PUC-Minas - Ciência da Computação
AED1 – Estudo Dirigido 03

Tema: Introdução à programação
Atividade: Métodos em Java

01.) Editar e salvar um esboço de programa em Java:

```
/**
  * Exemplo0041
  *
  * @author
  * @version 01
  */

// ---------------------------------------------  dependencias

import IO.*;

// ---------------------------------------------  definicao da classe principal

public class Exemplo0041
{
// ---------------------------------------------  definicao de metodo auxiliar

   public static void metodo01 ( )
   {
     // definir dados
        int x;
     // identificar o metodo
        IO.println ( “Metodo 01” );
     // ler valor inteiro do teclado
        x = IO.readint ( “Entrar com um valor inteiro: ” );
        IO.println ( “Valor lido = ” + x );
   } // fim metodo01( )

// ---------------------------------------------  definicao do metodo principal

   /**
     * main() – metodo principal
     */
   public static void main ( String [ ] args )
   {
     // identificar
        IO.println ( "EXEMPLO0041 - Programa em Java" );
        IO.println ( "Autor: _____" );
     // executar o metodo auxiliar
        metodo01 ( );   // ler e mostrar valor inteiro
     // encerrar
        IO.pause   ( “Apertar ENTER para terminar.” );
   } // fim main( )
} // fim class Exemplo0041
```

```
// ---------------------------------------------- documentacao complementar
//
// ---------------------------------------------- historico
//
// Versao      Data        Modificacao
// 0.1                     __ / __      esboco
//
// ---------------------------------------------- testes
//
// Versao      Teste
// 0.1                     01. ( ___ )     identificacao de programa e leitura
//
```

02.) Compilar o programa novamente.

Se houver erros, resolvê-los e compilar novamente, até que todos tenham sido resolvidos.

Se não houver erros, seguir para o próximo passo.

Em caso de dúvidas, consultar a apostila, recorrer aos monitores ou apresentá-las ao professor.

03.) Executar o programa.

Observar as saídas.

Registrar os dados e os resultados.

```
// ---------------------------------------------- testes
//
// Versao      Teste
// 0.1                     01. ( OK )      identificacao de programa e leitura
//                                         Valores previstos: 5, 0, -5
//
```

Em caso de erro (ou dúvida), usar comentários para registrar a ocorrência e, posteriormente, tentar resolvê-lo (ou esclarecer a dúvida).

04.) Copiar a versão atual do programa para outra nova – Exemplo0042.java.

05.) Editar mudanças no nome do programa e versão.
Acrescentar outro método com uma repetição para
ler uma certa quantidade de valores inteiros.
Na parte principal, editar a chamada para o novo método.
Prever novos testes.

```
// ------------------------------------------------  definicao de metodo auxiliar

   public static void metodo02 ( )
   {
    // definir dados
       int x;
    // definir dado para guardar o resultado
       int quantidade = 0;
    // identificar o metodo
       IO.println ( "Metodo 02" );
    // ler a quantidade do teclado
       quantidade = IO.readint ( "Entrar com a quantidade: " );
       while ( quantidade > 0 )
       {
         // ler valor inteiro do teclado
            x = IO.readint ( "Entrar com um valor inteiro: " );
            IO.println ( "Valor lido = " + x );
          // diminuir a quantidade
            quantidade = quantidade – 1;
       } // fim repetir
   } // fim metodo02( )
```

OBS.:
Ao terminar a repetição, a quantidade será zero.
O valor lido inicialmente não será mais conhecido.

06.) Compilar o programa novamente.
Se houver erros, resolvê-los e compilar novamente, até que todos tenham sido resolvidos.
Se não houver erros, seguir para o próximo passo.

07.) Executar o programa.
Observar as saídas.
Registrar os dados e os resultados.

08.) Copiar a versão atual do programa para outra nova – Exemplo0043.java.

09.) Editar mudanças no nome do programa e versão.
Acrescentar outro método com uma repetição com contagem decrescente para ler uma certa quantidade de valores inteiros.
Na parte principal, editar a chamada para o novo método.
Prever novos testes.

```
// ------------------------------------------------  definicao de metodo auxiliar

   public static void metodo03 ( )
   {
     // definir dados
        int x;
     // definir dado para guardar o resultado
        int quantidade = 0;
     // definir contador
        int contador;
     // identificar o metodo
        IO.println ( "Metodo 03" );
     // ler a quantidade do teclado
        quantidade = IO.readint ( "Entrar com a quantidade: " );
        contador = quantidade;
        while ( contador > 0 )
        {
          // ler valor inteiro do teclado
             x = IO.readint ( "Entrar com um valor inteiro: " );
             IO.println ( "Valor lido = " + x );
          // contar mais um valor lido
             contador = contador - 1;
        } // fim repetir
   } // fim metodo03( )
```

10.) Compilar o programa novamente.
Se houver erros, resolvê-los e compilar novamente, até que todos tenham sido resolvidos.
Se não houver erros, seguir para o próximo passo.

