Métodos

PROF. RICARDO MESQUITA

Dicas importantes:

Para promover a capacidade de reutilização de software, todos os métodos devem estar limitados à realização de uma única tarefa bem definida.

- *Um método que realiza uma única tarefa é mais fácil de ser testado e depurado*
- *O nome do método deve ser conciso e que expresse o que ele realiza*
  - *Caso você encontre dificuldade nisso, reavalie: talvez, seu método esteja fazendo muitas tarefas!*
  - *Divida este método em partes menores!*

Métodos **static**

A maioria dos métodos é executado em respostas a chamadas de método sobre objetos específicos.

Quando um método não depende do conteúdo de nenhum objeto, significa que ele se aplica à classe onde é declarado como um todo.

Tais métodos são chamado **métodos static** ou **métodos de classe.**

É comum que as classes contenham métodos **static** convenientes para realizar tarefas corriqueiras

Por que o método **main** é **static**?

Ao executar a JVM, esta tenta invocar o método **main** da classe que você especifica – ou seja, quando nenhum objeto tiver sido criado ainda.

Ao declarar **main** como **static**, permite-se que a JVM invoque o método sem que nenhuma instância da classe tenha sido criada.

Argumentos de um Método

Podem ser constantes, variáveis ou expressões.

Por exemplo:

```
System.out.println( Math.sqrt(c + d * f) );
```

Observações:

- A classe **Math** faz parte do pacote **java.lang**
- *Todos* os métodos da classe **Math** são **static**

Observações

Métodos podem retornar no máximo **um valor**, mas o valor retornado pode ser uma referência para um objeto que contenha muitos valores.

Variáveis devem ser declaradas como campos de uma classe somente se forem utilizadas em mais de um método da classe, ou se o programa deva salvar seus valores entre as chamadas aos métodos da classe.

Declarar parâmetros de mesmo tipo como **float x, y** em vez de **float x, float y** é um erro de sintaxe – o tipo é requerido para cada parâmetro declarado!

Montagem de objetos String:

- O operador " + " concatena Strings
- *Todos os objetos* têm um método **`toString`** que retorna uma representação String do objeto!

Formas de Invocar um Método

Só o nome do método – quando é da mesma classe

Utilizar uma variável que contem uma referência a um objeto seguido por um ponto

Nome da classe seguido de ponto e do nome do método, para chamar um método **static**

- *Observação*: Um método **static** pode chamar qualquer método da mesma classe diretamente; por outro lado, só pode chamar métodos **static** da mesma classe diretamente e só pode manipular campos **static** na mesma classe diretamente.

Pacotes Java API

java.applet – executam em navegadores Web

java.awt – Java Abstract Window Toolkit Package – criação e manipulação de GUIs (antigas versões)

java.awt.event – permite tratamento de eventos em GUIs

java.awt.geom – Java 2D Shapes Package – recursos gráficos

java.io – Java Input/Output Package

java.lang – Java Language Package – contém as principais classes e interfaces

java.net – Java Network Package – classes e interfaces para redes

Pacotes Java API

java.sql – JDBC Package – classes e interfaces para BD

java.text – Java Text Package – manipulação de strings

java.util – Java Utilities Package – utilitários

java.util.concurrent – Java Concurrency Package

javax.media – Java Media Framework Package – multimídia

javax.swing – Java Swing GUI Components Package – suporte a GUIs portáveis

javax.swing.event – Java Swing Event Package – tratamento de eventos para componentes GUI

javax.xml.ws – JAX-WS Package – serviços Web

Escopo

Vamos escrever um código para testar escopos de variáveis (de instância e locais)

Vamos declarar uma variável de instância x, um método que use a variável de instância e um método que declare e use uma variável local de mesmo nome.

Vamos ver como o programa se comporta...

