Data: 23/09/2008

Prof. Dr.: Jean Marcelo Simão

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ - Campus de Curitiba (Brasil) - **Departamento Acadêmico de Eletrônica** (DAELN). Disciplina: **Fundamentos de Programação** 2 - **IF62C**/Turma: **S12**. **Prof**: Jean M. **Simão**. Curso: **Engenharia Industrial Elétrica, ênfase Eletrônica/Telecomunicações**. **Prova sobre linguagem C++ – Prova da 1ª Parcial.**

Nome do Aluno: _____

Horário de Começo: _____ **Horário de Fim: _____**

Leia toda a prova antes de começar, pois os enunciados estão completados uns nos outros.

(Questão 1) Transforme o programa em linguagem C abaixo em um programa C++ orientado a objetos, onde os dados de equipamentos e seus tratamentos sejam programados em uma classe *Equipamento*. Os atributos relativos aos dados devem ser protegidos e corretamente iniciados quando da criação de objetos, por meio de mecanismo próprio para este fim. Ainda, a função membro *DescobreAntiguidade* deve ser sobrecarregada com uma outra função membro que considera como parâmetro o *mês atual*, além do *ano atual*. Por fim, certamente deverá existir uma classe principal nesta solução.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct Equipamento {  char nome [150];  int ano_fabricacao, mes_fabricacao, antiguidade; };

void DescobreAntiguidade ( struct Equipamento* equip, int ano_atual )
{
   equip->antiguidade = ano_atual –  ( equip->ano_fabricacao );
   printf ( " O %s tem %d  anos de fabricação. \n ", equip->nome, equip->antiguidade);
}

void main ( ) {
  struct Equipamento robo, prensa;
  robo.ano_fabricacao = 2006;    robo.mes_fabricacao = 6;    robo.antiguidade = -1;     strcpy( robo.nome, “robô”);
  prensa.ano_fabricacao = 1989; prensa.mes_fabricacao = 1; prensa.antiguidade = -1; strcpy( prensa.nome, “prensa”);
  DescobreAntiguidade ( &robo, 2008 ); DescobreAntiguidade ( &prensa, 2008 );   system ( “pause” );
}
```

Obs.: Utilizar apenas comandos de saída e (mesmo) de entrada efetivamente do C++ (i.e. *cout* e *cin*).

(Questão - 2) Elabore uma classe *Carro*, derivada da classe *Equipamento*. Esta classe *Carro* terá dois atributos privados chamado *chassi* e *placa* que deverão (nos seus objetos) ser acessíveis de alguma maneira, por algum método. A classe *Carro* terá também o operador de maior (>) sobrecarregado, de maneira que permita comparar os atributos antiguidade de dois objetos desta classe. Ainda, instancie (i.e. crie) dois objetos da classe *Carro* na classe *Principal* e os compare por meio do operador sobrecarregado.

(Questão - 3) Na classe *Principal*, cria um objeto de lista para (ponteiros de) objetos *Carros*. A lista pode ser baseada na definição das classes gabarito apresentada abaixo. Neste caso, não se faz necessário implementar os métodos delas. Entretanto, ressalta-se que a classe lista (gabarito) apresentada não possui um método para percorrer as listas instanciadas. Na verdade, isto deverá ser feito na classe *Principal*.

```
. . .
template<class TIPO>
class Lista
{ private:
    Elemento<TIPO>*  primeiro;
    Elemento<TIPO>*  atual; public:
    Lista ( );
    ~Lista ( );
    bool setElemento ( Elemento<TIPO>* elemento ) ;
    bool setInfo ( TIPO* info ) ;
    Elemento<TIPO>*  getPrimeiro ( ) ;
};
```

```
. . .
template<class TIPO>
class Elemento
{ private:
   Elemento<TIPO>*  proximo;
   TIPO*                info;
 public:
   Elemento ( );
   ~Elemento ( );
   void setProximo ( Elemento<TIPO>* p );
   Elemento<TIPO>* getProximo ( );
   void setInfo ( TIPO* i );
   TIPO* getInfo ( );  };
```

(Questão - 4) Construa um digrama de classes em UML das classes elaboradas nas questões anteriores.

A interpretação faz parte do conteúdo da prova! Inclua comentários (se for o caso) para deixar claras suas decisões.