11.) Executar o programa.
Observar as saídas.
Registrar os dados e os resultados.

12.) Copiar a versão atual do programa para outra nova – Exemplo0044.java.

13.) Editar mudanças no nome do programa e versão.
Acrescentar outro método com uma repetição com variação crescente para ler uma certa quantidade de valores inteiros.
Na parte principal, editar a chamada para o novo método.
Prever novos testes.

```
// ------------------------------------------------  definicao de metodo auxiliar

   public static void metodo04 ( )
   {
     // definir dados
        int x;
     // definir dado para guardar o resultado
        int quantidade = 0;
     // definir contador
        int contador;
     // identificar o metodo
        IO.println ( "Metodo 04" );
     // ler a quantidade do teclado
        quantidade = IO.readint ( "Entrar com a quantidade: " );
        contador = 1;
        while ( contador <= quantidade )
        {
          // ler valor inteiro do teclado
             x = IO.readint ( "Entrar com um valor inteiro: " );
             IO.println ( "Valor lido = " + x );
          // contar mais um valor lido
             contador = contador +  1;
        } // fim repetir
   } // fim metodo04( )
```

14.) Compilar o programa novamente.
Se houver erros, resolvê-los e compilar novamente, até que todos tenham sido resolvidos.
Se não houver erros, seguir para o próximo passo.

15.) Executar o programa.
Observar as saídas.
Registrar os dados e os resultados.

16.) Copiar a versão atual do programa para outra nova – Exemplo0045.java.

17.) Editar mudanças no nome do programa e versão.
Acrescentar outro método com uma repetição com variação crescente,
começando em zero, para ler uma certa quantidade de valores inteiros.
Na parte principal, editar a chamada para o novo método.
Prever novos testes.

```
// ------------------------------------------------  definicao de metodo auxiliar

   public static void metodo05 ( )
   {
     // definir dados
        int x;
     // definir dado para guardar o resultado
        int quantidade = 0;
     // definir contador
        int contador;
     // identificar o metodo
        IO.println ( "Metodo 05" );
     // ler a quantidade do teclado
        quantidade = IO.readint ( "Entrar com a quantidade: " );
        contador = 0;
        while ( contador < quantidade )
        {
          // ler valor inteiro do teclado
             x = IO.readint ( "Entrar com um valor inteiro: " );
             IO.println ( "Valor lido = " + x );
          // contar mais um valor lido
             contador = contador +  1;
        } // fim repetir
   } // fim metodo05( )
```

18.) Compilar o programa novamente.
Se houver erros, resolvê-los e compilar novamente, até que todos tenham sido resolvidos.
Se não houver erros, seguir para o próximo passo.

19.) Executar o programa.
Observar as saídas.
Registrar os dados e os resultados.

20.) Copiar a versão atual do programa para outra nova – Exemplo0046.java.

21.) Editar mudanças no nome do programa e versão.
Na parte principal, incluir uma escolha para executar qualquer um dos métodos já feitos.
Prever novos testes.

```
// ------------------------------------------------  definicao do metodo principal

   /**
    * main() – metodo principal
    */
   public static void main ( String [ ] args )
   {
     // definir dado
        int opcao;
     // identificar
        IO.println ( "EXEMPLO0046 - Programa em Java" );
        IO.println ( "Autor: _____" );
     // mostrar opcoes
        IO.println ( “Opcoes:” );
        IO.println ( “1 – Metodo 01” );
        IO.println ( “2 – Metodo 02” );
        IO.println ( “3 – Metodo 03” );
        IO.println ( “4 – Metodo 04” );
        IO.println ( “5 – Metodo 05” );
     // ler a opcao do teclado
       opcao = IO.readint ( “Qual a sua opcao? ” );
     // escolher qual metodo executar
        switch ( opcao )
        {
         case 1:    metodo01 ( );
          break;
         case 2:    metodo02 ( );
          break;
         case 3:    metodo03 ( );
          break;
         case 4:    metodo04 ( );
          break;
         case 5:    metodo05 ( );
          break;
        } // fim escolher
     // encerrar
        IO.pause   ( “Apertar ENTER para terminar.” );
   } // fim main( )
} // fim class Exemplo0046
```

22.) Compilar o programa novamente.
Se houver erros, resolvê-los e compilar novamente, até que todos tenham sido resolvidos.
Se não houver erros, seguir para o próximo passo.

