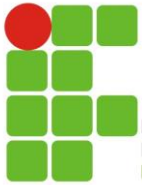


INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

Programação Orientada a Objetos em Jogos

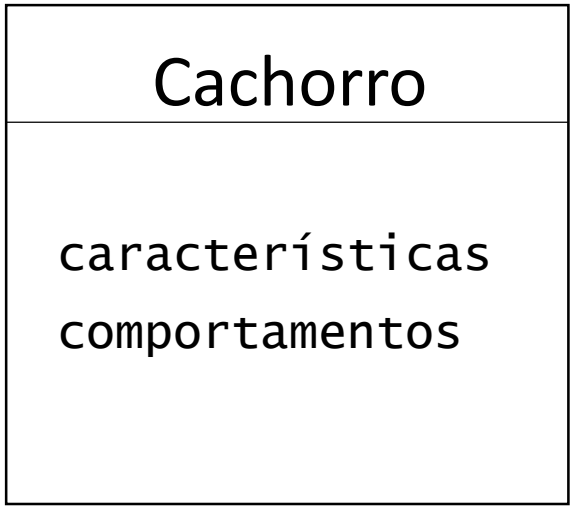
Marcelo de Barros Barbosa

COMO NÓS VAMOS ENSINAR
AO COMPUTADOR O QUE É UM OBJETO?

COMO DIZEMOS QUE UM OBJETO EXISTE?

com estruturas

ESTRUTURA PARA UM OBJETO


Cachorro
características
comportamentos

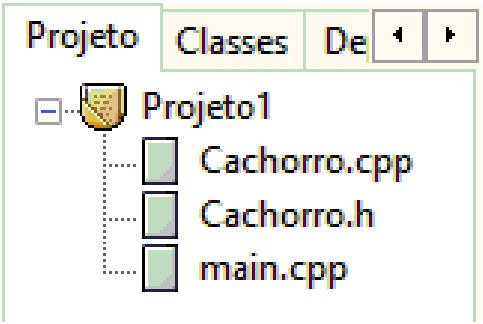
PRIMEIRO VAMOS CONFIGURAR NOSSO PROJETO

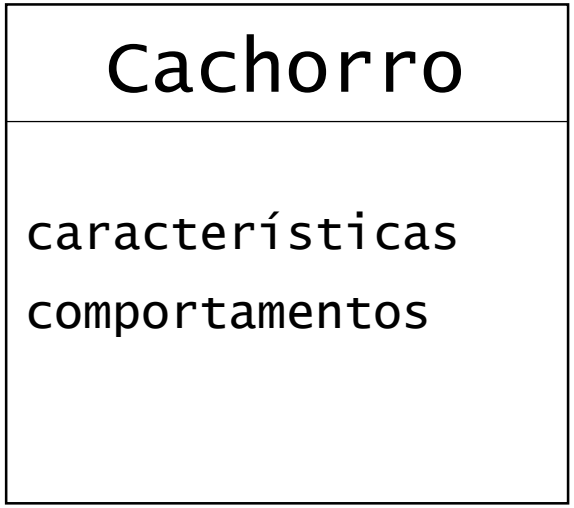
No Dev-C++:

1) Arquivo -> Novo -> Projeto -> Console Application

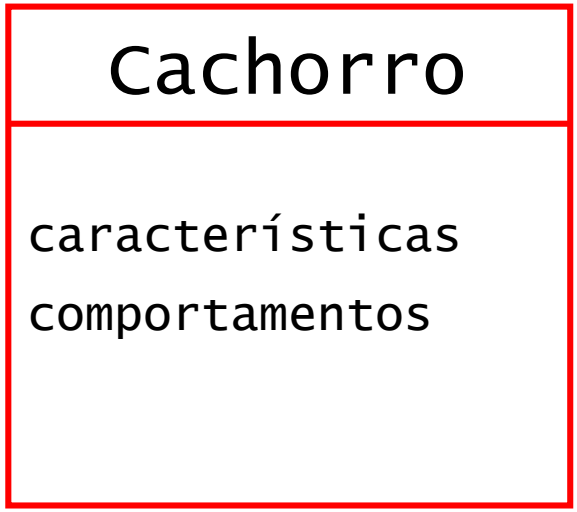
2) Botão Direito no nome do Projeto -> New File