Escopo

- ✓ Declare uma classe Escopo com uma variável de instância. Vamos inicializa-la com o valor 1.

```
public class Escopo
{
   private int x = 1; // x como variável de instância
```

- ✓ Vamos declarar um método “inicio” que chamará os demais métodos da classe para o nosso teste de escopo

```
public void inicio()
{
   int x = 5; // x como variável local

   System.out.printf( "O valor de x local no método inicio  %d\n", x );

   usaVariavelLocal(); // usa a variável local
   usaCampo();         // usa a variável de instância
   usaVariavelLocal();
   usaCampo();

   System.out.printf( "\nO valor de x local no método início vale agora %d\n", x );
}
```

Escopo

- ✓ Agora vamos declarar um método simples para declarar e utilizar uma variável local com o mesmo nome da variável de instância

```
public void usaVariavelLocal()
{
   int x = 25;

   System.out.printf(
      "\nO valor da variável local x no início do método usaVariavelLocal é %d\n", x );
   ++x;
   System.out.printf(
      "O valor da variável local x no final do método usaVariavelLocal é %d\n", x );
}
```

Escopo

- ✓ Agora vamos declarar um método simples para manipular a variável de instância

```
public void usaCampo()
{
   System.out.printf(
      "\nO valor da variável de instância x no início do método usaCampo é %d\n", x );
   x *= 10;
   System.out.printf(
      "O valor da variável de instância x no final do método usaCampo é %d\n", x );
}
```

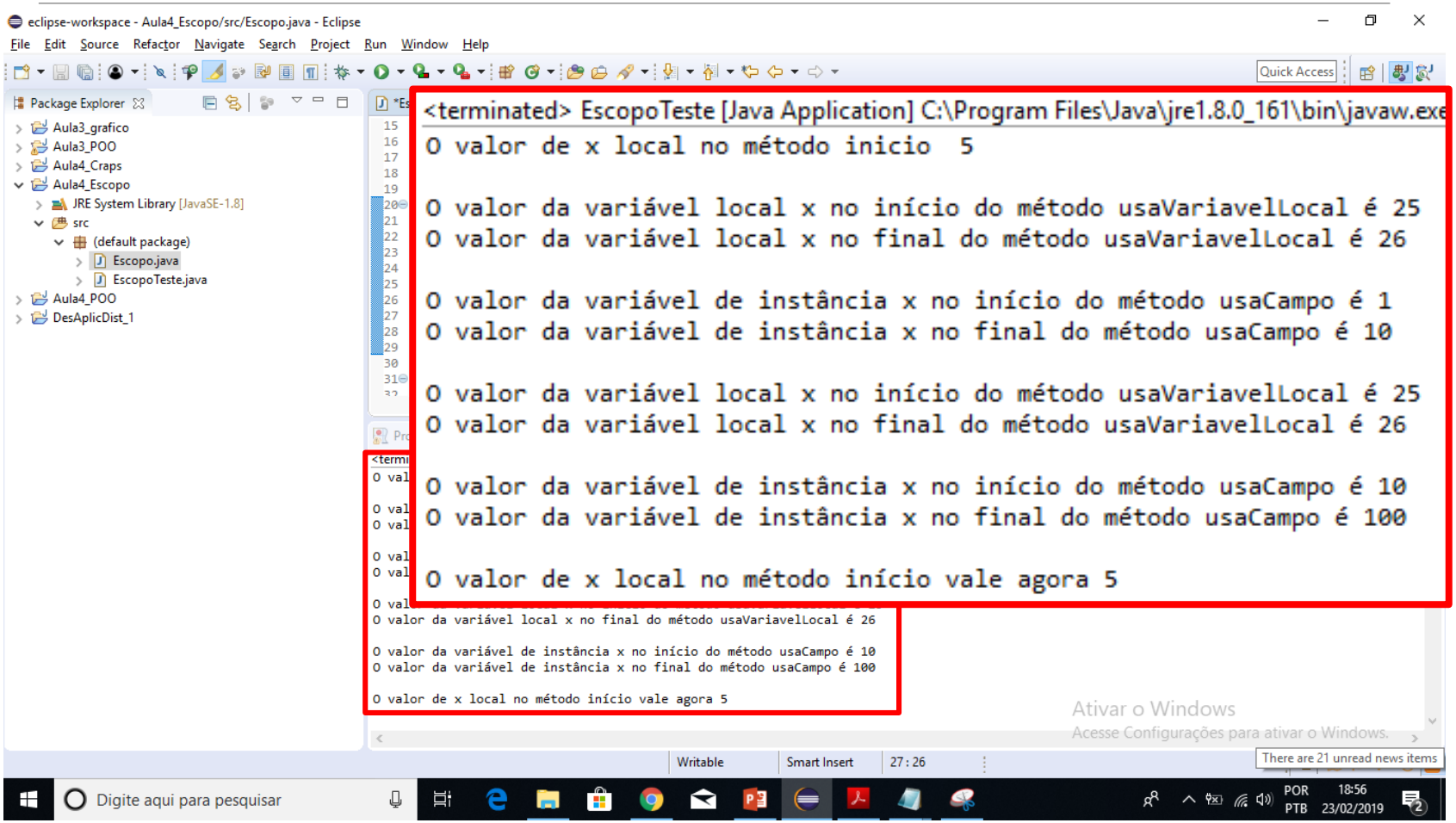
- ✓ Ok! Finalizamos a classe principal.

