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

Tema: Introdução à programação
Atividade: Repetições em C

01.) Editar e salvar um esboço de programa em C, cujo nome será Exemplo0301.c:
com modelos de repetições (teste no início e teste no fim):

```
/*
  Exemplo0301 - v0.0. - __ / __ / _____
  Author: _____
*/
// dependencias
#include "io.h"            // para definicoes proprias

/**
  Method00 - nao faz nada.
 */
void method00 ( )
{
 // nao faz nada
} // fim method00 ( )

/**
  Method01 - Repeticao com teste no inicio.
 */
void method01 ( )
{
 // definir dado
   int x = 0;

 // identificar
   IO_id ( "EXEMPLO0301 - Method01 - v0.0" );

 // ler do teclado o valor inicial
   x = IO_readint ( "Entrar com uma quantidade: " );

 // repetir (x) vezes
   while ( x > 0 )
   {
    // mostrar valor atual
      IO_println ( IO_toString_d ( x ) );
    // passar ao proximo valor
      x = x - 1;
   } // fim repetir

 // encerrar
   IO_pause ( "Apertar ENTER para continuar" );
} // fim method01 ( )
```

```
/*
  Funcao principal.
  @return codigo de encerramento
*/
int main ( )
{
 // definir dado
    int x = 0;

 // repetir até desejar parar
    do
    {
     // identificar
        IO_id ( "EXEMPLO0301 - Programa - v0.0" );

     // ler do teclado
        IO_println ( "Opcoes" );
        IO_println ( "0 - parar" );
        IO_println ( "1 - repeticao com teste no inicio" );
        IO_println ( "" );

        x = IO_readint ( "Entrar com uma opcao: " );

     // testar valor
        switch ( x )
        {
          case 0:
            method00 ( );
           break;
          case 1:
            method01 ( );
           break;
          default:
            IO_pause ( IO_concat ( "Valor diferente das opcoes [0,1] (",
                          IO_concat ( IO_toString_d ( x ), ")" ) ) );
        } // fim escolher
    }
    while ( x != 0 );

 // encerrar
    IO_pause ( "Apertar ENTER para terminar" );
    return ( 0 );
} // fim main( )
```

/*
---------- documentacao complementar

---------- notas / observacoes / comentarios

---------- previsao de testes

a.) 0
b.) 1
c.) 2
d.) 3
e.) 4
f.) -1

---------- historico

Versao	**Data**	**Modificacao**
0.1	**__/__**	**esboco**

---------- testes

Versao	**Teste**	
0.1	**01. (OK)**	**identificacao de programa**

***/**

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

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 erro (ou dúvida), usar comentários para registrar a ocorrência e,
posteriormente, tentar pesquisar solução (ou esclarecer a dúvida),
consultar a bibliografia ou apostila, recorrer aos monitores ou reportar ao professor.

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

04.) Copiar a versão atual do programa para outra nova – Exemplo0302.c.

05.) Editar mudanças no nome do programa e versão.
Incluir novo método, e na parte principal, incluir uma alternativa para executá-lo.
Uma forma alternativa de controle da repetição será apresentada.
Prever novos testes.

```
/**
  Method02 - Repeticao com teste no inicio.
 */
void method02 ( )
{
 // definir dado
   int x = 0;
   int y = 0;

 // identificar
   IO_id ( "EXEMPLO0302 - Method02 - v0.0" );

 // ler do teclado
   x = IO_readint ( "Entrar com uma quantidade: " );

 // repetir (x) vezes
   y = x;                    // copiar o valor lido (e' melhor)
   while ( y > 0 )
   {
    // mostrar valor atual
      IO_println ( IO_toString_d ( x ) );
    // passar ao proximo valor
      y = y - 1;
   } // fim repetir

 // encerrar
   IO_pause ( "Apertar ENTER para continuar" );
} // fim method02 ( )
```

```
/*
  Funcao principal.
*/
int main ( )
{
 // definir dado
    int x = 0;
 // repetir até desejar parar
    do
    {
     // identificar
        IO_id ( "EXEMPLO0302 - Programa - v0.0" );

     // ler do teclado
        IO_println ( "Opcoes" );
        IO_println ( "0 - parar" );
        IO_println ( "1 - repeticao com teste no inicio (decrescente)" );
        IO_println ( "2 - repeticao com teste no inicio ( alternativo  )" );
        IO_println ( "" );

        x = IO_readint ( "Entrar com uma opcao: " );

     // testar valor
        switch ( x )
        {
          case 0:
            method00 ( );
           break;
          case 1:
            method01 ( );
           break;
          case 2:
            method02 ( );
           break;
          default:
            IO_pause ( IO_concat ( "Valor diferente das opcoes [0,1,2] (",
                                   IO_concat ( IO_toString_d ( x ), ")" ) ) );
        } // fim escolher
    }
    while ( x != 0 );

 // encerrar
    IO_pause ( "Apertar ENTER para terminar" );
    return ( 0 );
} // fim main( )
```

