

 <p>INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA PIAUÍ</p>	<p>INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO PIAUÍ</p> <p>Curso: ADS</p> <p>Disciplina: Programação Orientada a Objetos</p> <p>Professor: Ely</p>
--	--

Avaliação

1. (0,5) Crie 3 classes conforme abaixo:
 - a. Crie uma classe Perfil que tenha `_id` (number), `_nome`, e `_email`, atributos como privados e inicializados em um construtor. Adicione métodos get de leitura;
 - b. Crie uma classe Postagem com `_id` (number), `_texto`, `_curtidas` (number), `_descurtidas` (number), `_data` e `_perfil` (classe Perfil criada anteriormente) como atributos privados, atributos inicializados no construtor e métodos de get de leitura;
 - c. Adicione à classe Perfil um atributo privado chamado `_postagens`:
`Postagem[]`
 - d. Crie uma classe chamada PostagemAvancada que além dos atributos de postagem, possua mais dois atributos privados: um array de strings chamado `_hashtags` e um outro atributo chamado `_visualizacoesRestantes` (number) que indica quantas vezes ela poderá ser exibida até expirar.
2. (1,5) Crie os seguintes métodos:
 - a. Na classe Postagem:
 - i. `curtir()`: void - incrementa em um o atributo `curtidas`;
 - ii. `descurtir()`: void - incrementa em um o atributo `_descurtidas`;
 - iii. `ehPopular()`: boolean - retorna true caso a quantidade de `_curtidas` seja 50% maior que a quantidade de `_descurtidas`.
 - b. Na classe PostagemAvancada:
 - i. `adicionarHashtag(hashtag: string)`: void - adiciona uma string ao array de `_hashtags`;
 - ii. `existeHashtag(hashtag: string)`: boolean - retorna true caso a postagem tenha a hashtag e false caso contrário;
 - iii. `decrementarVisualizacoes()`: void – decrementa em um a quantidade de vezes que a postagem foi visualizada.

3. (1,0) Crie uma classe chamada RepositorioDePerfis com:
 - a. Um atributo **_perfis: Perfil[]** privado;
 - b. incluir(perfil: Perfil): void - cadastra um perfil no array;
 - c. consultar(id: number, nome: string, email: string): Perfil - onde um, dois ou os 3 parâmetros são passados e retorna o perfil, caso seja encontrado no array. O método retorna null caso contrário.
4. (1,0) Crie uma classe chamada RepositorioDePostagens com:
 - a. Um atributo **postagens: Postagem[]** privado;
 - b. incluir(postagem: Postagem): void - cadastra uma postagem no array. Ao incluir uma postagem, adicione também ao array de postagens do Perfil;
 - c. consultar(i): Postagem[] - onde um, dois ou, três ou quatro parâmetros são passados e retorna os perfis encontrados e são consultadas pela hashtag apenas as postagens que forem do tipo PostagemAvancada;
5. (2,0) Crie uma classe chamada RedeSocial que tenha:
 - a. