
        return basePrice * 0.95  
    else:  
        return basePrice * 0.98
```

```
def calculateTotal():  
    if basePrice() > 1000:  
        return basePrice() * 0.95  
    else:  
        return basePrice() * 0.98  
  
def basePrice():  
    return quantity * itemPrice
```

Métodos grandes

Trocar variável  
temporária por  
consulta

Verifique se a  
variável só  
recebe esse  
valor

Use  
Extrair  
método

Troque a  
variável  
pela chamada  
do método

O método deve  
apenas retornar,  
não alterar o  
valor



**Métodos grandes**

**Trocar lista de  
parâmetros por  
objeto**

Métodos grandes

Trocar lista de  
parâmetros por  
objeto

Customer
amountInvoicedIn (start : Date, end : Date) amountReceivedIn (start : Date, end : Date) amountOverdueIn (start : Date, end : Date)

Métodos grandes

Trocar lista de  
parâmetros por  
objeto

Customer
amountInvoicedIn (start : Date, end : Date) amountReceivedIn (start : Date, end : Date) amountOverdueIn (start : Date, end : Date)

Customer
amountInvoicedIn (date : DateRange) amountReceivedIn (date : DateRange) amountOverdueIn (date : DateRange)

Métodos grandes

Trocar lista de  
parâmetros por  
objeto

Mais legível

Menos duplicado e  
mais reuso

Pode isolar os  
métodos do objeto

### Customer

amountInvoicedIn (start : Date, end : Date)  
amountReceivedIn (start : Date, end : Date)  
amountOverdueIn (start : Date, end : Date)

### Customer

amountInvoicedIn (date : DateRange)  
amountReceivedIn (date : DateRange)  
amountOverdueIn (date : DateRange)

**Métodos grandes**

**Trocar lista de  
parâmetros por  
objeto**

**Criar classe  
que representa  
o grupo**

**Métodos grandes**

**Trocar lista de  
parâmetros por  
objeto**

**Criar classe  
que representa  
o grupo**

**Adicione o  
parâmetro**

**Métodos grandes**

**Trocar lista de  
parâmetros por  
objeto**

**Criar classe  
que representa  
o grupo**

**Adicione o  
parâmetro**

**Deletar  
parâmetros  
antigos**

**Métodos grandes**

**Trocar lista de  
parâmetros por  
objeto**

**Criar classe  
que representa  
o grupo**

**Adicione o  
parâmetro**

**Deletar  
parâmetros  
antigos**

**Testar após  
a troca**



**Métodos grandes**

**Preservar objetos**

**Métodos grandes**

**Preservar objetos**

```
low = daysTempRange.getLow()  
high = daysTempRange.getHigh()  
withinPlan = plan.withinRange(low, high)
```

**Métodos grandes**

**Preservar objetos**

```
low = daysTempRange.getLow()  
high = daysTempRange.getHigh()  
withinPlan = plan.withinRange(low, high)
```

```
withinPlan = plan.withinRange(daysTempRange)
```

**Métodos grandes**

**Preservar objetos**

```
low = daysTempRange.getLow()  
high = daysTempRange.getHigh()  
withinPlan = plan.withinRange(low, high)
```

Mais compreensível

Se precisar de mais dados

**Cuidado com a inflexibilidade (interface)**

```
withinPlan = plan.withinRange(daysTempRange)
```

**Métodos grandes**

**Preservar objetos**

**Criar objeto  
para passar  
como parâmetro**

**Métodos grandes**

**Preservar objetos**

**Criar objeto  
para passar  
como parâmetro**

**Remover  
parâmetros  
antigos chamando  
os métodos**

**Métodos grandes**

**Preservar objetos**

**Criar objeto  
para passar  
como parâmetro**

**Remover  
parâmetros  
antigos chamando  
os métodos**

**Deletar  
variáveis e  
getters**

**Métodos grandes**

**Decomp  
condicionais**



Métodos grandes

Decompor  
condicionais

```
if date.before(SUMMER_START) or date.after(SUMMER_END):  
    charge = quantity * winterRate + winterServiceCharge  
else:  
    charge = quantity * summerRate
```

## Métodos grandes

## Decompor condicionais