/*
---------- documentacao complementar

---------- notas / observacoes / comentarios

---------- previsao de testes

a.) 0
b.) 1
c.) 5
d.) -5

---------- historico

Versao	Data	Modificacao
0.1	__/__	esboco

---------- testes

Versao	Teste	
0.1	01. (OK)	identificacao de programa

***/**

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 – Exemplo0303.c.

09.) Editar mudanças no nome do programa e versão.
Incluir novo método, e na parte principal, incluir uma alternativa para executá-lo.
Uma forma de repetição com variação crescente será apresentada.
Prever novos testes.

```
/**
  Method03 - Repeticao com teste no inicio.
 */
void method03 ( )
{
 // definir dado
   int x = 0;
   int y = 0;

 // identificar
   IO_id ( "EXEMPLO0303 - Method03 - v0.0" );

 // ler do teclado
   x = IO_readint ( "Entrar com uma quantidade: " );

 // repetir (x) vezes
   y = 1;                    // o valor lido devera' ser preservado
   while ( y <= x )
   {
    // mostrar valor atual
      IO_printf ( "%d\n", y );
    // passar ao proximo valor
      y = y + 1;
   } // fim repetir

 // encerrar
   IO_pause ( "Apertar ENTER para continuar" );
} // fim method03 ( )
```

```
/*
  Funcao principal.
*/
int main ( )
{
 // definir dado
    int x = 0;

 // repetir até desejar parar
    do
    {
     // identificar
        IO_id ( "EXEMPLO0303 - Programa - v0.0" );

     // ler do teclado
        IO_println ( "Opcoes" );
        IO_println ( "0 - parar" );
        IO_println ( "1 - repeticao com teste no inicio (decrescente)" );
        IO_println ( "2 - repeticao com teste no inicio (alternativo)" );
        IO_println ( "3 - repeticao com teste no inicio ( crescente )" );
        IO_println ( "" );

        x = IO_readint ( "Entrar com uma opcao: " );

     // testar valor
        switch ( x )
        {
          case 0:
            method00 ( );
           break;
          case 1:
            method01 ( );
           break;
          case 2:
            method02 ( );
           break;
          case 3:
            method03 ( );
           break;
          default:
            IO_pause ( IO_concat ( "Valor diferente das opcoes [0,1,2,3] (",
                                   IO_concat ( IO_toString_d ( x ), ")" ) ) );
        } // fim escolher
    }
    while ( x != 0 );

 // encerrar
    IO_pause ( "Apertar ENTER para terminar" );
    return ( 0 );
} // fim main( )
```

/*
---------- documentacao complementar

---------- notas / observacoes / comentarios

---------- previsao de testes

a.) 0
b.) 1
c.) 3
d.) 5
e.) -5

---------- historico

Versao	Data	Modificacao
0.1	__/__	esboco

---------- testes

Versao	Teste	
0.1	01. (OK)	identificacao de programa

***/**

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 – Exemplo0304.c.

