

Classes Abstratas, Funções Virtuais Puras e Funções Virtuais

Grupo: Silas, Nadhia, Mikael e Luiz

Universidade Federal de Campina Grande - PB

novembro 03, 2020



Sumário

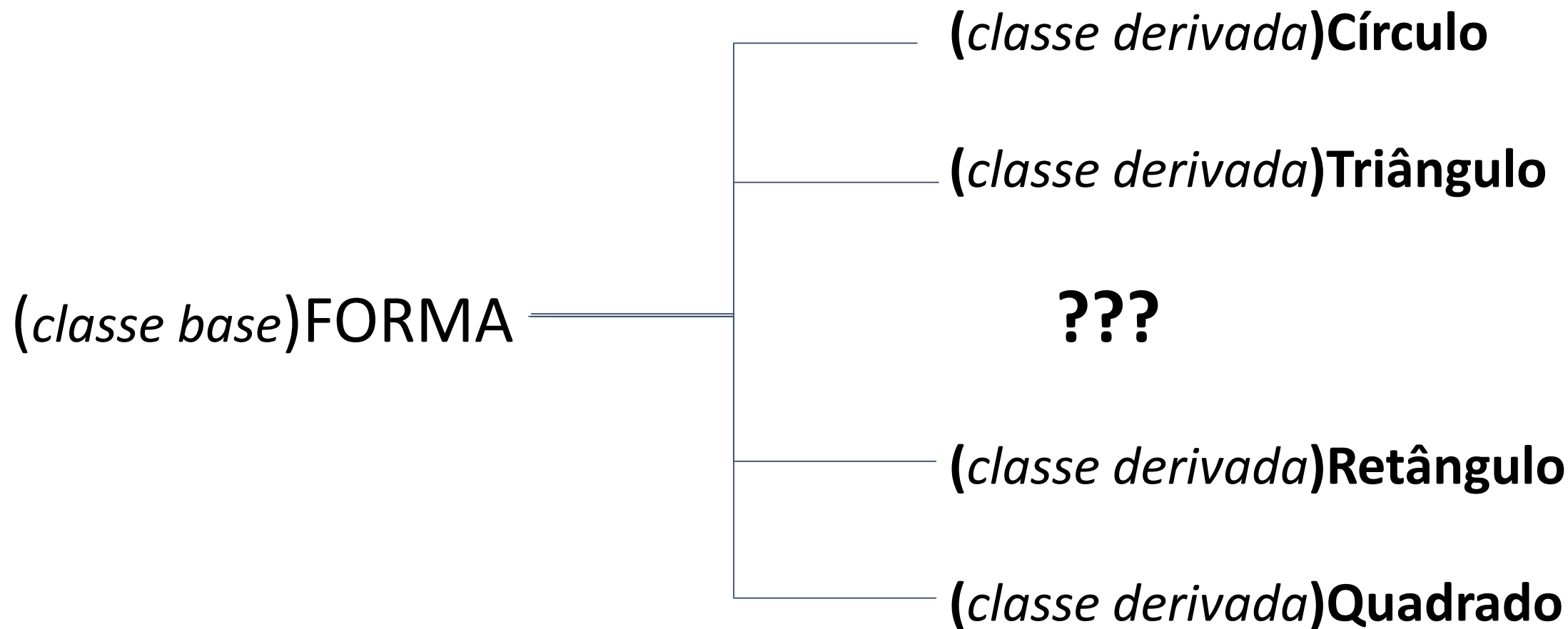
- ❑ Funções virtuais
- ❑ O que é uma função virtual
- ❑ Aplicações de funções virtuais
- ❑ Funções virtuais Puras
- ❑ Função virtual pura - aplicações
- ❑ Classes abstratas
- ❑ Exemplos de classes abstratas

Todos os exemplos podem ser encontrados no **GitHub** através do link:

<https://github.com/Silas-Joao/Exemplos-TP>



POLIMORFISMO



O que é uma função virtual

Uma função virtual é uma função que é declarada como virtual em uma classe base e redefinida pela classe derivada.

```
virtual void desenhar() const;
```

- ☐ Modo implícito;
- ☐ Modo explícito.