Criar 3 arquivos




Cachorro
características
comportamentos

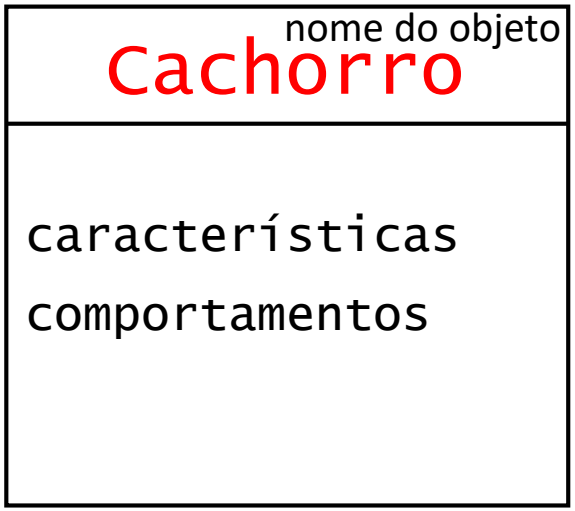
classe



Cachorro.h

```
class          {




};
```

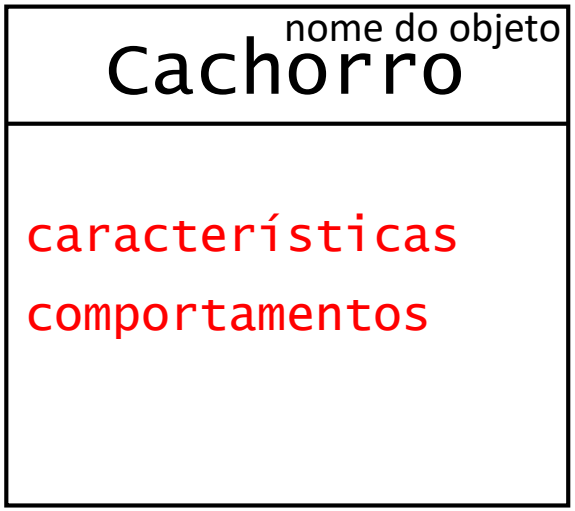
classe



Cachorro.h

```
class Cachorro {

};
```

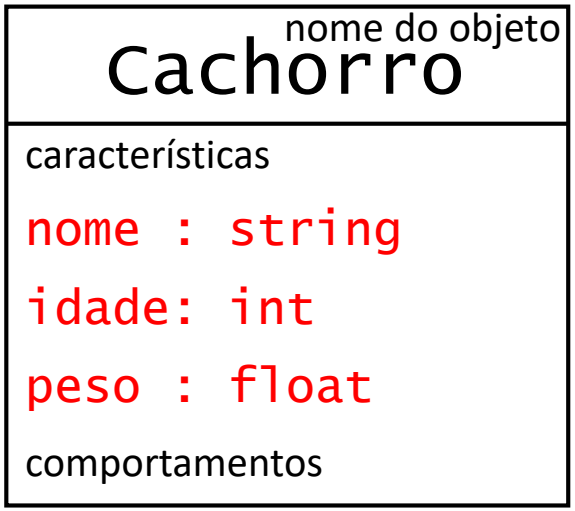
classe



Cachorro.h

```
class Cachorro {

};
```

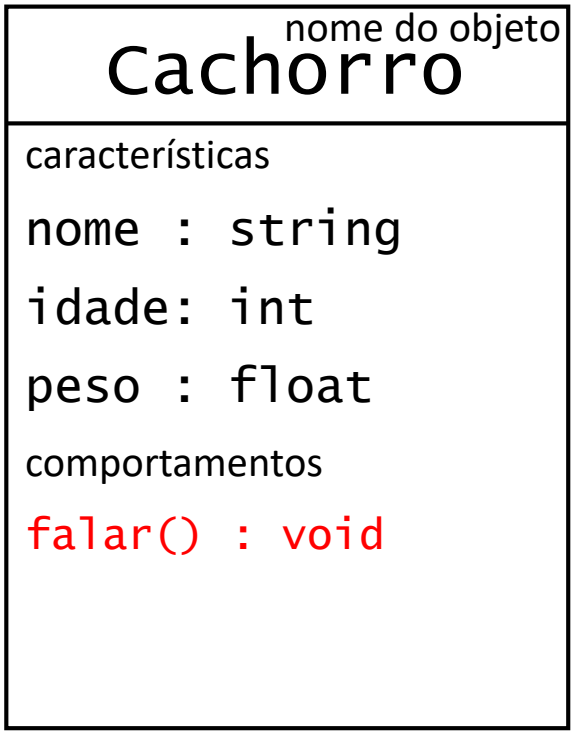
classe



Cachorro.h

```
class Cachorro {

    string nome;
    int idade;
    float peso;


};
```