13.) Editar mudanças no nome do programa e versão.
Incluir novo método, e na parte principal, incluir uma alternativa para executá-lo.
Uma forma mais compacta de enunciar a repetição com variação será apresentada.
Prever novos testes.

```
/**
  Method04 - Repeticao com teste no inicio e variacao.
 */
void method04 ( )
{
 // definir dado
    int x = 0;
    int y = 0;

 // identificar
    IO_id ( "EXEMPLO0304 - Method04 - v0.0" );

 // ler do teclado
    x = IO_readint ( "Entrar com uma quantidade: " );

 // repetir (x) vezes
 //       inicio    teste   variacao
    for ( y  = 1; y <= x; y = y + 1 )
    {
     // mostrar valor atual
        IO_printf ( "%d\n", y );
    } // fim repetir

 // encerrar
    IO_pause ( "Apertar ENTER para continuar" );
} // fim method04 ( )
```

```
/*
  Funcao principal.
*/
int main ( )
{
 // definir dado
    int x = 0;

 // repetir até desejar parar
    do
    {
     // identificar
        IO_id ( "EXEMPLO0304 - Programa - v0.0" );

     // ler do teclado
        IO_println ( "Opcoes" );
        IO_println ( "0 - parar" );
        IO_println ( "1 - repeticao com teste no inicio                (decrescente)" );
        IO_println ( "2 - repeticao com teste no inicio                (  alternativo )" );
        IO_println ( "3 - repeticao com teste no inicio                (  crescente  )" );
        IO_println ( "4 - repeticao com teste no inicio e variacao (  crescente  )" );
        IO_println ( "" );

        x = IO_readint ( "Entrar com uma opcao: " );

     // testar valor
        switch ( x )
        {
          case 0:
            method00 ( );
           break;
          case 1:
            method01 ( );
           break;
          case 2:
            method02 ( );
           break;
          case 3:
            method03 ( );
           break;
          case 4:
            method04 ( );
           break;
          default:
            IO_pause ( "ERRO: Valor invalido." );
        } // fim escolher
    }
    while ( x != 0 );

 // encerrar
    IO_pause ( "Apertar ENTER para terminar" );
    return ( 0 );
} // fim main( )
```

/*
---------- documentacao complementar

---------- notas / observacoes / comentarios

---------- previsao de testes

a.) 0
b.) 1
c.) 3
d.) 5
e.) -5

---------- historico

Versao	Data	Modificacao
0.1	__/__	esboco

---------- testes

Versao	Teste	
0.1	01. (OK)	identificacao de programa

***/**

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 – Exemplo0305.c.

17.) Editar mudanças no nome do programa e versão.
Incluir novo método, e na parte principal, incluir uma alternativa para executá-lo.
Uma forma mais compacta de repetição com variação decrescente será apresentada.
Prever novos testes.

```
/**
  Method05 - Repeticao com teste no inicio e variacao.
 */
void method05 ( )
{
 // definir dado
    int x = 0;
    int y = 0;

 // identificar
    IO_id ( "EXEMPLO0305 - Method05 - v0.0" );

 // ler do teclado
    x = IO_readint ( "Entrar com uma quantidade: " );

 // repetir (x) vezes
 //       inicio    teste   variacao
    for ( y  = x; y >= 1; y = y - 1 )
    {
     // mostrar valor atual
        IO_printf ( "%d\n", y );
    } // fim repetir

 // encerrar
    IO_pause ( "Apertar ENTER para continuar" );
} // fim method05 ( )
```

```
/*
  Funcao principal.
*/
int main ( )
{
 // definir dado
    int x = 0;

 // repetir até desejar parar
    do
    {
     // identificar
        IO_id ( "EXEMPLO0305 - Programa - v0.0" );

     // ler do teclado
        IO_println ( "Opcoes" );
        IO_println ( "0 - parar" );
        IO_println ( "1 - repeticao com teste no inicio                (decrescente)" );
        IO_println ( "2 - repeticao com teste no inicio                (  alternativo )" );
        IO_println ( "3 - repeticao com teste no inicio                (  crescente  )" );
        IO_println ( "4 - repeticao com teste no inicio e variacao (  crescente  )" );
        IO_println ( "5 - repeticao com teste no inicio e variacao (decrescente)" );
        IO_println ( "" );

        x = IO_readint ( "Entrar com uma opcao: " );

     // testar valor
        switch ( x )
        {
          case 0:
            method00 ( );
           break;
          case 1:
            method01 ( );
           break;
          case 2:
            method02 ( );
           break;
          case 3:
            method03 ( );
           break;
          case 4:
            method04 ( );
           break;
          case 5:
            method05 ( );
           break;
          default:
            IO_pause ( "ERRO: Valor invalido." );
        } // fim escolher
    }
    while ( x != 0 );

 // encerrar
    IO_pause ( "Apertar ENTER para terminar" );
    return ( 0 );
} // fim main( )
```

/*
---------- documentacao complementar

---------- notas / observacoes / comentarios

---------- previsao de testes

a.) 0
b.) 1
c.) 3
d.) 5
e.) -5

---------- historico

Versao	Data	Modificacao
0.1	__/__	esboco

---------- testes

Versao	Teste	
0.1	01. (OK)	identificacao de programa

***/**

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 – Exemplo0306.c.