Escopo

- ✓ Defina a classe driver EscopoTeste

```
public class EscopoTeste
{
   public static void main( String[] args )
   {
      Escopo testeDeEscopo = new Escopo();
      testeDeEscopo.inicio();
   }
}
```

Escopo


eclipse-workspace - Aula4_Escopo/src/Escopo.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Package Explorer
Aula3_grafico
Aula3_POO
Aula4_Craps
Aula4_Escopo
JRE System Library [JavaSE-1.8]
src
(default package)
Escopo.java
EscopoTeste.java
Aula4_POO
DesAplicDist_1
<terminated> EscopoTeste [Java Application] C:\Program Files\Java\jre1.8.0_161\bin\javaw.exe
O valor de x local no método inicio 5
O valor da variável local x no início do método usaVariavelLocal é 25
O valor da variável local x no final do método usaVariavelLocal é 26
O valor da variável de instância x no início do método usaCampo é 1
O valor da variável de instância x no final do método usaCampo é 10
O valor da variável local x no início do método usaVariavelLocal é 25
O valor da variável local x no final do método usaVariavelLocal é 26
O valor da variável de instância x no início do método usaCampo é 10
O valor da variável de instância x no final do método usaCampo é 100
O valor de x local no método início vale agora 5
Writable
Smart Insert
27 : 26
Ativar o Windows
Acesse Configurações para ativar o Windows.
There are 21 unread news items
Digite aqui para pesquisar
POR
PTB
18:56
23/02/2019

Exemplos de Implementação

Vamos agora implementar alguns exemplos para praticarmos a implementação de alguns métodos...

Como primeiro exemplo, vamos escrever um programa para simular o lançamento de um dado e visualizar os valores de 20 desses lançamentos.

Para isso, faremos um programa *bem* simples usando a classe *Random* do pacote java.util para gerar os números aleatórios.

Vamos ao código...