```
if date.before(SUMMER_START) or date.after(SUMMER_END):  
    charge = quantity * winterRate + winterServiceCharge  
else:  
    charge = quantity * summerRate
```

```
if notSummer(date):  
    charge = winterCharge(quantity)  
else:  
    charge = summerCharge(quantity)
```

Métodos grandes

Decompor  
condicionais

Mais legível

Nomes claros de  
métodos

```
if date.before(SUMMER_START) or date.after(SUMMER_END):  
    charge = quantity * winterRate + winterServiceCharge  
else:  
    charge = quantity * summerRate
```

```
if notSummer(date):  
    charge = winterCharge(quantity)  
else:  
    charge = summerCharge(quantity)
```

**Métodos grandes**

**Decompor  
condicionais**

**Usar extrair  
método**

**Verificar no  
if e else**

**Métodos grandes**

**E a  
performance?**

# ALGUNS INDICATIVOS

**Métodos grandes**

**Classe grande**

**Obsessão  
primitiva**

**Lista de  
parâmetros longa**

**Grupos de dados**

## Classes grandes

Sempre  
acrescenta algo

Nunca tira nada

Mais fácil  
escrever do que  
ler

Só percebe o  
problema quando  
virou um monstro

Classes grandes

Extraire classe



Classes grandes

Extraire classe

```
class Person:  
    name = "Harry Potter"  
    phone = 5551234  
    phoneArea = 10  
  
    def getTelephoneNumber():  
        ...
```

Classes grandes

Extraire classe

```
class Person:
    name = "Harry Potter"
    phone = 5551234
    phoneArea = 10

    def getTelephoneNumber():
        ...
```

```
class Person:
    name = "Harry Potter"

class Phone:
    phone = 5551234
    phoneArea = 10

    def getTelephoneNumber():
```

Classes grandes

Extrair classe

Mais legível

Nomes claros de  
métodos

```
class Person:  
    name = "Harry Potter"  
    phone = 5551234  
    phoneArea = 10  
  
    def getTelephoneNumber():  
        ...
```

```
class Person:  
    name = "Harry Potter"  
  