23.) Executar o programa.
Observar as saídas.
Registrar os dados e os resultados.

24.) Copiar a versão atual do programa para outra nova – Exemplo0047.java.

25.) Editar mudanças no nome do programa e versão.
Incluir o tratamento de uma escolha inválida.
Prever novos testes.

```
// ------------------------------------------------  definicao do metodo principal

   /**
    * main() – metodo principal
    */
   public static void main ( String [ ] args )
   {
     // definir dado
        int opcao;
     // identificar
        IO.println ( "EXEMPLO0047 - Programa em Java" );
        IO.println ( "Autor: _____" );
     // mostrar opcoes
        IO.println ( “Opcoes:” );
        IO.println ( “1 – Metodo 01” );
        IO.println ( “2 – Metodo 02” );
        IO.println ( “3 – Metodo 03” );
        IO.println ( “4 – Metodo 04” );
        IO.println ( “5 – Metodo 05” );
     // ler a opcao do teclado
       opcao = IO.readint ( “Qual a sua opcao? ” );
     // escolher qual metodo executar
        switch ( opcao )
        {
         case 1:    metodo01 ( );
          break;
         case 2:    metodo02 ( );
          break;
         case 3:    metodo03 ( );
          break;
         case 4:    metodo04 ( );
          break;
         case 5:    metodo05 ( );
          break;
         default:
           IO.println ( “ERRO: Opcao invalida.” );
        } // fim escolher
     // encerrar
        IO.pause   ( “Apertar ENTER para terminar.” );
   } // fim main( )
} // fim class Exemplo0047
```

26.) Compilar o programa novamente.
Se houver erros, resolvê-los e compilar novamente, até que todos tenham sido resolvidos.
Se não houver erros, seguir para o próximo passo.

27.) Executar o programa.
Observar as saídas.
Registrar os dados e resultados.

28.) Copiar a versão atual do programa para outra nova – Exemplo0048.java.

29.) Editar mudanças no nome do programa e versão.
Incluir uma repetição para executar a ação principal três vezes.
Prever novos testes.

```
// ------------------------------------------------  definicao do metodo principal

   /**
    * main() – metodo principal
    */
   public static void main ( String [ ] args )
   {
    // definir dado
      int opcao;
      int contador = 3;
      while ( contador > 0 )
      {
       // identificar
         IO.println ( "EXEMPLO0048 - Programa em Java" );
         IO.println ( "Autor: _____" );
       // mostrar opcoes
         IO.println ( “Opcoes:” );
         IO.println ( “1 – Metodo 01” );
         IO.println ( “2 – Metodo 02” );
         IO.println ( “3 – Metodo 03” );
         IO.println ( “4 – Metodo 04” );
         IO.println ( “5 – Metodo 05” );
       // ler a opcao do teclado
         opcao = IO.readint ( “Qual a sua opcao? ” );
       // escolher qual metodo executar
         switch ( opcao )
         {
          case 1:   metodo01 ( );
           break;
          case 2:   metodo02 ( );
           break;
          case 3:   metodo03 ( );
           break;
          case 4:   metodo04 ( );
           break;
          case 5:   metodo05 ( );
           break;
          default:
           IO.println ( “ERRO: Opcao invalida.” );
         } // fim escolher
       // diminuir o contador
         contador = contador – 1;
      } // fim repetir
    // encerrar
      IO.pause   ( “Apertar ENTER para terminar.” );
   } // fim main( )
} // fim class Exemplo0048
```

30.) Compilar o programa novamente.
Se houver erros, resolvê-los e compilar novamente, até que todos tenham sido resolvidos.
Se não houver erros, seguir para o próximo passo.

31.) Executar o programa. Observar as saídas. Registrar os dados e os resultados.

32.) Copiar a versão atual do programa para outra nova – Exemplo0049.java.