Exemplo 1

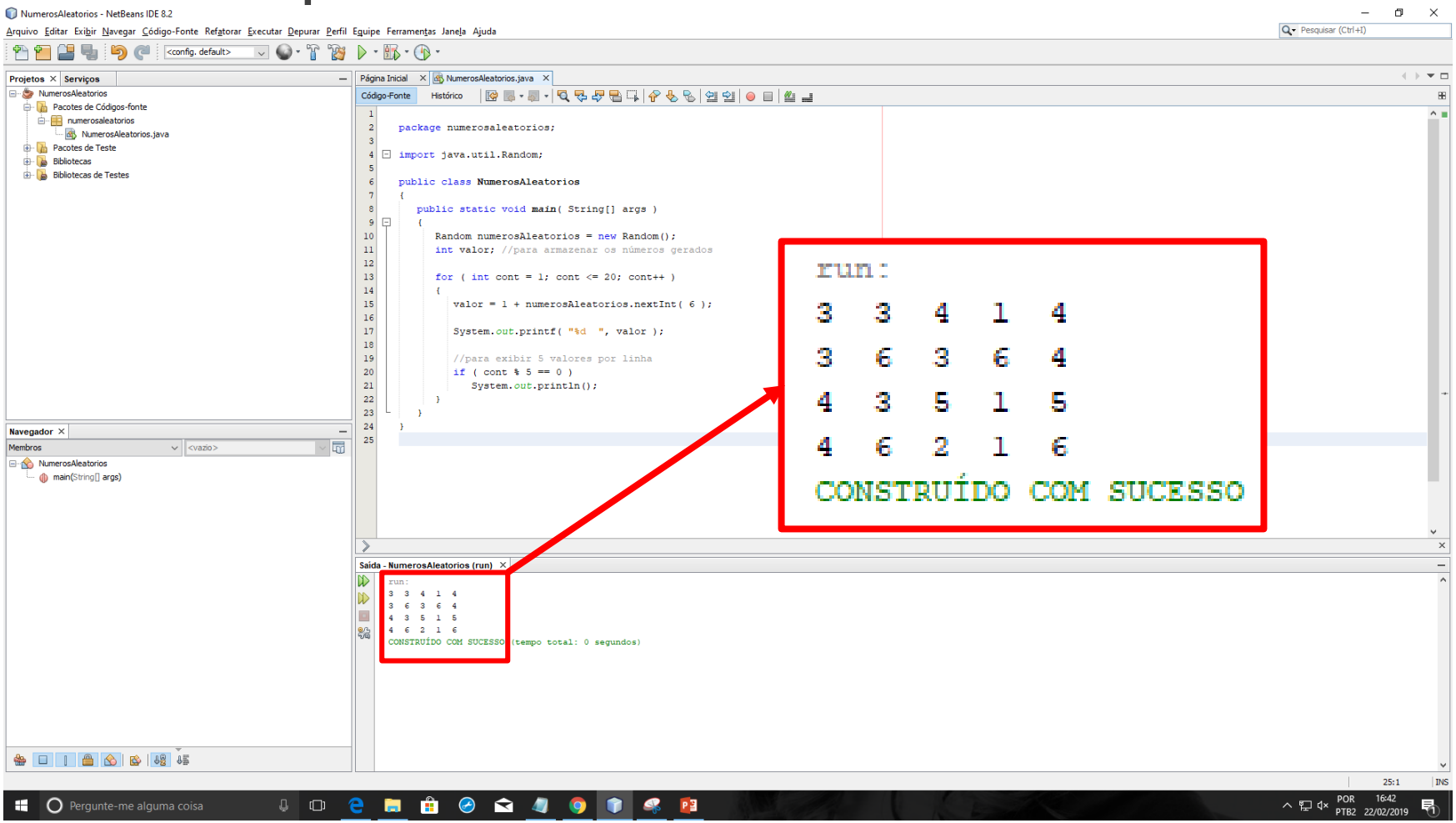
- ✓ Inicie a declaração da classe incluindo o import da classe Random
- ✓ No main, instancie um objeto para receber os números aleatórios
- ✓ Use um laço para gerar os 20 números (note que são mod 6)
- ✓ Vamos pular a linha de 5 em 5 para melhorar a leitura

```
import java.util.Random;

public class NumerosAleatorios
{
    public static void main( String[] args )
    {
        Random numerosAleatorios = new Random();
        int valor; //para armazenar os números gerados
        for ( int cont = 1; cont <= 20; cont++ )
        {
            valor = 1 + numerosAleatorios.nextInt( 6 );

            System.out.printf( "%d  ", valor );
            //para exibir 5 valores por linha
            if ( cont % 5 == 0 )
               System.out.println();
        }
    }
}
```

Exemplo 1


NumerosAleatorios - NetBeans IDE 8.2
Arquivo Editar Exibir Navegar Código-Fonte Refatorar Executar Depurar Perfil Equipe Ferramentas Janela Ajuda
Projetos
Serviços
NumerosAleatorios
Pacotes de Códigos-fonte
numerosaleatorios
NumerosAleatorios.java
Pacotes de Teste
Bibliotecas
Bibliotecas de Testes
Navegador
Membros
main(String[] args)
Página Inicial
NumerosAleatorios.java
Código-Fonte
Histórico
package numerosaleatorios;
import java.util.Random;
public class NumerosAleatorios
{
public static void main(String[] args)
{
Random numerosAleatorios = new Random();
int valor; //para armazenar os números gerados
for (int cont = 1; cont <= 20; cont++)
{
valor = 1 + numerosAleatorios.nextInt(6);
System.out.printf("%d ", valor);
//para exibir 5 valores por linha
if (cont % 5 == 0)
System.out.println();
}
}
}
Saída - NumerosAleatorios (run)
run:
3 3 4 1 4
3 6 3 6 4
4 3 5 1 5
4 6 2 1 6
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
run:
3 3 4 1 4
3 6 3 6 4
4 3 5 1 5
4 6 2 1 6
CONSTRUÍDO COM SUCESSO
Pergunte-me alguma coisa
25:1
INS
POR
PTB2
16:42
22/02/2019

Exemplo 2

Agora, vamos verificar se os números gerados são, estatisticamente, aleatórios. Para isso, vamos simular 1.000.000 lançamentos de um dado e computar as frequências.