class Phone:  
    phone = 5551234  
    phoneArea = 10  
  
    def getTelephoneNumber():
```

Classes grandes

Extrair classe

Criar nova  
classe

**Classes grandes**

**Extrair classe**

**Criar nova  
classe**

**Mover métodos  
privados**

**Classes grandes**

**Extrair classe**

**Criar nova  
classe**

**Mover métodos  
privados**

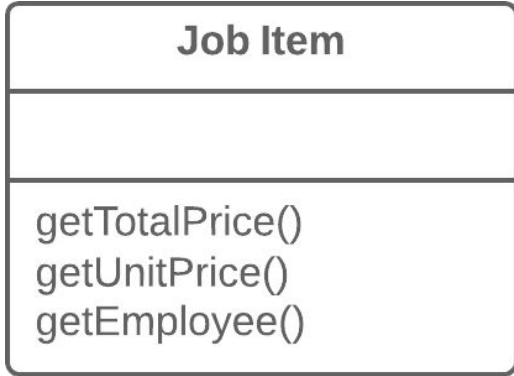
**Mover métodos  
públicos**

**Classes grandes**

**Extraire sous-classe**

Classes grandes

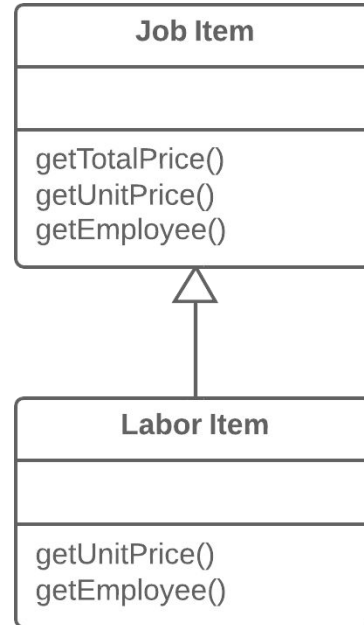
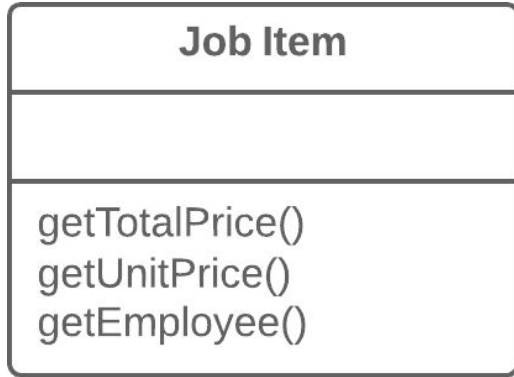
Extraire sous-classe





Classes grandes

Extraire sous-classe



**Classes grandes**

**Extrair subclasse**

**Criar nova  
classe**

Classes grandes

Extrair subclasse

Criar nova  
classe

Criar construtor

Classes grandes

Extrair subclasse

Criar nova  
classe

Criar construtor

**Classes grandes**

**Extrair subclasse**

**Criar nova  
classe**

**Criar construtor**

**Mover métodos  
públicos**

Classes grandes

Extrair subclasse

Criar nova  
classe

Criar construtor

Mover métodos  
públicos

Chamar construtor  
do pai

# ALGUNS INDICATIVOS

Métodos grandes

Classe grande

Obsessão  
primitiva

Lista de  
parâmetros longa

Grupos de dados

## Obsessão primitiva

Uso excessivo de  
tipos primitivos

Uso excessivo de  
constantes

Primitivos são  
mais simples que  
criar classes

Usados para  
simular tipo



**Obsessão  
primitiva**

**Substituir valor  
por classe**

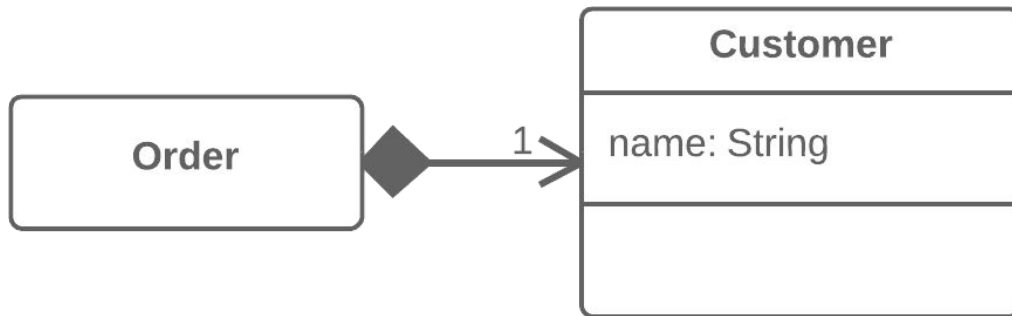
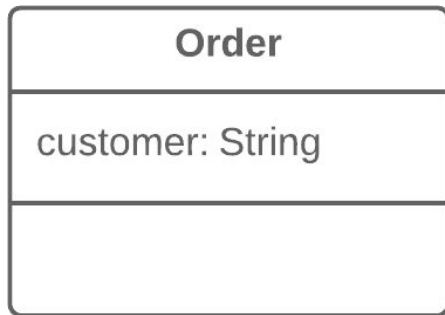
**Obsessão  
primitiva**

**Substituir valor  
por classe**

Order
customer: String

**Obsessão  
primitiva**

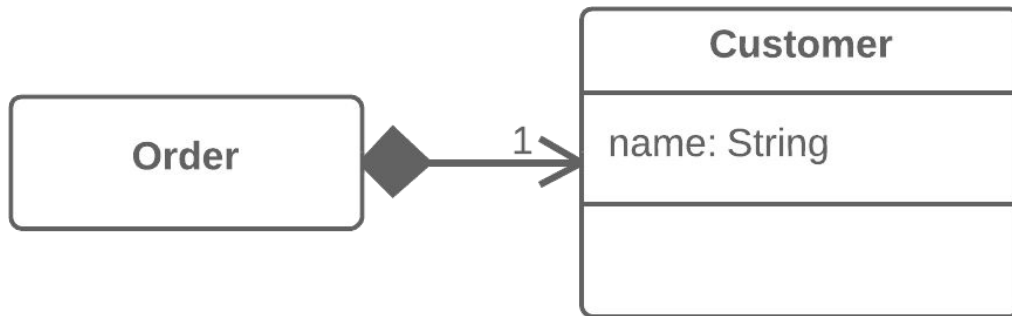
**Substituir valor  
por classe**



**Obsessão  
primitiva**

**Substituir valor  
por classe**

**Melhor  
escalabilidade**



**Obsessão  
primitiva**

**Substituir valor  
por classe**

**Criar nova  
classe**

**Obsessão  
primitiva**

**Substituir valor  
por classe**

**Criar nova  
classe**

**Criar getters e  
setters**

**Obsessão  
primitiva**

**Substituir valor  
por classe**

**Criar nova  
classe**

**Criar getters e  
setters**

**Mudar variável  
anterior**

**Obsessão  
primitiva**

**Substituir array  
por classe**



Obsessão  
primitiva

Substituir array  
por classe