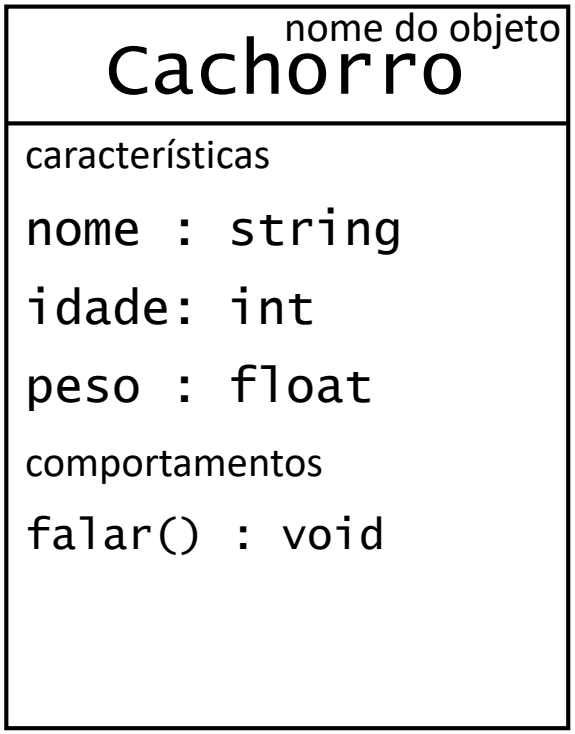
classe



Cachorro.h

```
class Cachorro {

    string nome;
    int idade;
    float peso;

    void falar();
};
```

classe



Cachorro.h

```
#include <string>

class Cachorro {

    string nome;
    int idade;
    float peso;

    void falar();
};
```

CRIAMOS A ESTRUTURA DE UM OBJETO

AGORA VAMOS DIZER O QUE ELE DEVE FAZER

classe

nome do objeto

Cachorro

características

nome : string

idade: int

peso : float

comportamentos

falar() : void

Cachorro.cpp

```
#include "Cachorro.h“
#include <iostream>
using namespace std;

void Cachorro::falar()
{
    cout << "AU AU" << endl;
};
```

QUAL A DIFERENÇA DO .h PARA O .cpp?

PARA QUE SERVE O .h?

PARA QUE SERVE O .cpp?

OK AGORA VAMOS CRIAR NOSSO OBJETO!

queremos um cachorro que fala

Main.cpp

```
#include <iostream>

int main() {




    return 0;
}
```

O QUE VOCÊ ACHA QUE DEVEMOS DIZER PARA O COMPUTADOR?

Main.cpp

```
#include <iostream>

int main() {

        Cachorro  novo;
        novo.falar();


        return 0;
}
```

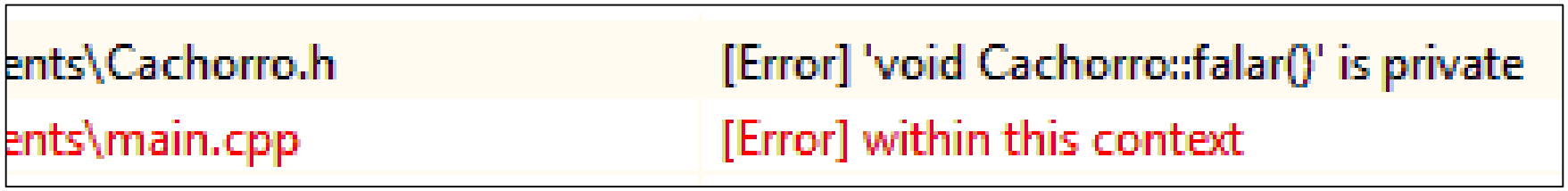
Main.cpp

```
#include <iostream>
#include “Cachorro.h”

int main() {

        Cachorro  novo;
        novo.falar();


        return 0;
}
```

NÓS SEMPRE INCLUÍMOS O .h

quando vamos usar um objeto, só precisamos saber a sua estrutura e não como ele funciona

11	8	C:\Users\marcelo\Documents\Cachorro.h	[Error] 'void Cachorro::falar()' is private
10	13	C:\Users\marcelo\Documents\main.cpp	[Error] within this context