Aplicações de funções virtuais

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

class mae{
public:
    virtual void Nome () {    //DECLARANDO DE FORMA IMPLÍCITA, NO PONTO MAIS ALTO DA HIERARQUIA
        cout << "\n\n\tMAE\n\n";
    }
};


class filho:public mae{
public:
    void Nome () {
        cout << "\n\n\tFILHO da MAE\n\n";
    }
};


class filha:public mae{
public:
    void Nome () {
        cout << "\n\n\tFILHA da MAE\n\n";
    }
};
```

Aplicações de funções virtuais

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

class mae{
    public:
    virtual void Nome () { //DECLARANDO DE FORMA IMPLÍCITA, NO PONTO MAIS ALTO DA HIERARQUIA
        cout << "\n\n\tMAE\n\n";
    }
};

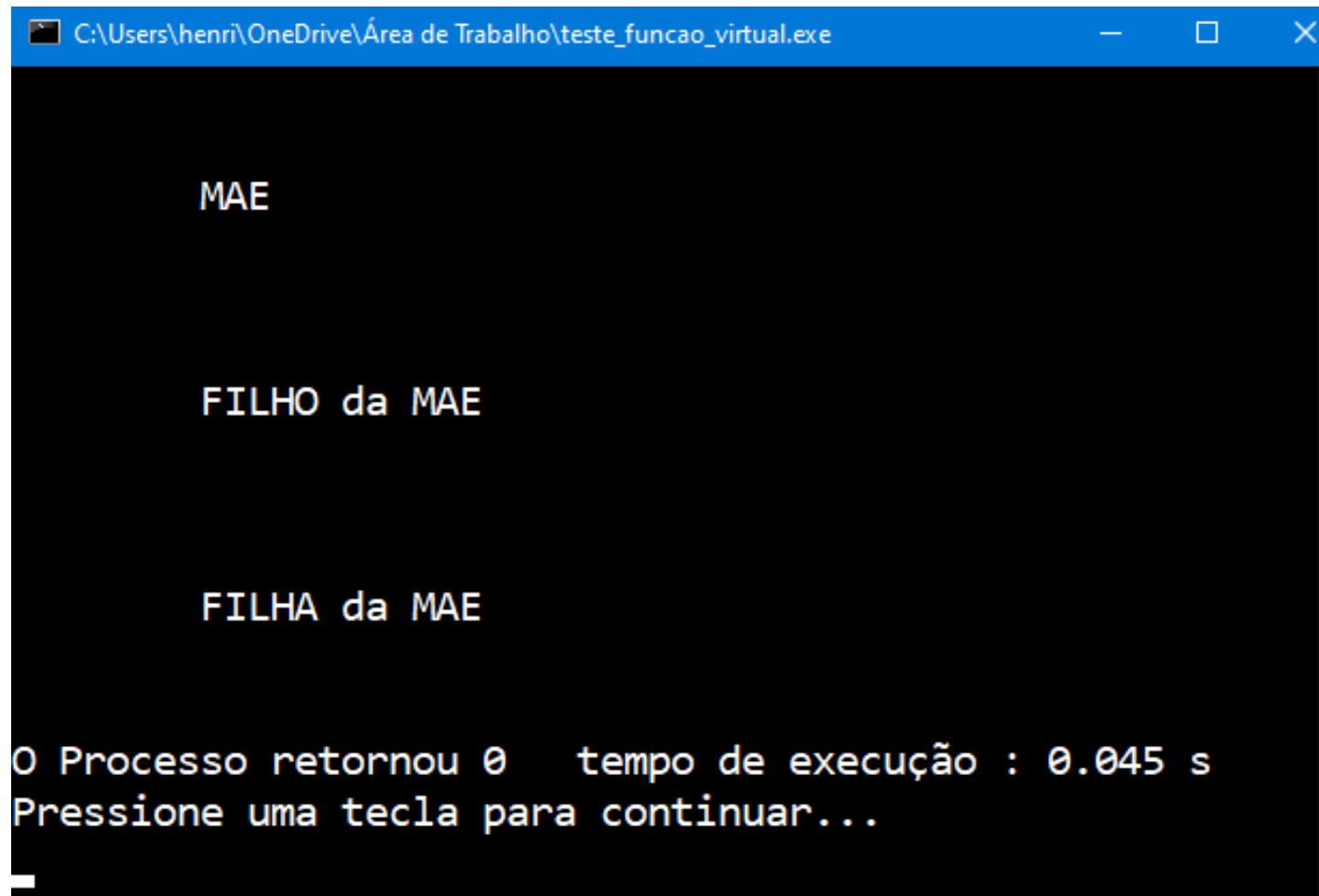
class filho:public mae{
     public:
        virtual void Nome () { //DECLARANDO DE FORMA EXPLÍCITA, EM CADA NÍVEL DE HIERARQUIA
            cout << "\n\n\tFILHO da MAE\n\n";
        }
};

class filha:public mae{
     public:
        virtual void Nome () { //DECLARANDO DE FORMA EXPLÍCITA, EM CADA NÍVEL DE HIERARQUIA
            cout << "\n\n\tFILHA da MAE\n\n";
        }
};
```