```
row = [None * 2]  
row[0] = "Liverpool"  
row[1] = "15"
```

Obsessão  
primitiva

Substituir array  
por classe

```
row = [None * 2]  
row[0] = "Liverpool"  
row[1] = "15"
```

```
row = Performance();  
row.setName("Liverpool")  
row.setWins("15")
```

Obsessão  
primitiva

Substituir array  
por classe

Comportamento no  
objeto

Nomes claros de  
métodos

```
row = [None * 2]
row[0] = "Liverpool"
row[1] = "15"
```

```
row = Performance();
row.setName("Liverpool")
row.setWins("15")
```

**Obsessão  
primitiva**

**Substituir array  
por classe**

**Criar nova  
classe**

**Obsessão  
primitiva**

**Substituir valor  
por classe**

**Criar nova  
classe**

**Coloca o array  
como público na  
classe**

**Obsessão  
primitiva**

**Substituir valor  
por classe**

**Criar nova  
classe**

**Coloca o array  
como público na  
classe**

**Receber o  
objeto no local  
original**

**Obsessão  
primitiva**

**Substituir valor  
por classe**

**Criar nova  
classe**

**Coloca o array  
como público na  
classe**

**Receber o  
objeto no local  
original**

**Criar métodos  
de acesso  
públicos**

**Obsessão  
primitiva**

**Substituir valor  
por classe**

**Criar nova  
classe**

**Coloca o array  
como público na  
classe**

**Receber o  
objeto no local  
original**

**Criar métodos  
de acesso  
públicos**

**Coloca o array  
como privado**



**Obsessão  
primitiva**

**Substituir valor  
por classe**

**Criar nova  
classe**

**Coloca o array  
como público na  
classe**

**Receber o  
objeto no local  
original**

**Criar métodos  
de acesso  
públicos**

**Coloca o array  
como privado**

**Criar campos  
para cada  
elemento do  
array**

**Obsessão  
primitiva**

**Substituir valor  
por classe**

**Criar nova  
classe**

**Coloca o array  
como público na  
classe**

**Receber o  
objeto no local  
original**

**Criar métodos  
de acesso  
públicos**

**Coloca o array  
como privado**

**Criar campos  
para cada  
elemento do  
array**

**Mudar métodos  
de acesso para  
o array**

**Obsessão  
primitiva**

**Substituir valor  
por classe**

**Criar nova  
classe**

**Coloca o array  
como público na  
classe**

**Receber o  
objeto no local  
original**

**Criar métodos  
de acesso  
públicos**

**Coloca o array  
como privado**

**Criar campos  
para cada  
elemento do  
array**

**Mudar métodos  
de acesso para  
o array**

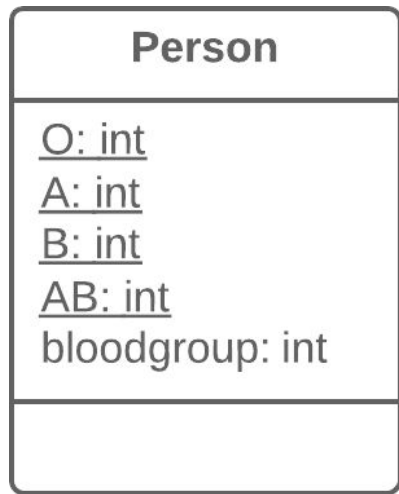