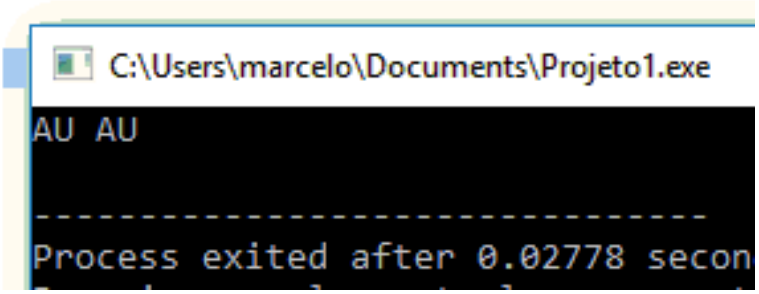


LEMBRA DO NOSSO ENCAPSULAMENTO?

tudo começa como privado

Cachorro.h

```
#include <string>

class Cachorro {

public:
    string nome;
    int idade;
    float peso;

    void falar();
};
```


C:\Users\marcelo\Documents\Projeto1.exe
AU AU

Process exited after 0.02778 secon

ENTENDENDO NOSSO CÓDIGO

ENCAPSULAMENTO

private (Privado)

Informações só podem ser *vistas* pelo próprio Objeto.

public (Públicos)

Informações podem ser *vistas* por qualquer um que interagir com o Objeto.

MELHORANDO NOSSO CÓDIGO

Cachorro.h

```
#include <string>

class Cachorro {

public:
    string nome;
    int idade;
    float peso;

    void falar();
} ;
```

Cachorro.cpp

```
#include "Cachorro.h“
#include <iostream>
using namespace std;

void Cachorro::falar()
{
    cout << "AU AU" << endl;
} ;
```

Cachorro.h

```
#include <string>
#include <iostream>
using namespace std;

class Cachorro {

public:
    string nome;
    int idade;
    float peso;

    void falar();
} ;
```

Cachorro.cpp

```
#include "Cachorro.h“

void Cachorro::falar()
{
    cout << "AU AU" << endl;
} ;
```

Recapitulando,

ONDE DIZEMOS O QUE VAMOS USAR, NO .h OU NO .cpp?

PARA QUE SERVE UM #include?

ENTÃO ONDE COLOCAMOS NOSSOS #include?

sempre no .h!

Cachorro.h

```
#include <string>
#include <iostream>
using namespace std;

class Cachorro {

public:
    string nome;
    int idade;
    float peso;

    void falar();
} ;
```

Cachorro.cpp

```
#include "Cachorro.h“

void Cachorro::falar()
{
    cout << "AU AU" << endl;
} ;
```

EXERCÍCIO

I. Crie uma Classe chamada Pessoa, para um objeto que possui as seguintes características e comportamentos:

Pessoa
nome : string endereco : string
mostrar() : void

mostrar () :
"Oi, me chamo %nome
moro em %endereco"

II. Crie uma Classe chamada Gato:

falar () : "MIAU MIAU"

Gato
nome : string idade : int peso : float
falar() : void

QUAIS SÃO OS VALORES PARA
nome, idade E peso?

Cachorro.h

```
#include <string>
#include <iostream>
using namespace std;

class Cachorro {

public:
    Cachorro(string n, int i, float p);

    string nome;
    int idade;
    float peso;

    void falar();
};
```

Cachorro.cpp

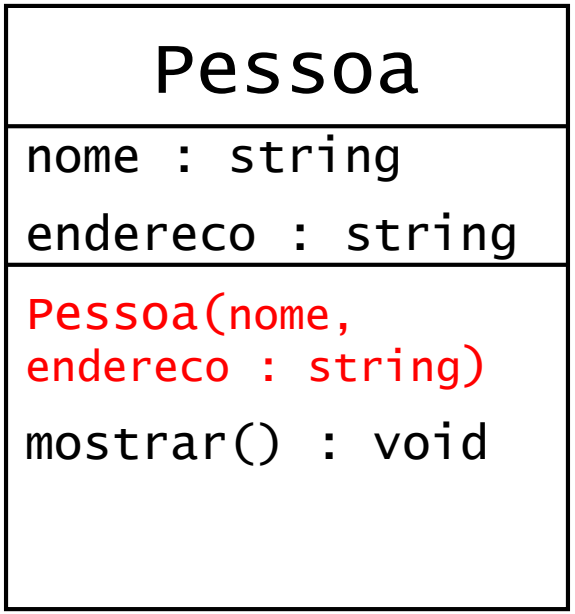
```
#include "Cachorro.h“

Cachorro::Cachorro(string n, int i,
float p)
{
    nome = n;
    idade = i;
    peso = p;
};

void Cachorro::falar()
{
    cout << "AU AU" << endl;
};
```