Aplicações de funções virtuais

```
int main() {  
    mae *N, obj01;  
    filho obj02;  
    filha obj03;  
  
    N = &obj01; //Ponteiro N de mae apontando para mae  
    N ->Nome();  
  
    N = &obj02; //Ponteiro N de mae apontando para filho  
    N ->Nome();  
  
    N = &obj03; //Ponteiro N de mae apontando para filha  
    N -> Nome();  
  
    return 0;  
}
```

Aplicações de funções virtuais



```
C:\Users\henri\OneDrive\Área de Trabalho\teste_funcao_virtual.exe

MAE

FILHO da MAE

FILHA da MAE

O Processo retornou 0 tempo de execução : 0.045 s
Pressione uma tecla para continuar...
```


Aplicações de funções virtuais

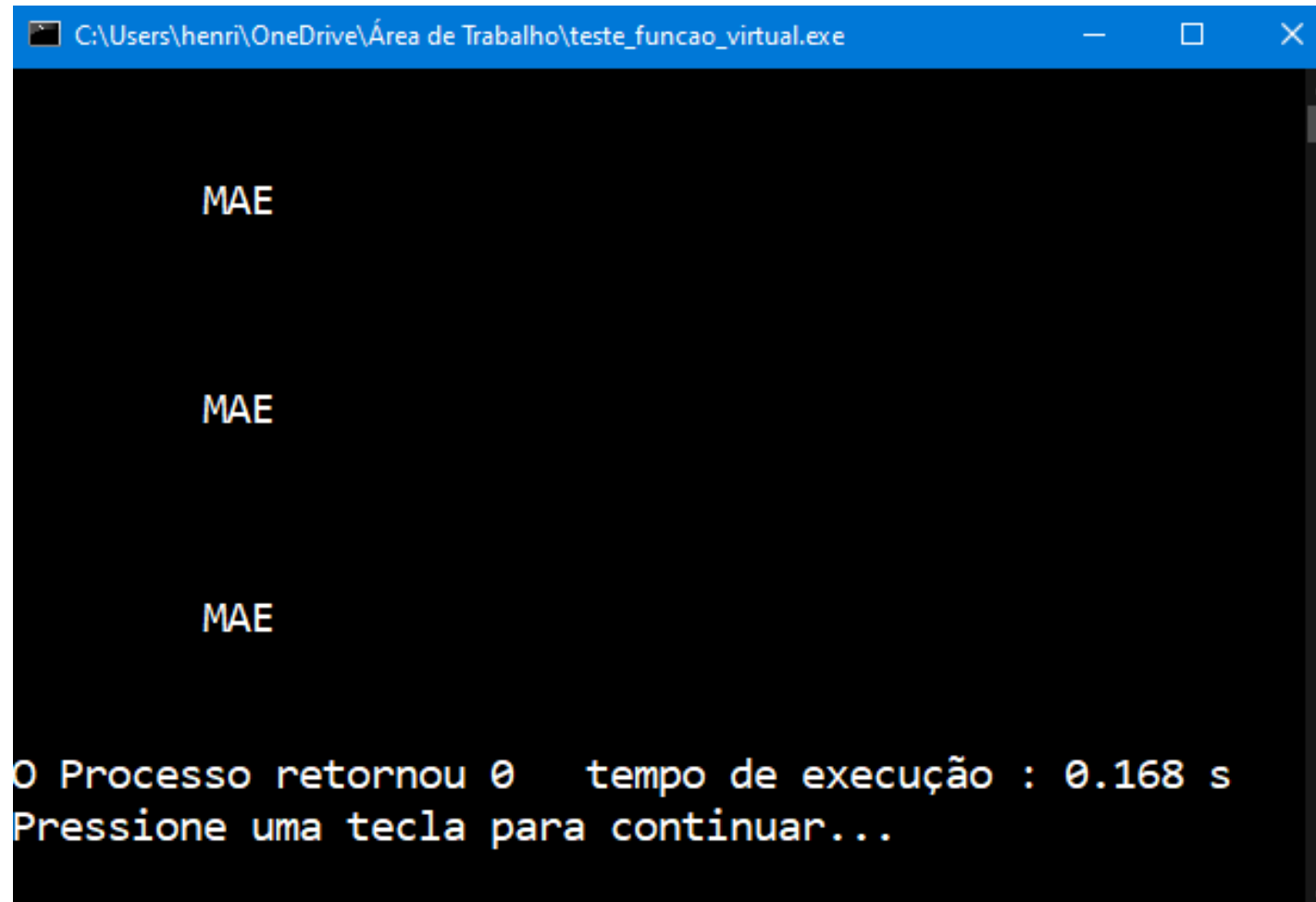
```
#include <iostream>
using namespace std;

class mae{
public:
    void Nome() { //DECLARANDO SEM A FUNÇÃO VIRTUAL
        cout << "\n\n\tMAE\n\n";
    }
};

class filho:public mae{
public:
    void Nome() {
        cout << "\n\n\tFILHO da MAE\n\n";
    }
};

class filha:public mae{
public:
    void Nome() {
        cout << "\n\n\tFILHA da MAE\n\n";
    }
};
```