Vamos ao código...

Exemplo 2

- ✓ Inicie a declaração da classe incluindo o import da classe Random
- ✓ No main, instancie um objeto para receber os números aleatórios e acrescente as variáveis apropriadas para controlar as respectivas frequências.
- ✓ Use um laço para gerar 1.000.000 de números aleatórios

```
import java.util.Random;

public class EstatisticaDado {

     public static void main( String[] args )
       {
          Random numerosAleatorios = new Random();

          int frequencia_1 = 0;
          int frequencia_2 = 0;
          int frequencia_3 = 0;
          int frequencia_4 = 0;
          int frequencia_5 = 0;
          int frequencia_6 = 0;

          int valor;

          for ( int cont = 1; cont <= 1000000; cont++ )
          {
             valor = 1 + numerosAleatorios.nextInt( 6 );
```

Exemplo 2

- ✓ Vamos usar um switch (dentro do laço) para controlar as frequências

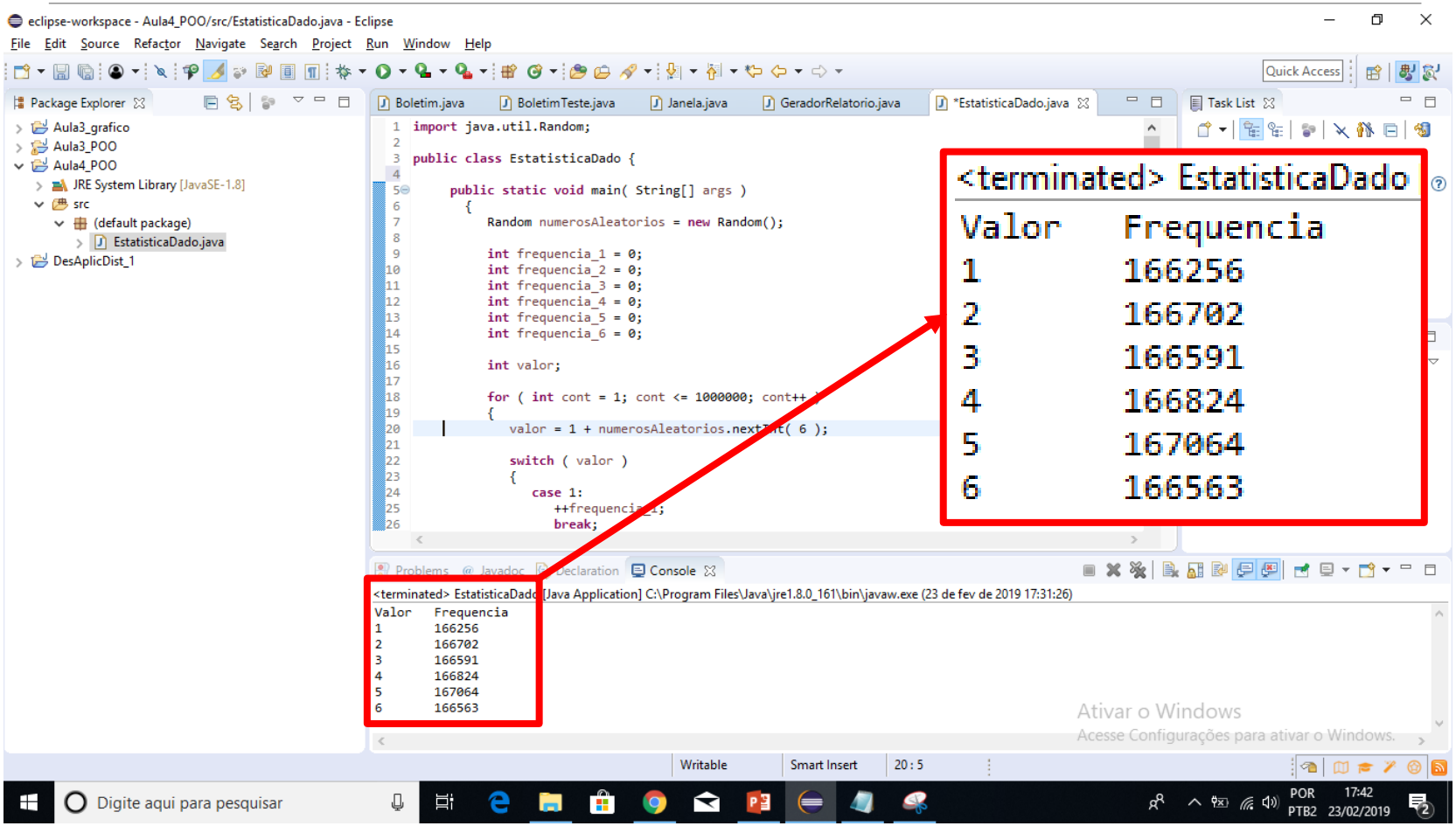
```
switch ( valor )
{
   case 1:
      ++frequencia_1;
      break;
   case 2:
      ++frequencia_2;
      break;
   case 3:
      ++frequencia_3;
      break;
   case 4:
      ++frequencia_4;
      break;
   case 5:
      ++frequencia_5;
      break;
   case 6:
      ++frequencia_6;
      break;
}
}
```