21.) Editar mudanças no nome do programa e versão.
Incluir novo método, e na parte principal, incluir uma alternativa para executá-lo.
Uma forma de repetição sobre cadeia de caracteres será apresentada.
Prever novos testes.

```
/**
  Method06 - Repeticao sobre cadeia de caracateres.
 */
void method06 ( )
{
 // definir dado
    int x = 0;
    int y = 0;

    chars palavra = IO_new_chars ( STR_SIZE );
    int  tamanho   = 0;

 // identificar
    IO_id ( "EXEMPLO0306 - Method06 - v0.0" );

 // ler do teclado
    palavra = IO_readstring ( "Entrar com uma palavra: " );

 // repetir para cada letra
    tamanho = strlen ( palavra ) - 1;
//  OBS: A cadeia de caracteres iniciam suas posições em zero.

 //        inicio              teste   variacao
    for ( y  = tamanho; y >= 0; y = y - 1 )
    {
     // mostrar valor atual
        IO_printf ( "%d: [%c]\n", y, palavra [y] );
    } // fim repetir

 // encerrar
    IO_pause ( "Apertar ENTER para continuar" );
} // fim method06 ( )
```

```
/*
  Funcao principal.
*/
int main ( )
{
 // definir dado
    int x = 0;

 // repetir até desejar parar
    do
    {
     // identificar
        IO_id ( "EXEMPLO0306 - Programa - v0.0" );

     // ler do teclado
        IO_println ( "Opcoes" );
        IO_println ( "0 - parar" );
        IO_println ( "1 - repeticao com teste no inicio                (decrescente)" );
        IO_println ( "2 - repeticao com teste no inicio                ( alternativo  )" );
        IO_println ( "3 - repeticao com teste no inicio                (  crescente  )" );
        IO_println ( "4 - repeticao com teste no inicio e variacao (  crescente  )" );
        IO_println ( "5 - repeticao com teste no inicio e variacao (decrescente)" );
        IO_println ( "6 - repeticao sobre cadeia de caracteres      (decrescente)" );
        IO_println ( "" );

        x = IO_readint ( "Entrar com uma opcao: " );

     // testar valor
        switch ( x )
        {
          case 0:
            method00 ( );
           break;
          case 1:
            method01 ( );
           break;
          case 2:
            method02 ( );
           break;
          case 3:
            method03 ( );
           break;
          case 4:
            method04 ( );
           break;
          case 5:
            method05 ( );
           break;
          case 6:
            method06 ( );
           break;
          default:
            IO_pause ( "ERRO: Valor invalido." );
        } // fim escolher
    }
    while ( x != 0 );
```

```
 // encerrar
    IO_pause ( "Apertar ENTER para terminar" );
    return ( 0 );
} // fim main( )

/*
---------------------------------------------- documentacao complementar

---------------------------------------------- notas / observacoes / comentarios



---------------------------------------------- previsao de testes

a.)   "a"
b.)   "abc"
c.)   "abc def"

---------------------------------------------- historico

Versao      Data            Modificacao
  0.1       __/__           esboco

 ---------------------------------------------- testes

Versao      Teste
  0.1       01. ( OK )      identificacao de programa

*/
```

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 – Exemplo0307.c.

