Aula 9

Objectivo: Herança Múltipla

Problema 1

Construa as seguintes classes: `PESSOA`, `PROFESSOR`, `ESTUDANTE` e `ASSISTENTE`, com pelo menos as seguintes interfaces:

- `PESSOA`: nome e idade
- `PROFESSOR`: departamento e lista de cadeiras
- `ESTUDANTE`: curso e grau académico pretendido

A classe `ASSISTENTE` deve ser feita herdando das classes `PROFESSOR` e `ESTUDANTE`.

1.1. Faça um programa simples que permita testar devidamente as classes desenvolvidas.

Problema 2

Continuando o problema 1 da aula 4, acrescente as classes (abstractas): `FIGURA_ABERTA` e `FIGURA_FECHADA`, definindo o método `perimetro` apenas para descendentes desta última.

2.1. Acrescente a classe `FIGURA_COMPOSTA`. Este tipo de figuras consiste numa lista de figuras (razão pela qual se torna útil que ela seja descendente quer da classe `FIGURA_ABERTA` quer de `LISTA_DE_FIGURAS`).

Problema 3

Pretende-se construir uma biblioteca de classes de armazenamento de informação. Todas as classes desta biblioteca são descendentes da classe (abstracta) `COLECCAO`, e – nesta primeira abordagem – ir-se-á restringir o tipo de informação armazenável nestas classes, como sendo objectos de classes descendentes da classe (abstracta): `ELEMENTO_DE_COLECCAO`.

```
class COLECCAO
{
public:

  virtual void adiciona(ELEMENTO_DE_COLECCAO& e) = 0;
  virtual ELEMENTO_DE_COLECCAO& actual(void) = 0;
  virtual void retira(void) = 0;
  virtual int vazia(void) = 0;
};

class ELEMENTO_DE_COLECCAO
{
public:

  virtual void define(void* valor) = 0;
  virtual void* valor(void) = 0;
};
```

3.1. Acrescente as classes abstractas `STACK` e `QUEUE`.

3.2. Acrescente a classe (não abstracta) `ARRAY`.

3.3. Construa as classes `STACK_COM_ARRAY` e `QUEUE_COM_ARRAY`, fazendo com que herdem a interface pública da classe abstracta respectiva, e a implementação da classe `ARRAY`.