PROGRAMAÇÃO ORIENTADA A OBJETOS

Introdução a Classes, Objetos, Métodos e Strings

Prof. Emanoel Barreiros

Classes e Objetos

- Aula passada usamos a classe Scanner, disponível no pacote java.util

```
1   import java.util.Scanner;
2
3   public class Soma {
4
5       public static void main(String[] args) {
6
7           //criar um novo objeto para obter entrada do usuário
8           Scanner scan = new Scanner(System.in);
9
10          //variáveis que serão usadas no código
11          int numero1;
12          int numero2;
13          int soma;
14
15          //solicita o primeiro número
16          System.out.print("Informe o número 1: ");
17          //obtém o primeiro número
18          numero1 = scan.nextInt();
```

Classes e Objetos

- Nós podemos no entanto criar as nossas próprias classes e fazê-las trabalhar para nós
- Criemos então, a classe LivroNotas
  - Crie um método chamado exibirBoasVindas, público, que não retorna nada e não tem nenhum parâmetro
  - Neste momento, ele deve apenas imprimir na tela a seguinte mensagem: “Bem-vindo ao Livro de Notas!”

- Qual a diferença entre o método ***main*** e esse método que estamos declarando?

Classes e Objetos

- O ***main*** é um método especial, chamado pela JVM quando executa nosso programa
- Os outros métodos precisam ser chamados por nós no programa
- Crie agora uma nova classe chamada LivroNotasTeste
- Esta classe será a classe principal do nosso programa, ela utilizará a classe LivroNotas
  - Declare o método ***main*** na classe LivroNotasTeste
  - Dentro do ***main*** crie um objeto do tipo LivroNotas e faça com que ele exiba a mensagem definida por ele

Métodos

- Como já vimos, métodos representam os comportamentos das classes
- Métodos podem ter parâmetros ou não
- O que acontece com as chamadas abaixo?
  - System.out.println(“Olá!”);
  - System.out.println();

- ATENÇÃO: Não confunda, acima estamos usando dois métodos diferentes. O exemplo serve apenas para mostrar que você não é obrigado a sempre declarar parâmetros para todos os métodos.

Métodos com parâmetros

- Altere a classe LivroNotas para que o método ***exibirBoasVindas*** receba como parâmetro o nome do curso para exibir e exiba a mensagem:
  - Bem-vindo ao Livro de Notas do curso <nome do curso>!
- Na classe LivroNotasTeste, altere o conteúdo do método ***main*** para que ele solicite o nome da disciplina ao usuário
- ***Dica 1: Use a classe Scanner, e seu método nextLine() para obter o nome do curso***
- ***Dica 2: O método nextLine() retorna uma String que deve ser armazenada para poder ser usada no código.***

Métodos com parâmetros

- A quantidade e tipos dos argumentos passados para o método devem ser os mesmos dos parâmetros definidos na declaração

- Se seu programa funcionou, você deve ter importado a classe Scanner: java.util.Scanner
- Porque você não precisou importar as classes String ou System?
  - As classes String e System estão no pacote java.lang que é automaticamente importado em todas as classes
  - Se você deseja usar classes do mesmo pacote onde está declarando uma outra classe, essas classes também não precisam ser importadas

Variáveis de instância

- Variáveis declaradas no bloco de um método tem escopo local. Isso significa que assim que o método termine sua execução, elas são “destruídas”.
- Variáveis locais não são visíveis fora do seu escopo
- Classes podem ter o que chamamos de variáveis de instância
  - Atributos da classe, suas características
  - Chamam-se variáveis de instância pois diferentes instâncias da classe podem ter valores diferentes para suas variáveis de instância
  - Também podemos chamar atributos de campos

Métodos *get* e *set*

- Por convenção, atributos de classe, na maioria das vezes, são privados
- Como faremos então para acessar os valores dos atributos (escrita e leitura)?
- O padrão *JavaBeans* define que valores de atributos devem ser acessados através de métodos *get* e *set*
- Métodos *get* e *set* devem ser assim na maioria dos casos:
  - public <tipo_da_variavel> getNomeDaVariavel()
  - public void setNomeDaVariavel(<tipo_da_variavel> valor)
- Os nomes destes métodos não são uma exigência da linguagem, mas uma convenção amplamente aceita, então faça assim ☺

Métodos *get* e *set* e o comando *return*

- Métodos *get* são usados para obter o valor dos atributos das classes, por isso possuem o valor de retorno do mesmo tipo do atributo associado a eles e não recebem nenhum parâmetro
- Em geral, a última linha dos métodos *get* são:
  - return <atributo>;
  - O comando *return* é usado para retornar um valor para quem chamou aquele método
- Métodos *set* são usados para atribuir valores aos atributos das classes, logo não precisam retornar nenhum valor (void) e recebe um parâmetro do mesmo tipo do atributo associado a ele

Modificadores de acesso *public* e *private*

- Modificadores de acesso podem aparecer em métodos, classes e atributos
- Eles definem a visibilidade de elementos no nosso código
- Quando algo é definido como *public* (público) qualquer outro objeto tem acesso àquele elemento, seja ele um método, classe ou atributo
- Quando algo é definido como *private* (privado) apenas a própria “entidade” que o define pode enxergá-lo e manipulá-lo
- Use o bom censo para definir o modificador de acesso, mas sempre utilize o privado a não ser que você tenha um bom motivo para não fazê-lo

Exercício

- Crie um atributo privado, do tipo String na classe LivroNotas chamado nomeCurso
- Crie os métodos *get* e *set* para este atributo novo
- Altere o método boasVindas para que ele não receba mais parâmetros, e agora use o valor do atributo nomeCurso
- Altere a classe LivroNotasTeste para que ela possa utilizar a classe LivroNotas após as suas alterações