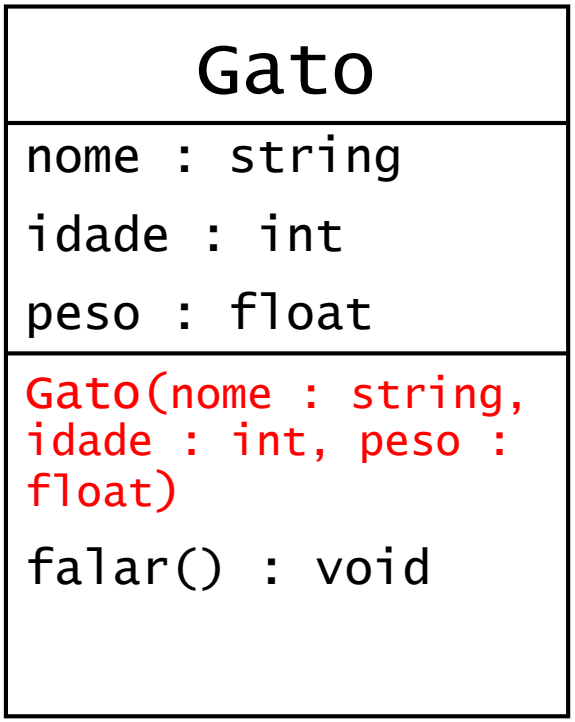
Main.cpp

```
#include <iostream>
#include “Cachorro.h”

int main() {

        Cachorro   novo(“Hulk”, 2, 9.9);
        novo.falar();

        std::cout << “Dog” << endl;
        std::cout << "Nome: " << novo.nome << " Idade: " << novo.idade;
        std::cout << " Peso: " << novo.peso << endl;


        return 0;
}
```

EXERCÍCIO

1. Fazer a mesma coisa para os objetos criados no exercício anterior: Gato e Pessoa.





O QUE NÓS ACABAMOS DE FAZER?

ISSO SE CHAMA Construtor DE UM OBJETO

pois estamos construindo ele com valores determinados

Main.cpp

```
#include <iostream>
#include “Cachorro.h”

int main() {

        // Cachorro  novo(“Hulk”, 2, 9.9);
        Cachorro novo;
        novo.falar();

        std::cout << “Dog” << endl;
        std::cout << "Nome: " << novo.nome << " Idade: " << novo.idade;
        std::cout << " Peso: " << novo.peso << endl;


        return 0;
}
```

Cachorro.h

```
#include <string>
#include <iostream>
using namespace std;

class Cachorro {

public:
    Cachorro();
    Cachorro(string n, int i, float p);

    string nome;
    int idade;
    float peso;

    void falar();
};
```

Cachorro.cpp

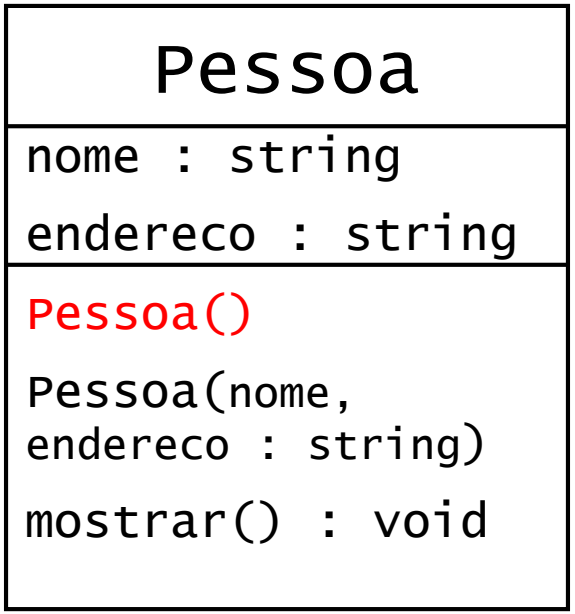
```
#include "Cachorro.h“

Cachorro::Cachorro()
{
    nome = "Padrao";
    idade = 0;
    peso = 99.99;
};

Cachorro::Cachorro(string n, int i,
float p)
{
    nome = n;
    idade = i;
    peso = p;
};

void Cachorro::falar()
{
    cout << "AU AU" << endl;
```

