ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO
Departamento de Engenharia de Computação e Sistemas Digitais

PCS 3111 - Laboratório de Programação Orientada a Objetos para Engenharia Elétrica

Aula 05 – Construtores e Destrutores

Atenção

Código inicial a ser usado na resolução dos exercícios encontra-se **disponível no e-Disciplinas**.

Exercício 01

Baseado nas implementações das classes `Pessoa` e `Atividade` das aulas anteriores, modifique os arquivos fornecidos de modo a adicionar construtores nas classes. Ao final das modificações, os arquivos de cabeçalho (.h) devem apresentar os métodos descritos abaixo (além dos atributos convenientes que devem ser alterados ou adicionados em relação ao código fornecido):

```
class Pessoa {
public:
    Pessoa(string nome, int horasDiarias);

    string getNome();
    int getHorasDiarias();

    void imprimir();
};
```

```
class Atividade {
public:
    Atividade(string nome, int horasNecessarias, int maxValores);

    string getNome();
    int getHorasNecessarias();
    int getQuantidade();
    int getDuracao();

    bool adicionar(Pessoa* p);
    void imprimir();
};
```

Adote as convenções a seguir:

- Na classe `Pessoa`, implemente o seu construtor. Este recebe o nome e as horas diárias como parâmetros e atribui, corretamente, ao respectivo atributo.
- O construtor da classe `Atividade` deve ser implementado de modo que:

- Os atributos `nome`, `horasNecessarias` e `quantidade` sejam inicializados adequadamente.
- O vetor de pessoas deve ser alocado dinamicamente, utilizando o parâmetro `maxValores` como novo tamanho, valor que deve ser armazenado em um atributo. Portanto **não** use mais o define NUMERO_MAXIMO_VALORES. (Atenção: devido a essa alteração, os métodos da classe `Atividade` que lidam com o vetor de pessoas precisam ser modificados!).

Dica: no `main`, crie instâncias das classes e teste suas funcionalidades.

Exercício 02

Implemente agora os destrutores das classes `Pessoa` e `Atividade`. Com a inclusão dos destrutores, os arquivos de cabeçalho apresentarão os seguintes métodos:

```
class Pessoa {
public:
    Pessoa(string nome, int horasDiarias);
    ~Pessoa();

    string getNome();
    int getHorasDiarias();

    void imprimir();
};
```

```
class Atividade
public:
    Atividade(string nome, int horasNecessarias, int maxValores);
    ~Atividade();

    string getNome();
    int getHorasNecessarias();
    int getQuantidade();
    int getDuracao();

    bool adicionar(Pessoa* p);
    void imprimir();
};
```

A implementação dos destrutores deve atender as seguintes especificações:

- O destrutor de `Pessoa` deve imprimir uma mensagem com o seguinte formato:

```
Pessoa <nome> destruida
```

Liberando os recursos utilizados.

Exemplo: `Pessoa Daniel destruida`

- O destrutor de `Atividade` deve imprimir uma mensagem com o seguinte formato:

```
Atividade <nome> de tamanho <maxValores> com <quantidade> pessoa(s) destruida
```

Com `maxValores` e `quantidade` referentes à `Atividade` destruída.

Exemplo: `Atividade EP1 de tamanho 2 com 1 pessoa(s) destruida`

- Ao chamar o destrutor de uma `Atividade`, todas as pessoas contidas nessa `Atividade` também devem ser destruídas. Atente-se a ordem com a qual esses objetos têm de ser destruídos!

Exercício 03

Implemente o seguinte método na classe `Atividade`:

```
Pessoa* getLider() const;
```

Esse método retorna um ponteiro para a pessoa que apresenta a maior quantidade de horas diárias. Na hipótese desse cenário acontecer mais de uma vez, deve-se retornar a primeira ocorrência. Caso a `Atividade` não disponha de pessoas, deve-se retornar NULL.

Note que o método é constante. Tente, dentro do método, modificar algum atributo da `Atividade` e verifique o que acontece.

Testes do Judge

Exercício 1

- Pessoa: construtor e getters
- Pessoa: `imprimir`
- Atividade: construtor e getters
- Atividade: `adicionar` com vetor vazio
- Atividade: `adicionar` com vetor parcialmente preenchido
- Atividade: `adicionar` com vetor cheio
- Atividade: `getDuracao`

Exercício 2

- Destrutor da Atividade destrói todas as pessoas
- Pessoa: Destrutor

Exercício 3

- `getLider` no início
- `getLider` no meio
- `getLider` no fim
- `getLider` com mais de uma ocorrência
- `getLider` sem pessoas