

Tema: Introdução à programação VI
Atividade: Apontadores

INSTRUÇÕES:

- Desenvolver classes/métodos em C++ para atender às especificações abaixo.
- Providenciar a documentação essencial:
nome e matrícula,
identificação, objetivo, parâmetros e condições especiais,
se houver, e relatório de testes (exemplos de valores usados e condições testadas).

SUGESTÃO: Montar um menu para a escolha do método a ser testado
(ver modelo em Lista00.cpp).

Testes deverão ser realizados e os valores usados deverão
ser guardados no final do programa como comentários (`/* e */`).
O uso de recursão é opcional; se desejar utilizá-lo,
fazer também a implementação da forma não-recursiva.

0.) Editar programa em C, na mesma pasta, cujo nome será Exemplo1500.cpp, para testar definições de métodos a serem desenvolvidos:

```
/*
    Exemplo1500 - v0.0. - __ / __ / ____
    Author: _____
*/

// ----- preparacao

// dependências

#include <iostream>

// ----- definicoes globais

using namespace std;

// ----- metodos

/**
    Method_00 - nao faz nada.
*/
void method_00 ( )
{
    // nao faz nada
} // end method_00 ( )

/**
    Method_01 - Testar definicoes da classe.
*/
void method_01 ( )
{
    // definir dados

    // identificar
    cout << "\nMethod_01 - v0.0\n" << endl;

    // encerrar
    pause ( "Apertar ENTER para continuar" );
} // end method_01 ( )
```

```

// ----- acao principal

/*
Funcao principal.
@return codigo de encerramento
*/
int main ( int argc, char** argv )
{
    // definir dado
    int x = 0;          // definir variavel com valor inicial

    // repetir até desejar parar
    do
    {
        // identificar
        cout << "EXEMPLO1500 - Programa - v0.0\n" << endl;

        // mostrar opcoes
        cout << "Opcoes" << endl;
        cout << " 0 - parar" << endl;
        cout << " 1 - testar definicoes" << endl;

        // ler do teclado
        cout << endl << "Entrar com uma opcao: ";
        cin >> x;

        // escolher acao
        switch ( x )
        {
            case 0:
                method_00 ( );
                break;
            case 1:
                method_01 ( );
                break;
            default:
                cout << endl << "ERRO: Valor invalido." << endl;
        } // end switch
    }
    while ( x != 0 );

    // encerrar
    pause ( "Apertar ENTER para terminar" );
    return ( 0 );
} // end main ( )

```

/*

----- documentacao complementar

----- notas / observacoes / comentarios

----- previsao de testes

----- historico

Versao	Data	Modificacao
0.1	__/__/__	esboco

----- testes

Versao	Teste	
0.1	01. (OK)	identificacao de programa

*/

Exercícios:

DICAS GERAIS: Consultar o Anexo CPP 02 na apostila para outros exemplos.

Prever, realizar e registrar todos os testes efetuados.

- Desenvolver e testar cada um dos protótipos de métodos sugeridos abaixo, usando apenas apontadores.

Integrar as chamadas de todos os programas em um só.

01.)

```
/**  
    Funcao para acrescentar caractere ao final de uma cadeia  
    por meio de apontador.  
    @return apontador para a cadeia atualizada  
    @param s - apontador para cadeia de caracteres  
*/  
char* str_push_back ( char *s )
```

02.)

```
/**  
    Funcao para remover caractere do final de uma cadeia  
    por meio de apontador.  
    @return apontador para a cadeia atualizada  
    @param s - apontador para cadeia de caracteres  
*/  
char* str_pop_back ( char *s )
```

03.)

```
/**  
    Funcao para acrescentar caractere ao início de uma cadeia  
    por meio de apontador.  
    @return apontador para a cadeia atualizada  
    @param s - apontador para cadeia de caracteres  
*/  
char* str_push_front ( char *s )
```

04.)

```
/**  
    Funcao para remover caractere do início de uma cadeia  
    por meio de apontador.  
    @return apontador para a cadeia atualizada  
    @param s - apontador para cadeia de caracteres  
*/  
char* str_pop_front ( char *s )
```

05.)

```
/**
 * Funcao para procurar pela primeira ocorrencia de caractere em cadeia
 * por meio de apontador.
 * @return apontador para a primeira ocorrência; NULL, caso contrario
 * @param s - apontador para cadeia de caracteres
 * @param c - caractere a ser procurado
 */
char* str_chr ( char *s, char c )
```

06.)

```
/**
 * Funcao para separar caracteres ate' a ocorrencia de delimitador em cadeia
 * por meio de apontador.
 * @return apontador para caracteres; NULL, caso contrario
 * @param s - apontador para cadeia de caracteres
 * @param delimiter - caractere a ser procurado
 */
char* str_tok ( char *s, char delimiter )
```

Para os próximos exercícios considerar as seguintes definições de tipo/classe

```
typedef struct s_intArray { int length; int *data } intArray;
```

```
typedef intArray* ref_intArray;
```

07.)

```
/**
 * Funcao para comparar arranjos de inteiros
 * por meio de apontadores.
 * @return zero, se forem iguais;
 *         negativo, se o valor da diferenca for menor e estiver no primeiro arranjo
 *         positivo , se o valor da diferenca for maior e estiver no primeiro arranjo
 * @param p - apontador para inicio do primeiro arranjo
 * @param q - apontador para inicio do segundo arranjo
 */
int intArray_cmp ( ref_intArray p, ref_intArray q )
```

08.)

```
/**
 * Funcao para juntar arranjos de inteiros
 * por meio de apontadores.
 * @return apontador para inicio do arranjo com a uniao
 * @param p - apontador para inicio do primeiro arranjo
 * @param q - apontador para inicio do segundo arranjo
 */
ref_intArray intArray_cat ( ref_intArray p, ref_intArray q )
```

09.)

```
/**  
    Funcao para procurar pela primeira ocorrencia de valor em arranjo  
    por meio de apontador.  
    @return apontador para a primeira ocorrência; NULL, caso contrario  
    @param a - apontador para arranjo de inteiros  
    @param x - valor ser procurado  
*/  
ref_intArray intArray_seek ( ref_intArray a, int x )
```

10.)

```
/**  
    Funcao para separar sequencia de valores em arranjo  
    por meio de apontador.  
    @return apontador para inicio da sequencia de inteiros; NULL, caso contrario  
    @param a - apontador para arranjo de inteiros  
    @param start - posicao inicial  
    @param size - quantidade de dados  
*/  
ref_intArray intArray_sub ( ref_intArray a, int start, int size )
```

Tarefas extras

E1.)

```
/**  
    Funcao para intercalar arranjos de inteiros  
    por meio de apontadores.  
    @return apontador para inicio do arranjo com a uniao  
    @param p - apontador para inicio do primeiro arranjo  
    @param q - apontador para inicio do segundo arranjo  
*/  
ref_intArray intArray_merge ( ref_intArray p, ref_intArray q )
```

E2.)

```
/**  
    Funcao para intercalar arranjos de inteiros em ordem crescente  
    por meio de apontadores.  
    @return apontador para inicio do arranjo com a uniao  
    @param p - apontador para inicio do primeiro arranjo  
    @param q - apontador para inicio do segundo arranjo  
*/  
ref_intArray intArray_mergeUp ( ref_intArray p, ref_intArray q )
```