Exemplo 2

- ✓ Agora só falta exibir os resultados

```
            System.out.println( "Valor\tFrequencia" );
            System.out.printf( "1\t%d\n2\t%d\n3\t%d\n4\t%d\n5\t%d\n6\t%d\n",
               frequencia_1, frequencia_2, frequencia_3, frequencia_4,
               frequencia_5, frequencia_6 );
         }
   }
```

Exemplo 2


eclipse-workspace - Aula4_POO/src/EstatisticaDado.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
Aula3_grafico
Aula3_POO
Aula4_POO
JRE System Library [JavaSE-1.8]
src
(default package)
EstatisticaDado.java
DesAplicDist_1
Boletim.java
BoletimTeste.java
Janela.java
GeradorRelatorio.java
*EstatisticaDado.java
Task List
Quick Access
1 import java.util.Random;
3 public class EstatisticaDado {
5 public static void main(String[] args)
7 Random numerosAleatorios = new Random();
9 int frequencia_1 = 0;
10 int frequencia_2 = 0;
11 int frequencia_3 = 0;
12 int frequencia_4 = 0;
13 int frequencia_5 = 0;
14 int frequencia_6 = 0;
16 int valor;
18 for (int cont = 1; cont <= 1000000; cont++)
20 valor = 1 + numerosAleatorios.nextInt(6);
22 switch (valor)
24 case 1:
25 ++frequencia_1;
26 break;
Problems Javadoc Declaration Console
<terminated> EstatisticaDado [Java Application] C:\Program Files\Java\jre1.8.0_161\bin\javaw.exe (23 de fev de 2019 17:31:26)
Valor Frequencia
1 166256
2 166702
3 166591
4 166824
5 167064
6 166563
<terminated> EstatisticaDado
Valor Frequencia
1 166256
2 166702
3 166591
4 166824
5 167064
6 166563
Ativar o Windows
Acesse Configurações para ativar o Windows.
Writable
Smart Insert
20 : 5
Digite aqui para pesquisar
POR
PTB2
17:42
23/02/2019

Exemplo 3

Craps

Você lança dois dados. Cada dado tem seis faces que contêm 1, 2, 3, 4, 5 e 6 pontos, respectivamente. Depois que os dados param de rolar, a soma dos pontos nas faces viradas para cima é calculada. Se a soma for 7 ou 11, no primeiro lance, você ganha. Se a soma for 2, 3 ou 12 no primeiro lance (“craps”), você perde (a casa ganha). Se a soma for 4, 5, 6, 8, 9, ou 10 no primeiro lance, essa soma torna-se sua “pontuação”. Para ganhar você deverá continuar a rolar os dados até “fazer a sua pontuação” (isto é, obter novamente o valor da pontuação). Você perde se obtiver um 7 antes de fazer sua pontuação.

Vamos inicialmente fazer uma aplicação que simula todo o jogo, sem interações com o usuário (fazer apostas, por exemplo)

Vamos ao código...