25.) Editar mudanças no nome do programa e versão.
Incluir novo método, e na parte principal, incluir uma alternativa para executá-lo.
Uma forma de repetição sobre cadeia de caracteres com variação crescente será apresentada.
Prever novos testes.

```
/**
  Method07 - Repeticao sobre cadeia de caracateres.
 */
void method07 ( )
{
 // definir dado
    int x = 0;
    int y = 0;

    char palavra [STR_SIZE];
    int  tamanho = 0;

 // identificar
    IO_id ( "EXEMPLO0307 - Method07 - v0.0" );

 // ler do teclado
    IO_printf ( "Entrar com uma palavra: " );
    scanf  ( "%s", palavra );

 // repetir para cada letra
    tamanho = strlen ( palavra );
//  OBS: A cadeia de caracteres iniciam suas posições em zero.

 //        inicio       teste       variacao
    for ( y  = 0; y < tamanho; y = y + 1 )
    {
     // mostrar valor atual
        IO_printf ( "%d: [%c]\n", y, palavra [y] );
    } // fim repetir

 // encerrar
    IO_pause ( "Apertar ENTER para continuar" );
} // fim method07 ( )
```

```
/*
  Funcao principal.
*/
int main ( )
{
 // definir dado
    int x = 0;

 // repetir até desejar parar
    do
    {
     // identificar
        IO_id ( "EXEMPLO0307 - Programa - v0.0" );

     // ler do teclado
        IO_println ( "Opcoes" );
        IO_println ( "0 - parar" );
        IO_println ( "1 - repeticao com teste no inicio                (decrescente)" );
        IO_println ( "2 - repeticao com teste no inicio                (  alternativo )" );
        IO_println ( "3 - repeticao com teste no inicio                (  crescente  )" );
        IO_println ( "4 - repeticao com teste no inicio e variacao (  crescente  )" );
        IO_println ( "5 - repeticao com teste no inicio e variacao (decrescente)" );
        IO_println ( "6 - repeticao sobre cadeia de caracteres      (decrescente)" );
        IO_println ( "7 - repeticao sobre cadeia de caracteres      (  crescente  )" );
        IO_println ( "" );

        x = IO_readint ( "Entrar com uma opcao: " );

     // testar valor
        switch ( x )
        {
          case 0:
            method00 ( );
           break;
          case 1:
            method01 ( );
           break;
          case 2:
            method02 ( );
           break;
          case 3:
            method03 ( );
           break;
          case 4:
            method04 ( );
           break;
          case 5:
            method05 ( );
           break;
          case 6:
            method06 ( );
           break;
          case 7:
            method07 ( );
           break;
          default:
            IO_pause ( "ERRO: Valor invalido." );
        } // fim escolher
    }
    while ( x != 0 );
```

```
 // encerrar
    IO_pause ( "Apertar ENTER para terminar" );
    return ( 0 );
} // fim main( )

/*
---------------------------------------------- documentacao complementar

---------------------------------------------- notas / observacoes / comentarios



---------------------------------------------- previsao de testes


a.)   "a"
b.)   "abc"
c.)   "abc def"


---------------------------------------------- historico

Versao      Data              Modificacao
  0.1       __/__             esboco

 ---------------------------------------------- testes

Versao      Teste
  0.1       01. ( OK )        identificacao de programa

*/
```

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 – Exemplo0308.c.

29.) Editar mudanças no nome do programa e versão.
Incluir novo método, e na parte principal, incluir uma alternativa para executá-lo.
Uma forma de repetição sobre intervalo de valores será apresentada.
Prever novos testes.

```
/**
  Method08 - Repeticao com intervalos.
 */
void method08 ( )
{
 // definir dado
    int inferior   = 0;
    int superior = 0;
    int x              = 0;

 // identificar
    IO_id ( "EXEMPLO0310 - Method08 - v0.0" );

 // ler do teclado
    inferior   = IO_readint ( "Limite inferior do intervalo: " );
    superior = IO_readint ( "Limite superior do intervalo : " );

 //              inicio           teste        variacao
    for ( x  = inferior; x <= superior; x = x + 1 )
    {
     // mostrar valor atual
        IO_printf ( "%d\n", x );
    } // fim repetir

 // encerrar
    IO_pause ( "Apertar ENTER para continuar" );
} // fim method08 ( )
```

```
/*
  Funcao principal.
*/
int main ( )
{
 // definir dado
    int x = 0;

 // repetir até desejar parar
    do
    {
     // identificar
        IO_id ( "EXEMPLO0308 - Programa - v0.0" );

     // ler do teclado
        IO_println ( "Opcoes" );
        IO_println ( "0 - parar" );
        IO_println ( "1 - repeticao com teste no inicio                (decrescente)" );
        IO_println ( "2 - repeticao com teste no inicio                (  alternativo )" );
        IO_println ( "3 - repeticao com teste no inicio                (  crescente  )" );
        IO_println ( "4 - repeticao com teste no inicio e variacao (  crescente  )" );
        IO_println ( "5 - repeticao com teste no inicio e variacao (decrescente)" );
        IO_println ( "6 - repeticao sobre cadeia de caracteres      (decrescente)" );
        IO_println ( "7 - repeticao sobre cadeia de caracteres      (  crescente  )" );
        IO_println ( "8 - repeticao com intervalos                     (  crescente  )" );
        IO_println ( "" );

        x = IO_readint ( "Entrar com uma opcao: " );

     // testar valor
        switch ( x )
        {
          case 0:
            method00 ( );
           break;
          case 1:
            method01 ( );
           break;
          case 2:
            method02 ( );
           break;
          case 3:
            method03 ( );
           break;
          case 4:
            method04 ( );
           break;
          case 5:
            method05 ( );
           break;
          case 6:
            method06 ( );
           break;
          case 7:
            method07 ( );
           break;
```

```
        case 8:
          method08 ( );
         break;
        default:
          IO_pause ( "ERRO: Valor invalido." );
       } // fim escolher
    }
    while ( x != 0 );

 // encerrar
    IO_pause ( "Apertar ENTER para terminar" );
    return ( 0 );
} // fim main( )

/*
---------------------------------------------- documentacao complementar

---------------------------------------------- notas / observacoes / comentarios



---------------------------------------------- previsao de testes

a.)   0 e 1
b.)   1 e  5
c.)   3 e  5
d.)  -5 e  5

---------------------------------------------- historico

Versao        Data              Modificacao
  0.1         __/__             esboco

 ---------------------------------------------- testes

Versao        Teste
  0.1         01. ( OK )        identificacao de programa

*/
```

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 – Exemplo0309.c.

