Prof.: Jean Marcelo Simão

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ (UTFPR)
Campus de Curitiba-PR (Brasil) – DAELN - **Departamento Acadêmico de Eletrônica**.

Curso: **Engenharia Industrial Elétrica, ênfase Eletrônica/Telecomunicações**.
Disciplina: **Fundamentos de Programação 2 – IF62C** – Turma: **S11**.
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 relativos às pessoas e o tratamento destes sejam programados em uma classe. Além do mais, os dados devem ser programados de forma que não sejam acessíveis a partir de outros partes do programa. Entretanto, eles devem ser acessíveis por meio de herança. Estes dados devem ainda ser corretamente iniciados, quando da criação de objetos, por meio de mecanismo próprio para este fim.

```
#include <stdio.h>
struct Pessoa
{
   int dia;
   int mes;
   int ano;
   int idade;
};

int Calc_Idade ( struct Pessoa p, int dia, int mes, int ano )
{
   int idade = ano - p.ano;
   if (p.mes > mes)
   {
      idade = idade - 1;
   }
   else
   {
       if (p.mes == mes)
       {
          if (p.dia > dia)
          {
             idade = idade - 1;
          }
       }
   }
   return idade;
}

int main()
{
  struct Pessoa Einstein, Newton;
```

```
  Einstein.dia   = 14;
  Einstein.mes =  3;
  Einstein.ano  = 1879;

  Newton.dia    = 4;
  Newton.mes  = 1;
  Newton.ano   = 1643;

  Einstein.idade = Calc_Idade (Einstein, 8, 2, 2007);
  Newton.idade  = Calc_Idade (Newton, 8, 2, 2007);

  printf ("A idade de Einstein seria %d \n", Einstein.idade);
  printf ("A idade de Newton seria %d  \n", Newton.idade);
  getchar();
  return 0;
}
```

(Questão - 2) Considerando a classe abaixo, responda objetivamente as perguntas:

```
class Principal
{
private:
   Pessoa Einstein;
   Pessoa Newton;
public:
   Principal();
   void Executar();
};
```

A - Por que deve-se ter uma classe *Principal* ao invés de instanciar objetos diretamente na *main()*?

B - Os elementos *Einstein* e *Newton* são *classes*, *objetos* ou *ponteiros para objetos* definidos no escopo da classe *Principal*?

C - Qual é o relacionamento de *Einstein* ou *Newton* para com a Classe *Principal*? Relacionamento simples (associação), agregação ou herança?

D - Por que há elementos públicos e elementos privados na classe *Principal*? Qual é a principal diferença entre elementos públicos e privados em geral?

E - Qual é função do método *Principal()*? Como métodos desta natureza se chamam?

(Questão - 3) Elabore uma classe *Disciplina* e outra *Aluno*. A Classe *Disciplina* deve ser capaz de ter uma lista de objetos instanciados a partir da classe *Aluno*, sendo esta lista de tamanho variável. A classe *Aluno*, por sua vez, será derivada da classe *Pessoa*, normalmente elaborada na questão 1. Ainda, a classe *Disciplina* deve ter um método para listar os *Alunos* relacionados, bem como outro para incluí-los. Finalmente, salienta-se que um *Aluno* pode participar de diversas listas de *Disciplinas*. **Faça, primeiramente, um modelo em UML e, depois, o código em C++**.

Obs.: Utilizar alocação dinâmica de memória para resolver esta questão, no que tange o código.

(Questão - 4) Escreva um método virtual na classe *Pessoa*, chamado de *setIdentificacao*, que sirva para armazenar o valor de seu RG (Registro Geral). Já na classe *Aluno*, faça com que o *setIdentificacao* seja um método que sirva para armazenar o valor do RA (Registro Acadêmico) do aluno. Uma vez feito isto, diga se isto se constitui em polimorfismo e justifique sua resposta.

(Questão - 5) A título de exemplo, considere que na classe *Pessoa* existe os seguintes 'métodos':

- **void** Calc_Idade **(int** diaAT**, int** mesAT**, int** anoAT **);**
- **void** Calc_Idade **(int** anoAT **);**

O exemplo demonstra dois métodos de mesmo nome, mas com parâmetros diferentes. Neste sentido, como se chama a existência de dois ou mais métodos em uma mesma classe com o mesmo nome e com conjuntos de parâmetros diferentes?

(Questão - 6) Considerando o código abaixo, responda objetivamente:
a) Para que serve um *template* ou gabarito?
b) Onde este código estaria implementado, no *.h* ou no *.cpp*?
c) Onde a implementação dos métodos estaria, no *.h* ou no *.cpp*?
d) Este código é suficiente para se ter listas efetivamente genéricas?

```
#include <stdlib.h>
template<class TIPO>
class Elemento
{
private:
   Elemento<TIPO>*  proximo;
   Elemento<TIPO>*  anterior;
   TIPO*            info;
   char        nome[150];

public:
   Elemento();
   ~Elemento();

   void setProximo ( Elemento<TIPO>* p );
   Elemento<TIPO>* getProximo ( );

   void setAnterior ( Elemento<TIPO>* a );
   Elemento<TIPO>* getAnterior ( );

   void setInfo ( TIPO* i );
   TIPO* getInfo ( );

   void setNome ( char* n );
   char* getNome ( );
};
```

(Questão – 7) Considerando a relação definida entre as classes *Disciplina* e *Alunos* (na questão 3), elaborar uma solução para armazenar as notas (1ª parcial, 2ª parcial e final) e o número de faltas de cada Aluno em cada Disciplina.