Exemplo 3

- ✓ Vamos fazer as declarações iniciais

Este objeto é compartilhado em todas as chamadas do método jogaOsDados()

Status é uma variável de enumeração (constantes representadas por identificadores)

```
import java.util.Random;

public class Craps
{
   private static final Random randomNumbers = new Random();

   private enum Status { CONTINUE, GANHOU, PERDEU };

   private static final int SNAKE_EYES = 2;
   private static final int TREY = 3;
   private static final int SEVEN = 7;
   private static final int YO_LEVEN = 11;
   private static final int BOX_CARS = 12;
```

Exemplo 3

- ✓ Iniciemos o método play()

```
public void play()
{
   int meusPontos = 0;
   Status statusDoJogo;

   int somaDePontos = jogaOsDados();

   switch ( somaDePontos )
   {
      case SEVEN:
      case YO_LEVEN:
         statusDoJogo = Status.GANHOU;
         break;
      case SNAKE_EYES:
      case TREY:
      case BOX_CARS:
         statusDoJogo = Status.PERDEU;
         break;
      default:
         statusDoJogo = Status.CONTINUE;
         meusPontos = somaDePontos;
         System.out.printf( "Seus pontos são %d\n", meusPontos );
         break;
   }
```

Chamada a outro método

Bloco switch para controlar o status do jogo

Exemplo 3

- ✓ A finalização da lógica do método play() inclui um laço para evolução do status...

```
        while ( statusDoJogo == Status.CONTINUE )
        {
            somaDePontos = jogaOsDados();

            if ( somaDePontos == meusPontos )
                statusDoJogo = Status.GANHOU;
            else
                if ( somaDePontos == SEVEN )
                    statusDoJogo = Status.PERDEU;
        }

        if ( statusDoJogo == Status.GANHOU )
            System.out.println( "Você Ganhou!!" );
        else
            System.out.println( "Você Perdeu!!" );
    }
```

Exemplo 3

- ✓ Agora, vamos declarar o método jogaOsDados()...

```
    public int jogaOsDados()
    {

        int dado1 = 1 + randomNumbers.nextInt( 6 );
        int dado2 = 1 + randomNumbers.nextInt( 6 );

        int soma = dado1 + dado2;

        System.out.printf( "Sua jogada: %d + %d = %d\n",
           dado1, dado2, soma );

        return soma;
    }
}
```