33.) Editar mudanças no nome do programa e versão.
Incluir novo método, e na parte principal, incluir uma alternativa para executá-lo.
Uma forma de repetição sobre intervalo de valores com variação decrescente será apresentada.
Prever novos testes.

```
/**
  Method09 - Repeticao com intervalos.
 */
void method09 ( )
{
 // definir dado
    double inferior   = 0;
    double superior = 0;
    double passo     = 0;
    double x           = 0;

 // identificar
    IO_id ( "EXEMPLO0310 - Method09 - v0.0" );

 // ler do teclado
    inferior   = IO_readdouble ( "Limite inferior do intervalo : " );
    superior = IO_readdouble ( "Limite superior do intervalo : " );
    passo     = IO_readdouble ( "Variacao no intervalo (passo): " );

 //              inicio              teste          variacao
    for ( x  = superior; x >= inferior; x = x - passo )
    {
     // mostrar valor atual
        IO_printf ( "%lf\n", x );
    } // fim repetir

 // encerrar
    IO_pause ( "Apertar ENTER para continuar" );
} // fim method09 ( )
```

```
/*
  Funcao principal.
*/
int main ( )
{
 // definir dado
    int x = 0;

 // repetir até desejar parar
    do
    {
     // identificar
        IO_id ( "EXEMPLO0309 - Programa - v0.0" );

     // ler do teclado
        IO_println ( "Opcoes" );
        IO_println ( "0 - parar" );
        IO_println ( "1 - repeticao com teste no inicio                (decrescente)" );
        IO_println ( "2 - repeticao com teste no inicio                (  alternativo )" );
        IO_println ( "3 - repeticao com teste no inicio                (  crescente  )" );
        IO_println ( "4 - repeticao com teste no inicio e variacao (  crescente  )" );
        IO_println ( "5 - repeticao com teste no inicio e variacao (decrescente)" );
        IO_println ( "6 - repeticao sobre cadeia de caracteres      (decrescente)" );
        IO_println ( "7 - repeticao sobre cadeia de caracteres      (  crescente  )" );
        IO_println ( "8 - repeticao com intervalos                  (  crescente  )" );
        IO_println ( "9 - repeticao com intervalos                  (decrescente)" );
        IO_println ( "" );

        x = IO_readint ( "Entrar com uma opcao: " );

     // testar valor
        switch ( x )
        {
          case 0:
            method00 ( );
           break;
          case 1:
            method01 ( );
           break;
          case 2:
            method02 ( );
           break;
          case 3:
            method03 ( );
           break;
          case 4:
            method04 ( );
           break;
          case 5:
            method05 ( );
           break;
          case 6:
            method06 ( );
           break;
          case 7:
            method07 ( );
           break;
```

```
        case 8:
          method08 ( );
         break;
        case 9:
          method09 ( );
         break;
        default:
          IO_pause ( "ERRO: Valor invalido." );
      } // fim escolher
   }
   while ( x != 0 );

 // encerrar
   IO_pause ( "Apertar ENTER para terminar" );
   return ( 0 );
} // fim main( )

/*
---------------------------------------------- documentacao complementar

---------------------------------------------- notas / observacoes / comentarios



---------------------------------------------- previsao de testes

a.)   0 e  1, passo  1
b.)   1 e  5, passo  1
c.)   1 e  5, passo  2
d.)   3 e  5, passo  1
e.)   3 e  5, passo  2
f. )  -5 e  5, passo  1
g.)  -5 e  5, passo  2
h.)  -5 e  5, passo  5
 i.)  -5 e  5, passo -1

---------------------------------------------- historico

Versao        Data              Modificacao
  0.1         __/__             esboco

 ---------------------------------------------- testes

Versao        Teste
  0.1         01. ( OK )        identificacao de programa

*/
```

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 – Exemplo0310.c.

37.) Editar mudanças no nome do programa e versão.
Incluir novo método, e na parte principal, incluir uma alternativa para executá-lo.
Uma forma de repetição para confirmação de características de dados será apresentada.
Prever novos testes.

```
/**
  Method10 - Repeticao com confirmacao.
 */
void method10 ( )
{
 // definir dado
    double inferior   = 0;
    double superior = 0;
    double passo     = 0;
    double x           = 0;

 // identificar
    IO_id ( "EXEMPLO0310 - Method10 - v0.0" );

 // ler do teclado
    inferior = IO_readdouble ( "Limite inferior do intervalo : " );

 // repetir ate' haver confirmacao de validade
    do
    {
       superior = IO_readint ( "Limite superior do intervalo: " );
    }
    while ( inferior >= superior );

 // repetir ate' haver confirmacao de validade
    do
    {
       passo = IO_readdouble ( "Variacao no intervalo (passo): " );
    }
    while ( passo <= 0.0 );

 //              inicio            teste            variacao
    for ( x  = inferior; x <= superior; x = x + passo )
    {
     // mostrar valor atual
        IO_printf ( "%lf\n", x );
    } // fim repetir

 // encerrar
    IO_pause ( "Apertar ENTER para continuar" );
} // fim method10 ( )
```

```
/*
  Funcao principal.
*/
int main ( )
{
 // definir dado
    int x = 0;              // definir variavel com valor inicial

 // repetir até desejar parar
    do
    {
     // identificar
        IO_id ( "EXEMPLO0310 - Programa - v0.0" );

     // ler do teclado
        IO_println ( "Opcoes" );
        IO_println ( " 0 - parar" );
        IO_println ( " 1 - repeticao com teste no inicio                (decrescente)" );
        IO_println ( " 2 - repeticao com teste no inicio                (  alternativo )" );
        IO_println ( " 3 - repeticao com teste no inicio                (  crescente  )" );
        IO_println ( " 4 - repeticao com teste no inicio e variacao (  crescente  )" );
        IO_println ( " 5 - repeticao com teste no inicio e variacao (decrescente)" );
        IO_println ( " 6 - repeticao sobre cadeia de caracteres      (decrescente)" );
        IO_println ( " 7 - repeticao sobre cadeia de caracteres      (  crescente  )" );
        IO_println ( " 8 - repeticao com intervalos                  (  crescente  )" );
        IO_println ( " 9 - repeticao com intervalos                  (decrescente)" );
        IO_println ( "10 - repeticao com confirmacao                                 " );
        IO_println ( "" );

        x = IO_readint ( "Entrar com uma opcao: " );

     // testar valor
        switch ( x )
        {
          case 0:
            method00 ( );
           break;
          case 1:
            method01 ( );
           break;
          case 2:
            method02 ( );
           break;
          case 3:
            method03 ( );
           break;
          case 4:
            method04 ( );
           break;
          case 5:
            method05 ( );
           break;
          case 6:
            method06 ( );
           break;
          case 7:
            method07 ( );
           break;
          case 8:
            method08 ( );
           break;
```

```
        case 9:
          method09 ( );
         break;
        case 10:
          method10 ( );
         break;
        default:
          IO_pause ( "ERRO: Valor invalido." );
       } // fim escolher
   }
   while ( x != 0 );

 // encerrar
   IO_pause ( "Apertar ENTER para terminar" );
   return ( 0 );
} // fim main( )

/*
---------------------------------------------- documentacao complementar

---------------------------------------------- notas / observacoes / comentarios



---------------------------------------------- previsao de testes

a.)  [ 0.1 : 0.5 ] e passo =  0.1
b.)  [ 0.1 : 0.5 ] e passo =  0.1
c.)  [ 0.5 : 0.1 ] e passo =  0.1
d.)  [ 0.1 : 0.5 ] e passo = -0.1

---------------------------------------------- historico

Versao        Data              Modificacao
  0.1         __/__             esboco

 ---------------------------------------------- testes

Versao        Teste
  0.1         01. ( OK )        identificacao de programa

*/
```

Exercícios:
DICAS GERAIS: Consultar o Anexo C 02 na apostila para outros exemplos.
Montar todos os métodos em um único programa conforme o último exemplo.

01.) Incluir um método (Exemplo0311) para:
- ler uma palavra do teclado;
- mostrar as letras minúsculas.
  DICA: Definir um teste para determinar se um caractere é letra minúscula.