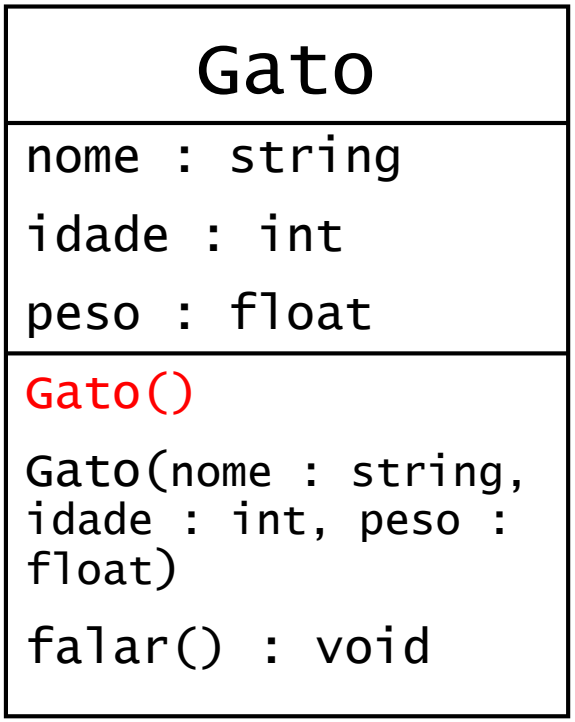
SE TEMOS O Construtor COM VALORES Padrão,

ENTÃO CHAMAMOS ISSO DE UM ... ?

EXERCÍCIO

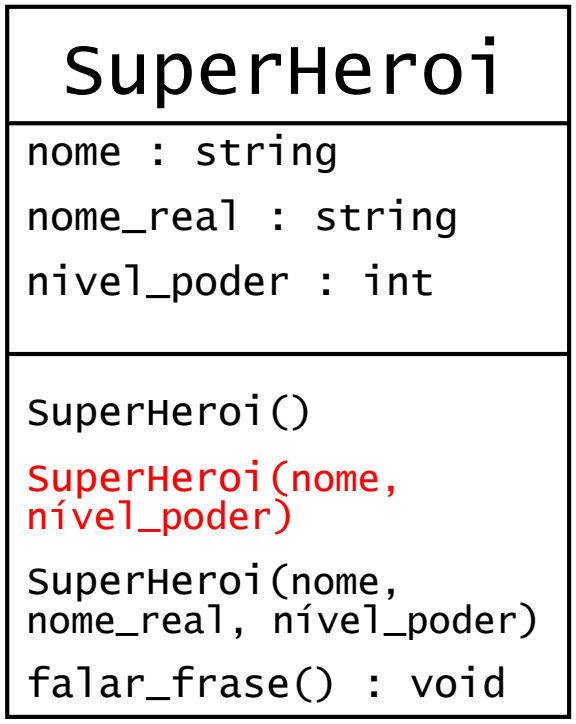
1. Fazer a mesma coisa para os objetos criados no exercício anterior: Gato e Pessoa.





EXERCÍCIO

11. Crie uma classe para o objeto abaixo, com Construtor e Construtor Padrão:



Estamos fazendo,

```
SuperHeroi(nome, nível_poder)
SuperHeroi(nome, nome_real, nível_poder)
```

E SE TENTARMOS FAZER:

```
SuperHeroi(nome, nome_real)
SuperHeroi(nome, nome_real, nível_poder)
```

AS FUNÇÕES POSSUEM O MESMO NOME?

ELAS FAZEM A MESMA COISA?

QUAL O NOME QUE DAMOS,
QUANDO TEMOS COISAS IGUAIS OU PARECIDAS
MAS QUE FAZEM COISAS DIFERENTES?

LEMBRA DO NOSSO POLIMORFISMO?

esse tipo de Polimorfismo chamamos de Sobrecarga

ENTENDENDO NOSSO CÓDIGO

POLIMORFISMO

Sobrecarga

Duas funções possuem o mesmo nome mas com assinaturas diferentes.

SOBRECARGA

Duas funções possuem o mesmo nome mas com assinaturas diferentes.

```
Cachorro()
Cachorro(nome, idade, peso)

SuperHeroi(nome, nível_poder)
SuperHeroi(nome, nome_real, nível_poder)
```

POR QUE O NOME Sobrecarga?

valendo um chocolate

EXERCÍCIO

III. Crie uma classe para o objeto abaixo, com Construtor e Construtor Padrão: