

Algoritmos e Programação II

José Ricardo Mello Viana

Aula 03
Introdução a classes e objetos

2010.2

- 1 Introdução
- 2 Classes, objetos, métodos e atributos
- 3 Definindo uma classe com um método

- Conceitos básicos de orientação a objeto
- Função **main** e uma ou mais classes contendo funções membro e membro de dados
- Exemplo de classe do mundo real

- Ex: Guiar um carro e fazê-lo andar mais rápido
- Alguém projetou e construiu o carro
- O pedal oculta os mecanismos complexos por trás
- Acelerador, freio, embreagem, volante, etc
- Interfaces simples e amigáveis

- É preciso construí-lo a partir dos desenhos
- O motorista deve pressionar o acelerador

- Realizar uma tarefa em um programa requer uma função (*main*)
- Descreve os mecanismos que realmente executam suas tarefas
- Oculta mecanismos complexos
- Criamos uma unidade de programa chamada **classe** para abrigar as funções
- Função pertencente a uma classe: **função membro** ou **método**

- Ex: Classe conta bancária
- Funções membro:
 - Depositar dinheiro
 - Retirar dinheiro
 - Consultar saldo
- É necessário criar um **objeto** de uma classe para poder utilizá-lo
- Muitos objetos podem ser construídos a partir da mesma classe

- Além das capacidades (funções) o carro também possui **atributos**
 - Cor, número de portas, quantidade de gasolina no tanque, velocidade atual, etc.
- Cada carro mantém seus próprios atributos
- Um **objeto** também possui atributos que são portados com ele durante a utilização no programa
- São especificados como parte da classe do objeto
- Ex: objetos conta bancária sabem seu próprio saldo, mas não o de outras contas

```
#include <iostream>
using namespace std;
class GradeBook {
public:
    //função que exibe uma mensagem de boas-vindas
    void displayMessage() {
        cout << "Bem vindo ao sistema GradeBook"
    } //fim da função displayMessage
} // fim da classe GradeBook
// função main
int main() {
    GradeBook objeto; // criação do objeto
    objeto.displayMessage(); // chamada do método
    return 0;
} // fim de main
```

- Possíveis erros
 - Esquecer o ponto-e-vírgula depois da classe (sintaxe)
 - Retornar um valor para um método declarado como void (compilação)
 - Definir uma função dentro de outra função (sintaxe)

- Passado parâmetro para métodos

```
void displayMessage(string nomeCurso) {  
    cout << "Bem vindo ao sistema GradeBook para o curso "  
    << nomeCurso << endl;  
}
```