

== Laboratório 6 ==

1. Escreva um programa que inicializa dois objetos da classe `MeuNome`, sendo que um desses objetos é constante. O construtor da classe `MeuNome` recebe como argumento uma string que é utilizada para inicializar um atributo privado do tipo `string` chamado `nome`. A inicialização é feita por lista de inicializadores de membro. Após a criação dos dois objetos, o método `getNome` é invocado na função principal para os dois objetos, retornando o valor do atributo `nome` para ser impresso na tela. A classe `MeuNome` implementa duas versões sobrecarregadas do método `getNome`, uma para objetos constantes e outra para objetos não-constantes.
2. Reescreva o programa anterior, alterando o tipo do atributo privado `nome` de `string` para `char *`. Faça todas as alterações necessárias para que o programa continue funcionando da mesma forma descrita no exercício anterior. Note apenas que a memória deve ser alocada dinamicamente para que o atributo `nome` possa ser inicializado com a string recebida pelo construtor. Use a função `malloc` para tal. Já para liberação da memória, use a função `free` que deve ser invocada no destrutor da classe. Caso seja necessário, utilize o método `length()` da classe `string` para saber o tamanho da string.
3. Escreva um programa de um Relógio. Para isso, crie uma classe chamada `Relogio` que possui cinco atributos privados: `time_t hora`, `struct tm *infoHora`, `int segundos`, `int minutos` e `int horas`. Os tipos `time_t` e `struct tm` estão definidos no arquivo `time.h` que deve ser inserido. A classe `Relogio` ainda possui um construtor usado para acertar o horário. O horário é definido por um método chamado `resetHora`, que usa funções também definidas no arquivo `time.h`. O método `resetHora` está implementado como se segue:

```
void Relogio::resetHora () {  
    time (&hora);  
    infoHora = localtime (&hora);  
}
```

Saiba ainda que a estrutura `infoHora` possui um campos `tm_sec`, `tm_min` e `tm_hour` que armazenam inteiros contendo, respectivamente o valor dos segundos, dos minutos e das horas atuais. Além do método construtor e do método `resetHora`, a classe `Relogio` possui ainda métodos do tipo “set” e “get” para os atributos `segundos`, `minutos` e `horas` que serão usados na função principal para, respectivamente, reinicializar e obter o horário atual. Os métodos do tipo “set” podem ser usados em cascata e os métodos do tipo “get” são const.

4. Altere o programa anterior, incluindo a classe `PonteiroDeRelogio`. Para isso, altere a classe `Relogio` para que ela possua apenas objetos da nova classe `PonteiroDeRelogio` como atributos privados (um objeto para segundos, outro para minutos e outro para horas). A classe `Relogio` possui, neste exercício, um construtor que inicializa os atributos da classe `PonteiroDeRelogio`. Além disso, a classe `Relogio` oferece métodos do tipo “get” que retornam o valor dos

segundos, minutos e horas através de chamadas ao método `getHora` definido na classe `PonteiroDeRelogio`.

A classe `PonteiroDeRelogio` não possui construtor-padrão. Ao invés disso, ela possui um construtor que recebe uma string usada para inicializar o atributo privado `tipo`. Esse atributo define o tipo do ponteiro, ou seja, se é um ponteiro para segundos, minutos ou horas. Além disso, o construtor ainda invoca o método `resetHora`, como definido no exercício anterior, para inicializar os atributos privados `time_t hora` e `struct tm *infoHora`. Por fim, a classe `PonteiroDeRelogio` implementa o método `getHora` que testa o atributo `tipo` para depois retornar o campo da estrutura correspondente (`tm_sec`, `tm_min` ou `tm_hour`) contida na estrutura `infoHora`.

== Respostas ==

1.

```

/*****
/***** Programa Principal *****/

#include <iostream>
#include <string>
#include "meunome.h"

using namespace std;

int main() {
    MeuNome nome ("Miguel");
    const MeuNome nomeConst ("Elias");

    cout << nome.getNome () << endl;
    cout << nomeConst.getNome () << endl;

    return 0;
}

/*****
/***** Arquivo meunome.h *****/

#include <iostream>
#include <string>

using namespace std;

class MeuNome {
public:
    MeuNome (string);

    string getNome () const;
    string getNome ();

private:
    string nome;
};

/*****
/***** Arquivo meunome.cpp *****/

#include "meunome.h"

MeuNome::MeuNome (string n) : nome (n) { }

string MeuNome::getNome () const {
    cout << "Metodo const!" << endl;
    return nome;
}

```

```

string MeuNome::getNome () {
    cout << "Metodo nao-const!" << endl;
    return nome;
}

```

```

/*****/

```

2.

```

/*****/
/***** Programa Principal *****/

```

```

#include <iostream>
#include <string>
#include "meunome.h"

```

```

using namespace std;