33.) Editar mudanças no nome do programa e versão.
Incluir uma repetição para executar a ação principal qualquer quantidade de vezes.
Prever novos testes.

```
// ------------------------------------------------  definicao do metodo principal

   /**
    * main() – metodo principal
    */
   public static void main ( String [ ] args )
   {
    // definir dado
      int opcao;
      int contador = IO.readint ( “Quantas vezes quer executar o programa? ” );
      while ( contador > 0 )
      {
       // identificar
         IO.println ( "EXEMPLO0049 - Programa em Java" );
         IO.println ( "Autor: _____" );
       // mostrar opcoes
         IO.println ( “Opcoes:” );
         IO.println ( “1 – Metodo 01” );
         IO.println ( “2 – Metodo 02” );
         IO.println ( “3 – Metodo 03” );
         IO.println ( “4 – Metodo 04” );
         IO.println ( “5 – Metodo 05” );
       // ler a opcao do teclado
         opcao = IO.readint ( “Qual a sua opcao? ” );
       // escolher qual metodo executar
         switch ( opcao )
         {
          case 1:   metodo01 ( );
           break;
          case 2:   metodo02 ( );
           break;
          case 3:   metodo03 ( );
           break;
          case 4:   metodo04 ( );
           break;
          case 5:   metodo05 ( );
           break;
          default:
           IO.println ( “ERRO: Opcao invalida.” );
         } // fim escolher
       // diminuir o contador
         contador = contador – 1;
      } // fim repetir
    // encerrar
      IO.pause   ( “Apertar ENTER para terminar.” );
   } // fim main( )
} // fim class Exemplo0049
```

34.) Compilar o programa novamente. Se houver erros, resolvê-los; senão seguir para o próximo passo.

35.) Executar o programa. Observar as saídas. Registrar os dados e os resultados.

36.) Copiar a versão atual do programa para outra nova – Exemplo0050.java.

37.) Editar mudanças no nome do programa e versão.
Incluir uma opção para encerrar o programa quando quiser.

```
// ------------------------------------------  definicao do metodo principal

   /**
    * main() – metodo principal
    */
   public static void main ( String [ ] args )
   {
     // definir dado
        int opcao;
        do
        {
          // identificar
            IO.println ( "EXEMPLO0050 - Programa em Java" );
            IO.println ( "Autor: _____" );
          // mostrar opcoes
            IO.println ( “Opcoes:” );
            IO.println ( “0 – Parar );
            IO.println ( “1 – Metodo 01” );
            IO.println ( “2 – Metodo 02” );
            IO.println ( “3 – Metodo 03” );
            IO.println ( “4 – Metodo 04” );
            IO.println ( “5 – Metodo 05” );
          // ler a opcao do teclado
            opcao = IO.readint ( “Qual a sua opcao? ” );
          // escolher qual metodo executar
            switch ( opcao )
            {
             case 0:    //  nao faz nada
               break;
             case 1:    metodo01 ( );
              break;
             case 2:    metodo02 ( );
              break;
             case 3:    metodo03 ( );
              break;
             case 4:    metodo04 ( );
              break;
             case 5:    metodo05 ( );
              break;
             default:
              IO.println ( “ERRO: Opcao invalida.” );
            } // fim escolher
        }
        while ( opcao != 0 );
     // encerrar
        IO.pause   ( “Apertar ENTER para terminar.” );
   } // fim main( )
} // fim class Exemplo0050
```

Exercícios:
DICAS GERAIS: Consultar o Anexo Java 02 na apostila para outros exemplos.
Prever, realizar e registrar todos os testes efetuados.

01.) Fazer um programa (Exemplo0051) com um método (06)
capaz de lidar com dados com valores inteiros (**int**).
Ler a quantidade, primeiro; e vários outros valores, depois, um por vez;
separar e mostrar apenas os valores menores que 100.
DICA: Incluir um teste para verificar se cada valor lido é menor que 100,
e mostrá-lo somente nesse caso.

02.) Fazer um programa (Exemplo0052) com um método (07)
capaz de lidar com dados com valores inteiros (**int**).
Ler a quantidade, primeiro; e vários outros valores, depois, um por vez;
separar os ímpares menores que 100.
DICA: Incluir um teste para verificar se cada valor lido é ímpar e menor que 100,
e mostrá-lo somente nesse caso.

03.) Fazer um programa (Exemplo0053) com um método (08)
capaz de lidar com dados com valores reais (**double**).
Ler a quantidade, primeiro; e vários outros valores, depois, um por vez;
separar os ímpares e positivos entre 25 e 125.
DICA: Incluir um teste para verificar se cada valor é ímpar e positivo,
pertence ao intervalo [25:125]
e mostrá-lo somente nesse caso.