Aplicações de funções virtuais



```
C:\Users\henri\OneDrive\Área de Trabalho\teste_funcao_virtual.exe

MAE

MAE

MAE

O Processo retornou 0   tempo de execução : 0.168 s
Pressione uma tecla para continuar...
```

Função Virtual Pura

Conceito - Método Virtual Puro

Uma função virtual pura é uma função que não tem implementação na própria classe e terá que ser implementada nas subclasses. Quando temos uma classe que possui esse método chamamos essa classe de classe abstrata que veremos mais a frente, porém as classes abstratas servem apenas de base para as suas subclasses. Sendo assim todas as classes que herdarem a classe que possui o método virtual puro terá que ser implementado esse novo método declarado na classe base.

Declaração de uma função virtual pura:

```
class G{  
    protected:  
        int comprimento, altura;  
  
    public:  
        //Declaração  
        virtual void Area() = 0;  
}
```

Observação: Quando uma função virtual é feita pura, qualquer classe derivada deve fornecer sua própria definição. Caso contrário, um erro de compilação será acusado.

Função Virtual Pura - Aplicações

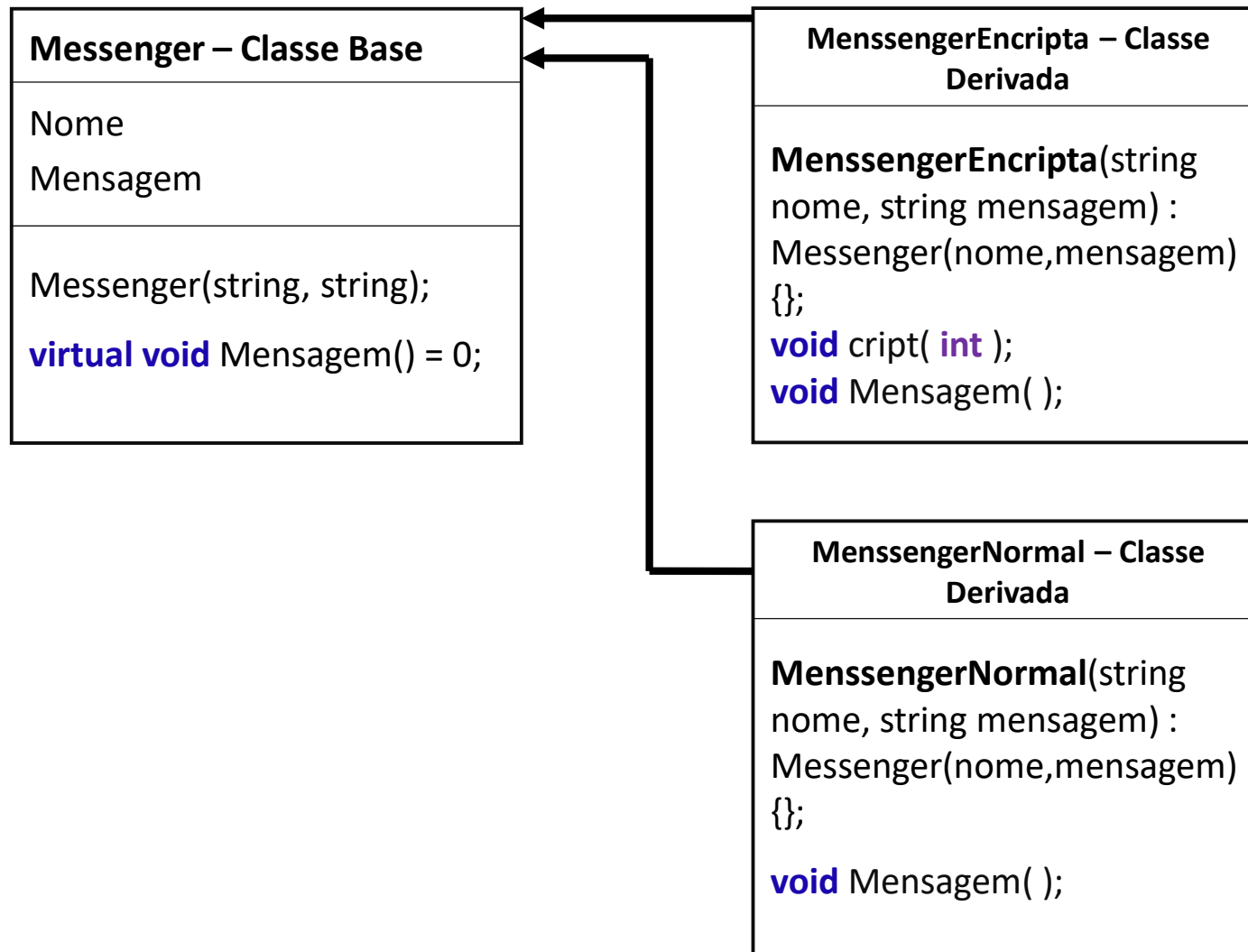
Quando usar uma função virtual pura ?