```

```

int main() {
    MeuNome nome ("Miguel");
    const MeuNome nomeConst ("Elias");

    cout << nome.getNome () << endl;
    cout << nomeConst.getNome () << endl;

    return 0;
}

```

```

/*****/
/***** Arquivo meunome.h *****/

```

```

#include <iostream>
#include <string>

```

```

using namespace std;

```

```

class MeuNome {
public:
    MeuNome (string);
    ~MeuNome ();

    char *getNome () const;
    char *getNome ();

private:
    char *nome;
};

```

```

/*****/
/***** Arquivo meunome.cpp *****/

```

```

#include "meunome.h"

```

```

MeuNome::MeuNome (string n) :
    nome ((char *)malloc ((n.length () + 1)*sizeof (char))) {
    for (int i = 0; i < n.length (); i++)
        nome [i] = n [i];
    nome [n.length ()]='\0';
}

```

```

MeuNome::~MeuNome () {
    free (nome);
}

```

```

char *MeuNome::getNome () const {
    cout << "Metodo const!" << endl;
    return nome;
}

```

```

char *MeuNome::getNome () {
    cout << "Metodo nao-const!" << endl;
    return nome;
}

```

```

/*****/

```

3.

```

/*****
/***** Programa Principal *****/

#include <iostream>
#include "relogio.h"

using namespace std;

int main() {
    Relogio relogio;

    // Uso dos métodos set em cascata.
    relogio.setHoras ().setMinutos ().setSegundos ();

    cout << relogio.getHoras () << ":"
         << relogio.getMinutos () << ":"
         << relogio.getSegundos () << endl;

    return 0;
}

/*****
/***** Arquivo relogio.h *****/

#include <iostream>
#include <string>
#include <time.h>

using namespace std;

class Relogio {
public:
    Relogio ();

    void resetHora ();

    // Retorno de referência é usado para cascadeamento dos métodos
    Relogio & setSegundos ();
    Relogio & setMinutos ();
    Relogio & setHoras ();

    int getSegundos () const;
    int getMinutos () const;
    int getHoras () const;

private:
    time_t hora;
    struct tm *infoHora;
    int segundos, minutos, horas;
};

/*****
/***** Arquivo relogio.cpp *****/

#include "relogio.h"

Relogio::Relogio () {
    resetHora ();
}

void Relogio::resetHora () {
    time (&hora);
    infoHora = localtime (&hora);
}

Relogio & Relogio::setSegundos () {
    segundos = infoHora->tm_sec;
    return *this;
}

Relogio & Relogio::setMinutos () {
    minutos = infoHora->tm_min;
    return *this;
}

Relogio & Relogio::setHoras () {
```

```

        horas = infoHora->tm_hour;
        return *this;
    }
    int Relogio::getSegundos () const {
        return segundos;
    }
    int Relogio::getMinutos () const {
        return minutos;
    }
    int Relogio::getHoras () const {
        return horas;
    }
}

/*****/

```

4.

```

/*****/
/***** Programa Principal *****/

#include <iostream>
#include "relogio.h"

using namespace std;

int main() {
    Relogio relogio;

    cout << relogio.getHoras () << ":"
         << relogio.getMinutos () << ":"
         << relogio.getSegundos ()
         << endl;

    return 0;
}

/*****/
/***** Arquivo ponteiro.h *****/

#include <iostream>
#include <string>
#include <time.h>

using namespace std;

class Pontoeiro {
public:
    Pontoeiro (string);

    void resetHora ();
    int getHora ();

private:
    string tipo;
    time_t hora;
    struct tm *infoHora;
};

/*****/
/***** Arquivo ponteiro.cpp *****/

#include "ponteiro.h"

Pontoeiro::Pontoeiro (string t) : tipo (t) {
    resetHora ();
}

void Pontoeiro::resetHora () {
    time (&hora);
    infoHora = localtime (&hora);
}

int Pontoeiro::getHora () {
    if (!tipo.compare ("segundos")) {

```

```

        return infoHora->tm_sec;
    } else if (!tipo.compare ("minutos")) {
        return infoHora->tm_min;
    } else if (!tipo.compare ("horas")) {
        return infoHora->tm_hour;
    } else {
        cout << "Ponteiro nao existe!" << endl;
    }
}

/*****
*****/
/***** Arquivo relógio.h *****/
/*****/

#include <iostream>
#include "ponteiro.h"

using namespace std;

class Relogio {
public:
    Relogio ();

    int getSegundos ();
    int getMinutos ();
    int getHoras ();

private:
    Ponteiro segundos, minutos, horas;
};

/*****
*****/
/***** Arquivo relógio.cpp *****/
/*****/

#include "relogio.h"

Relogio::Relogio (): segundos ("segundos"), minutos ("minutos"), horas ("horas"){

int Relogio::getSegundos () {
    return segundos.getHora();
}
int Relogio::getMinutos () {
    return minutos.getHora();
}
int Relogio::getHoras () {
    return horas.getHora();
}

/*****
*****/

```