04.) Fazer um programa (Exemplo0054) com um método (09)
capaz de lidar com dados com valores reais (**double**).
Ler a quantidade, primeiro; e vários outros valores, depois, um por vez;
separar os positivos, menores que 100, mas mostrar somente a parte inteira.
DICA: Incluir um teste para verificar se cada valor lido é positivo,
e mostrá-lo somente nesse caso.
Para mostrar a parte inteira, usar a conformação de tipo: “” + (**int**) x
ou guardar essa parte em um dado inteiro antes de mostrar: y = (int) x;
OBS: Nesse último caso, o armazenador do dado deverá ser definido primeiro: **int** y;

05.) Fazer um programa (Exemplo0055) com um método (10)
capaz de lidar com dados com valores reais (**double**).
Ler a quantidade, primeiro; e vários outros valores, depois, um por vez;
separar os positivos, menores que 100,
mas mostrar a parte fracionária, primeiro, separada da parte inteira.
DICA: Incluir um teste para verificar se cada valor lido é positivo,
e mostrá-lo somente nesse caso.
Para mostrar a parte fracionária, subtrair a parte inteira (ver acima).

06.) Fazer um programa (Exemplo0056) com um método (11)
capaz de lidar com dados com valores literais (**char**).
Ler a quantidade, primeiro; e vários outros valores, depois, um por vez;
separar as letras minúsculas.
DICA: Incluir um teste para verificar se cada valor lido é uma letra minúscula,
e mostrá-lo somente nesse caso.
Para testar se é letra minúscula, verificar se é maior ou igual a ‘a’ e menor ou igual a ‘z’.

07.) Fazer um programa (Exemplo0057) com um método (12)
capaz de lidar com dados com valores literais (**char**).
Ler a quantidade, primeiro; e vários outros valores, depois, um por vez;
separar as letras maiúsculas.
DICA: Incluir um teste para verificar se cada valor lido é uma letra maiúscula,
e mostrá-lo somente nesse caso.
Para testar se é letra maiúscula, verificar se é maior ou igual a ‘A’ e menor ou igual a ‘Z’.

08.) Fazer um programa (Exemplo0058) com um método (13)
capaz de lidar com dados com valores literais (**char**).
Ler a quantidade, primeiro; e vários outros valores, depois, um por vez;
separar os algarismos.
DICA: Incluir um teste para verificar se cada valor lido é um algarismo,
e mostrá-lo somente nesse caso.
Para testar se é algarismo, verificar se é maior ou igual a ‘0’ e menor ou igual a ‘9’.

09.) Fazer um programa (Exemplo0059) com um método (14)
capaz de lidar com dados com valores literais (**char**).
Ler a quantidade, primeiro; e vários outros valores, depois, um por vez;
separar as letras.
DICA: Incluir um teste para verificar se cada valor lido é uma letra,
e mostrá-lo somente nesse caso.
Para testar se é letra, verificar se é maiúscula ou minúscula.

10.) Fazer um programa (Exemplo0060) com um método (15)
capaz de lidar com dados com valores literais (**char**).
Ler a quantidade, primeiro; e vários outros valores, depois, um por vez;
separar os símbolos alfanuméricos (letras e algarismos).
DICA: Incluir o uso de um teste para verificar, primeiro, se cada valor lido é uma letra,
independente de ser maiúscula ou minúscula.

Tarefas extras

E1.) Fazer um programa com um método
capaz de lidar com dados com cadeia de caracteres (**String**).
Ler uma cadeia de caracteres do teclado,
separar e mostrar separadamente os símbolos alfanuméricos (letras e algarismos).
DICA: Para saber a quantidade de símbolos na cadeia de caracteres, usar:

```
String s;
int quantidade;

. . .

quantidade = s.length( );
```

Para ter acesso a determinado caractere na cadeia, usar:

```
char c;

. . .

c = s.charAt( 0 );                // para o primeiro

c = s.charAt( x );                // para determinada posicao (x)

c = s.charAt( s.length()-1 );     // para o ultimo
```

E2.) Fazer um programa com um método
capaz de lidar com dados com valores em cadeias de caracteres (**String**).
Ler a quantidade, primeiro; e vários outros valores, depois, um por vez;
separar os símbolos alfanuméricos (letras e algarismos) em cada cadeia.