Funções virtuais puras também são usadas onde as declarações de métodos servem para definir uma interface para a qual as classes derivadas suprirão todas as implementações. Como por exemplo um programa que calcula todas as áreas de figuras geométricas onde poderíamos criar uma função virtual área na classe base “Figura” e todas as classes herdadas se encarregariam de calcular a área de determinada figura geométrica, sendo assim todas as classes que herdarem “Figura” teriam o método de área implementado.

-> Exemplo Prático !



Função Virtual Pura - Aplicações



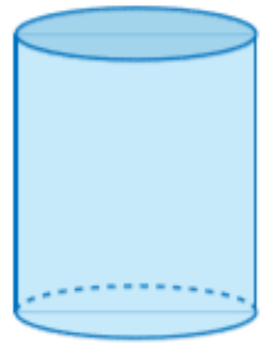
Conceito – Classes abstratas

- ❑ A classe abstrata não pode ser instanciada;
- ❑ Uma classe abstrata é uma classe que serve de modelo para outras classes;
- ❑ Ela sempre será uma superclasse genérica, e suas subclasses serão mais específicas.

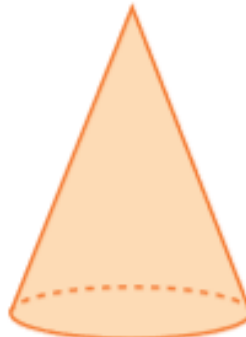
Imaginem uma Classe Tridimensional



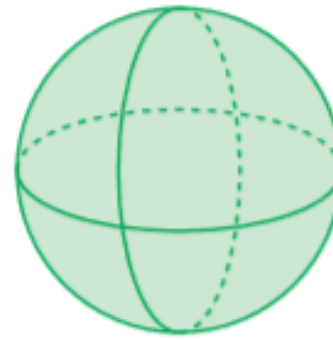
Imaginem as subclasses Cilindro, Cone e Esfera



Cilindro



Cone



Esfera

Fonte: Dante, 2012

Exemplo de classe abstrata

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

// CLASSE BASE
class Mamifero {
public:
    virtual void locomoverSe() = 0;
};

//SUBCLASSE
class Macaco : public Mamifero {
public:
    void locomoverSe() {
        cout << "pulando de galho em galho." << endl;
    }
};

//SUBCLASSE
class Baleia : public Mamifero {
public:
    void locomoverSe() {
        cout << "nadando." << endl;
    }
};

main() {
    Macaco macaco;
    Baleia baleia;
    macaco.locomoverSe();
    baleia.locomoverSe();
}
```