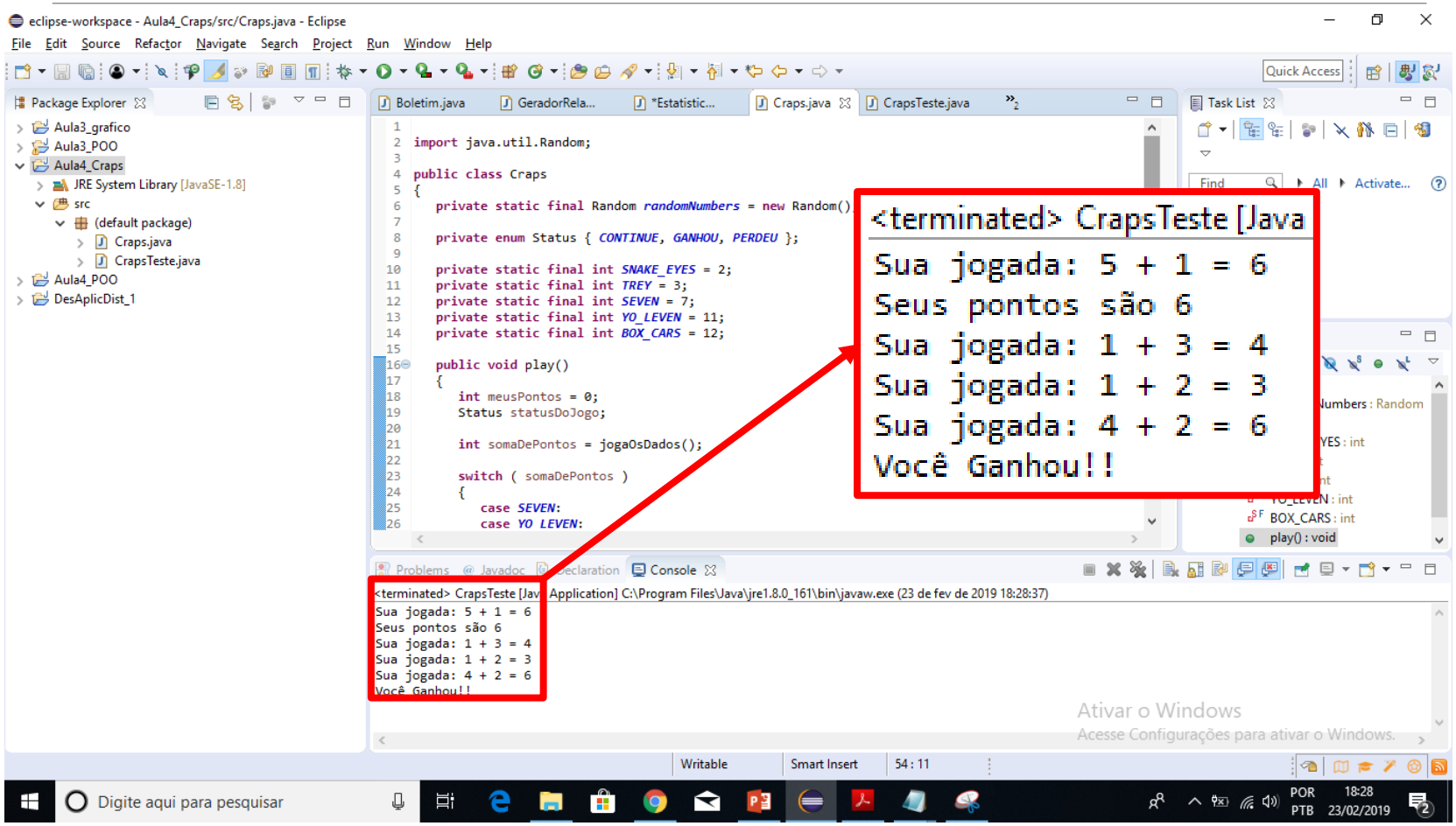
Exemplo 3

- ✓ Para finalizar, vamos declarar uma classe driver para chamar a execução
- ✓ Lembre-se: trata-se de outro arquivo, pois é uma outra classe pública!

```
public class CrapsTeste {

    public static void main( String[] args )
       {
          Craps game = new Craps();
          game.play();
       }
}
```

Exemplo 3


eclipse-workspace - Aula4_Craps/src/Craps.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
Aula3_grafico
Aula3_POO
Aula4_Craps
JRE System Library [JavaSE-1.8]
src
(default package)
Craps.java
CrapsTeste.java
Aula4_POO
DesAplicDist_1
Boletim.java
GeradorRela...
*Estatistic...
Craps.java
CrapsTeste.java
import java.util.Random;
public class Craps
{
private static final Random randomNumbers = new Random();
private enum Status { CONTINUE, GANHOU, PERDEU };
private static final int SNAKE_EYES = 2;
private static final int TREY = 3;
private static final int SEVEN = 7;
private static final int YO_LEVEN = 11;
private static final int BOX_CARS = 12;
public void play()
{
int meusPontos = 0;
Status statusDoJogo;
int somaDePontos = jogaOsDados();
switch (somaDePontos)
{
case SEVEN:
case YO_LEVEN:
<terminated> CrapsTeste [Java
Sua jogada: 5 + 1 = 6
Seus pontos são 6
Sua jogada: 1 + 3 = 4
Sua jogada: 1 + 2 = 3
Sua jogada: 4 + 2 = 6
Você Ganhou!!
Task List
Problems Javadoc Declaration Console
<terminated> CrapsTeste [Java Application] C:\Program Files\Java\jre1.8.0_161\bin\javaw.exe (23 de fev de 2019 18:28:37)
Sua jogada: 5 + 1 = 6
Seus pontos são 6
Sua jogada: 1 + 3 = 4
Sua jogada: 1 + 2 = 3
Sua jogada: 4 + 2 = 6
Você Ganhou!!
Ativar o Windows
Acesse Configurações para ativar o Windows.
Writable
Smart Insert
54 : 11
Digite aqui para pesquisar
POR
PTB
18:28
23/02/2019

Dúvidas?