Classes, instâncias e métodos de acesso

prática

Exercício 01

- Crie a classe Ponto que possui duas dimensões (int x, int y).
  - Crie os métodos get e set.
  - Faça o main para instanciar e testar a classe.
  - Adicione o método distancia (int x, int y) que calcula a distância do ponto às coordenadas (x,y). Teste!
  - Sobrecarregue o com o método distancia(Ponto p). Teste!

Saida de dados na console

- Use System.out.println(*dados*)

```
public class Teste {
   public static void main(String a[]){
      float total = 5.5F;
      System.out.print("O valor de total é ");
      System.out.println(total);
      // mesmo que
      System.out.println("O valor de total é "+total);

   }
}
```

Uso do println

Literais de caracteres são expressos por meio de um único carater delimitado por **apóstrofos**. Alguns literais de caracteres úteis são apresentados abaixo:

`\n`	nova linha
`\t`	tabulação
`\b`	retrocesso
`\\`	barra
`\'`	apóstrofo
`\"`	aspas

```
public class Teste {
   public static void main(String a[]){
      System.out.println("Clube de Futebol Mineiro:
         \n\"forte e vingador!\"");
   }
}
```

Saída de dados com printf

Um novo recurso, introduzido em **J2SE 5.0**, é o método **`System.out.printf`**, capaz de formatar mais adequadamente uma saída em Java, em uma forma semelhante à função **`printf`** da linguagem C.

```
public class Teste {
    public static void main(String[] a) {
        String s = "Uberlândia Esporte";
        System.out.printf("Grandioso %s\n",s);
    }
}
```

```
Grandioso Uberlândia Esporte
```

O primeiro argumento de **`printf`** é a **string de formatação**, composta de **constantes string** e **caracteres de formatação**. No exemplo acima, o **delimitador** **`%`** seguido de **`s`** indica que o método deve ter um outro argumento do tipo **`String`**.

Os **caracteres de formatação** podem também ser usados para compor uma **String**, usando o método **`String.format`**, conforme no exemplo a seguir:

```
public class Main {
  public static void main(String[] args) {
    int a = 65;
    String s = String.format("char: %c integer: %d
      hexa: %h octal: %o",a,a,a,a);
    System.out.println(s);
  }
}
```

Para o exemplo acima, onde é usado o **mesmo argumento** para a formatação, existem **alternativas** para a codificação:

```
String s = String.format("char: %c integer: %<d
      hexa: %<h octal: %<o",a);
```

```
String s = String.format("char: %1$c integer: %1$d
      hexa: %1$h octal: %1$o",a);
```

Entrada de dados com Scanner

A **leitura de dados de entrada em aplicativos através da janela de comando do sistema operacional** era surpreendentemente complexa até a versão **1.4.2** do J2SE. Isto não era problema para aplicativos com recursos GUI, mas tornava-se desconfortável para estudantes interessados em escrever programas simples com o propósito de aprender Java.

J2SE 5.0 introduz a classe **`java.util.Scanner`** para a **entrada de dados**, inclusive através da **entrada padrão**, **`System.in`**.

O exemplo a seguir apresenta uma **aplicação Java** para a entrada de dois números pelo teclado e a saída da soma na janela de comando do sistema operacional.

É necessário criar uma **instância da classe `Scanner`**, passando como argumento o **objeto padrão de entrada**, **`System.in`**, que irá capturar as entradas de dados via o **teclado**. O **método `nextInt()`**, da instância da classe **`Scanner`**, lê o valor digitado e o armazena em uma variável do tipo **`int`**.

```java
import java.util.Scanner;
public class Teste {
  public static void main(String[] args) {
    Scanner entrada = new Scanner(System.in);
    int num1;
    int num2;
    int soma;
    System.out.print("Entre com o primeiro número: ");
    num1 = entrada.nextInt();
    System.out.print("Entre com o segundo número: ");
    num2 = entrada.nextInt();
    soma = num1 + num2;
    System.out.printf("A soma é: %d", soma);
  }
}
```

Um exemplo de saída do aplicativo acima:

```
Entre com o primeiro número: 34
Entre com o segundo número: 23
A soma é: 57
```

O exemplo seguinte calcula o seno de um valor fornecido em tempo de execução:

```
import java.util.Scanner;
public class Teste {
  public static void main(String[] args) {
    Scanner entrada = new Scanner(System.in);
    double angulo;
    double seno;
    System.out.print("Entre com o ângulo em graus: ");
    angulo = entrada.nextDouble();
    seno = Math.sin(Math.toRadians(ângulo));
    System.out.println("Seno: " + seno);
  }
}
```

O método **`nextDouble()`**, da classe **`Scanner`**, captura um **`double`** da **entrada padrão `System.in`** e o armazerna em uma variável do tipo **`double`**.

A classe **`Math`** do pacote **`java.lang`** define a constante **`PI`** através de uma **variável de classe**. O **método estático `Math.sin(double)`** retorna o **seno** de um **ângulo em radianos** passado como **argumento**.

import static

A versão **J2SE 5.0** incorporou o recurso **`import static`**, que permite **importar os métodos e atributos estáticos**, permitindo utilizá-los **sem** o prefixo de classe.

O código anterior pode ser reescrito, a partir da versão 5.0, conforme abaixo:

```
import java.util.Scanner;
import static java.lang.Math.*;
public class Teste {
  public static void main(String[] args) {
    Scanner entrada = new Scanner(System.in);
    double angulo;
    double seno;
    System.out.print("Entre com o ângulo em graus: ");
    angulo = entrada.nextDouble();
    seno = sin(toRadians(ângulo));
    System.out.println("Seno: " + seno);
  }
}
```

Exercício 02

- Crie a classe Circulo.
  - Crie um construtor para inicializar a instância que recebe como parâmetros o raio e o valor do centro, que é um ponto em duas dimensões. Utilize a classe Ponto.
  - Adicione como atributos, também, um nome.
- Adicione como métodos:
  - Calcular diâmetro
  - Calcular área
  - Calcular circunferência
  - Acessar e modificar nome (não pode ser vazio)
  - Exibir os dados, utilizando o seguinte layout:

```
======================================

 Dados do circulo de raio 5,00
 Diametro      :  10,00
 Circunferencia:  31,42
 Area          :  78,54

======================================
```

Ex. 03: Implemente a classe “Conta”

- Atributos:
  - nome do cliente, senha, numero da conta e saldo
    - o saldo deverá ser inicializado com zero;
    - o número da conta deverá ser sequencial autoincremental (fazer um atributo estático contadorContas e incrementar a cada objeto instanciado. Use-o para definir o número da conta no construtor)
- Operações:
  - Construtor de conta, considerando que as informações passadas são APENAS: NOME, SENHA,
  - depositar(quantia) : adiciona quantia ao saldo;
  - sacar(quantia) : se saldo maior que quantia, diminuir quantia do saldo, caso contrário exibir mensagem "Saldo insuficiente";
  - exibirSaldo(), no seguinte formato:

```
-------------------------------------------------
 BANCO ONLINE    Conta: 17
 Olá João. O saldo da sua conta é de R$30,00.
-------------------------------------------------
```

Ex04: Cadastro de nomes

- Fazer uma classe Cadastro que contem
  - um atributo array de Strings
    - Lembre-se de criar as variáveis de controle (cont e MAX)
  - Métodos para manipular os Strings no array
    - void inserir (nome)
    - String buscar (posicao)
    - boolean existe (nome)
    - int posicao (nome)
    - void excluir (posicao)
    - void excluir (nome)
  - Fazer o main para testar