**Deletar o array**

**Obsessão  
primitiva**

**Substituir tipo  
por classe**

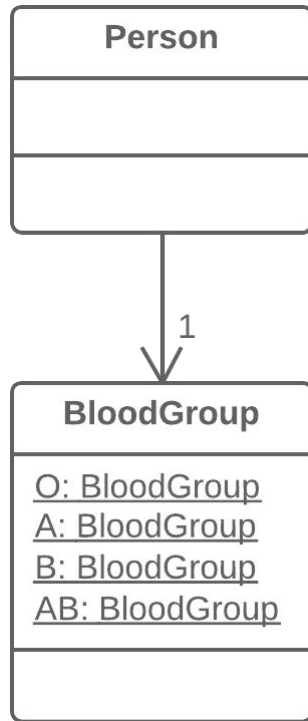
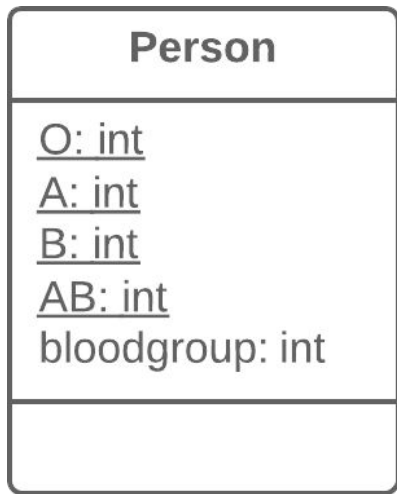
Obsessão  
primitiva

Substituir tipo  
por classe



Obsessão  
primitiva

Substituir tipo  
por classe



Comportamento no  
objeto

Nomes claros de  
classes

**Obsessão  
primitiva**

**Substituir tipo  
por classe**

**Criar nova  
classe**

**Obsessão  
primitiva**

**Substituir tipo  
por classe**

**Criar nova  
classe**

**Mover campos  
para a nova  
classe  
(privados)**



**Obsessão  
primitiva**

**Substituir tipo  
por classe**

**Criar nova  
classe**

**Mover campos  
para a nova  
classe  
(privados)**

**Criar getters e  
constructor**

**Obsessão  
primitiva**

**Substituir tipo  
por classe**

**Criar nova  
classe**

**Mover campos  
para a nova  
classe  
(privados)**

**Criar getters e  
constructor**

**Criar métodos  
de estático  
para os campos**

**Obsessão  
primitiva**

**Substituir tipo  
por classe**

**Criar nova  
classe**

**Mover campos  
para a nova  
classe  
(privados)**

**Criar getters e  
constructor**

**Criar métodos  
de estático  
para os campos**

**Trocar o uso na  
classe original**

**Obsessão  
primitiva**

**Substituir tipo  
por classe**

**Criar nova  
classe**

**Mover campos  
para a nova  
classe  
(privados)**

**Criar getters e  
constructor**

**Criar métodos  
de estático  
para os campos**

**Trocar o uso na  
classe original**

**Remover  
constantes de  
tipo**

**Obsessão  
primitiva**

**Trocar lista de  
parâmetros por  
objeto**

**Preservar objetos**

# ALGUNS INDICATIVOS

Métodos grandes

Classe grande

Obsessão  
primitiva

Lista de  
parâmetros longa

Grupos de dados

## Lista de parâmetros longa

Vários  
algoritmos em um  
único método

Tentativa de  
classes serem  
mais  
independentes

Mais difícil de  
usar ao crescer

Mais difícil de  
usar ao crescer

Lista de  
parâmetros longa

Trocar parâmetro  
por chamada de  
método

```
basePrice = quantity * itemPrice  
seasonalDiscount = this.getSeasonalDiscount()  
fees = this.getFees()  
finalPrice = discountedPrice(basePrice, seasonalDiscount, fees)
```

```
basePrice = quantity * itemPrice  
finalPrice = discountedPrice(basePrice)
```



**Lista de  
parâmetros longa**

**Trocar parâmetro  
por chamada de  
método**

**Alterar  
método com  
muitos  
parâmetros**

**Utilizando  
métodos de  
acesso  
anteriores**