Exemplo: palavra = "PaLaVrA"

02.) Incluir um método (Exemplo0312) para:
- ler uma palavra do teclado;
- contar e mostrar apenas as letras minúsculas.

Exemplo: palavra = "PaLaVrA"

03.) Incluir um método (Exemplo0313) para:
- ler uma palavra do teclado;
- contar e mostrar as letras minúsculas percorrendo do fim para o início da palavra.

Exemplo: palavra = "PaLaVrA"

04.) Incluir um método (Exemplo0314) para:
- ler uma cadeia de caracteres do teclado;
- contar e mostrar todos símbolos que forem letras, ou maiúsculas ou minúsculas.

Exemplo: palavra = "P4LaVr@"

05.) Incluir um método (Exemplo0315) para:
- ler uma cadeia de caracteres do teclado;
- contar e mostrar todos os dígitos, percorrendo do fim para o início da cadeia de caracteres.

Exemplo: palavra = "P4LaVr@1"

06.) Incluir um método (Exemplo0316) para:
- ler uma cadeia de caracteres do teclado;
- contar e mostrar tudo o que não for dígito e também não for letra.

Exemplo: palavra = "P4LaVr@O!"

07.) Incluir um método (Exemplo0317) para:
- ler dois valores inteiros (a,b), limites para definirem um intervalo [a:b];
- ler uma quantidade (n) de valores inteiros a serem testados;
- ler outros tantos valores quantos os indicados pela quantidade, um (x) por vez;
- contar e mostrar quantos dentre esses valores lidos (x)
  os que forem múltiplos de 5, e pertençam ao intervalo [a:b].

Exemplo: [20: 60], e n = 7, com { 10, 20, 30, 41, 55, 60, 84 }

08.) Incluir um método (Exemplo0318) para:
- ler dois valores inteiros (a,b), limites para definirem um intervalo [a:b];
- ler uma quantidade (n) de valores inteiros a serem testados;
- ler outros tantos valores quantos os indicados pela quantidade, um (x) por vez;
- contar e mostrar quantos dentre esses valores lidos (x)
  os que forem múltiplos de 5, que não forem também múltiplos de 3,
  e pertençam ao intervalo [a:b].

Exemplo: [20: 60], e n = 7, com { 10, 20, 30, 41, 55, 60, 84 }

09.) Incluir um método (Exemplo0319) para:
- ler dois valores reais (a e b), o primeiro (a) menor que o segundo (b), confirmadamente,
  para definirem um intervalo aberto (a:b);
- ler a quantidade (n) de valores reais a serem testados, e
  ler outros tantos valores (x) quantos os indicados por essa quantidade;
- contar e mostrar todos os valores lidos, pertencentes ao do intervalo,
  cujas partes inteiras forem ímpares.
  DICA: Usar conformação de tipo (***type casting***) para isolar a parte inteira (***int***),
  antes de testar se é ímpar (ver Exemplo0110).

Exemplo: (2.4: 6.3), e n = 7, com { 1.0, 2.4, 3.6, 4.1, 5.5, 6.3, 8.4 }

10.) Incluir um método (Exemplo0320) para:
- ler dois valores reais (a e b), maiores que 0 e menores que 1, confirmadamente,
  para definirem um intervalo aberto (a:b);
- ler uma quantidade (n) de valores reais a serem testados, e
  ler outros tantos valores quantos os indicados por essa quantidade;
- contar e mostrar todos os valores lidos
  que tenham suas partes fracionárias fora do intervalo]a:b[.
  DICA: Usar conformação de tipo (***type casting***) para isolar a parte inteira (***int***),
  e obter a parte fracionária mediante a subtração da parte inteira, antes de testar.

Exemplo: (0.2: 0.6), e n = 7, com { 1.0, 2.8, 3.6, 4.1, 5.5, 6.9, 8.4 }

Tarefas extras

E1.) Incluir um método (Exemplo03E1) para:

- ler uma linha do teclado;
- separar em outra cadeia de caracteres e mostrar
  todos os símbolos não alfanuméricos (letras ou dígitos) na cadeia de caracteres.
  DICA: A leitura de uma linha inteira, incluindo espaços em branco,
  poderá ser feita por meio de IO_readln("_"), ou gets( ), embora menos recomendado.

  Exemplo: sequência = "P4LaVr@O! & pAl4vR1n#a"

E2.) .) Incluir um método (Exemplo03E2) para:

- ler uma cadeia de caracteres do teclado;
- dizer se a sequência contém apenas símbolos que não são letras.

  Exemplo: sequência = "4@0!1#"