| | |
|---|---|
| **Iniciada** | sexta, 12 de fevereiro de 2021 às 11:44 |
| **Estado** | Terminada |
| **Terminada em** | sexta, 12 de fevereiro de 2021 às 12:24 |
| **Tempo gasto** | 40 minutos |

**Pergunta 1**

Respondida

Nota: 20,00

Pretende-se uma classe que represente um livro de notícias de um determinado ano, envolvendo pessoas importantes. O livro armazenará as notícias, que farão parte integrante do livro, e referirá as tais pessoas, cujos objetos se assume que já existem e pertencem ao programa. As notícias não referem diretamente as pessoas. O livro é que refere as pessoas, mas de uma forma global, sem especificar que notícia diz respeito a que pessoa. É uma espécie de anuário que contém tudo o que aconteceu num ano, e diz quem esteve envolvido em coisas que aconteceram nesse ano, sem especificar que pessoa em qual coisa.

A classe seguinte representa as pessoas. Já existe e só deve alterá-la se achar que precisa. Se não precisar, nem sequer deve transcrever o seu código para a resposta.

```
class Pessoa {
   // …
public:
   Pessoa(string nome, int BI, int NIF);
   // …
};
```

A classe **Noticia** contém informações acerca de um acontecimento (sem registar diretamente as pessoas envolvidas - isso será feito pelo livro). Já existe e só deve alterá-la se achar que precisa.

```
class Noticia {
    string descricao, local;
    int ano, mes, dia;
public:
   Noticia (string ac, string loc, int a, int m, int d) { /* … */ }
   // pode acrescentar métodos (apenas métodos) desde que faça sentido
};
```

A classe **Livro** deve suportar as seguintes funcionalidades:

- Garantir que um novo livro só é criado mediante a indicação do ano a que se refere.
- Registar uma nova notícia, dada a informação que a descreve e a indicação da pessoa envolvida, cujo objeto já existe. Se esta pessoa ainda não tiver sido referida no livro, então passa a constar. Esta operação só tem efeito se a notícia for do ano a que o livro se refere.
- Apagar todas as notícias de um determinado mês (não é necessário actualizar as pessoas envolvidas no livro).

A classe deve ser autocontida e os seus objetos devem ser compatíveis com os usos habituais em que os

objectos de C++ podem ser usados.

Faça a classe **Livro**. Não precisa separar o código em .h/.cpp. Não precisa de incluir header files.

```
class Livro{
private:
vector <Pessoas>  vp

}
```

Pergunta 2
Respondida
Nota: 3,00

b) Imagine que existe ainda uma classe controlador, usada para controlar os vários dispositivos de cozinhas, tal como a descrita abaixo:

```
class Controlador {
   vector <Cozinhador *> dispositivos;
   //...
public:
   //...
};
```

Quando este controlador é destruído destrói também cada um dos dispositivos a ele associado. Indique o código que deve ser acrescentado ao header das outras classes (incluindo a Cozinhador) de modo a que seja possível fazer corretamente a cópia de um controlador. Para cada classe basta indicar se a declaração deve ser private, protected ou public, e a declaração propriamente dita.

```
class Cozinhador {
    int id;
public:
    Cozinhador(int p) : id(p){};
    int getID() const { return id; };
    virtual void aquece(int temperatura, int minutos) =0;
    virtual ~Cozinhador(){};
    virtual Cozinhador * clone() const =0;
};


class Exaustor {
private:
    int ligado; // 1-> ligado
    int minutos;
public:

    Exaustor(int estado) {
        ligado = 0;
    };

    void liga(int minutos){};

};

class Fritadeira : public Cozinhador {
    Exaustor e;
    int capacidade;
    int temperatura;

public:
    Fritadeira(Exaustor a, int c, int id) : e(a), capacidade(c), Cozinhador(id){
    temperatura =0;
    };
    void aquece( int temperatura , int minutos ){
        e.liga(minutos);
        /*...*/
    };

    Cozinhador *clone() const{
```

**Pergunta 3**
Respondida
Nota: 12,00

Numa cozinha automatizada existem vários eletrodomésticos de cozinha que servem para aquecer alimentos.
Um eletrodoméstico cuja função é cozinhar possui sempre uma função "aquece(temperatura, minutos)" que permite indicar por quanto tempo, e a que temperatura esse aparelho deve aquecer.

Dada a seguinte declaração de um dispositivo genérico que sirva para cozinhar alimentos:

```
class Cozinhador {
	int id;
public:
	Cozinhador(int p) : id(p) {}
	int getId() const { return id; }
	virtual void aquece(int temperatura, int minutos) = 0;
	virtual ~Cozinhador(){}
};
```

Existem diversos dispositivos que permitem cozinhar alimentos, que diferem entre si nas ações específicas realizadas durante o aquecimento da comida. Por exemplo:
- fritadeira - tem determinada capacidade; consegue comunicar com o exaustor; em resposta ao comando de aquecer pede ao exaustor que se ligue, e começa a aumentar a temperatura; Um exaustor pode ser associado à fritadeira em qualquer momento da existência da fritadeira.

O exaustor sabe responder a pedidos da forma "liga(minutos)". Podem existir várias fritadeiras na cozinha e todos conseguem comunicar com o exaustor. O exaustor conhece a fritadeira e consegue pedir-lhe que se desligue.

a) Apresente a declaração das classes que representam os eletrodomésticos acima referidos (ficheiros .h). Não deve alterar de nenhum modo a classe já fornecida. Não deve implementar função nenhuma – apresente apenas os ficheiros .h. Não precisa de especificar using namespaces nem includes excepto se forem includes e outras declarações de coisas suas.

```
class Cozinhador {
int id;
public:
Cozinhador(int p) : id(p){};
int getID() const { return id; };
```

```
virtual void aquece(int temperatura, int minutos) =0;
virtual ~Cozinhador(){};
};

class Exaustor {
private:
int ligado; // 1-> ligado
int minutos;
public:
Exaustor(int estado) {
ligado = 0;
};

void liga(int minutos){};
};
class Fritadeira : public Cozinhador {
Exaustor e;
int capacidade;
int temperatura;

public:
Fritadeira(Exaustor a, int c, int id) : e(a), capacidade(c), Cozinhador(id){
temperatura =0;
};
void aquece( int temperatura , int minutos ){
e.liga(minutos);
/*...*/
};
};
```

« PREVIOUS ACTIVITY
Exame de Recurso parte 2 de 3

NEXT ACTIVITY
Exam 2nd call - Part 1 of 3 »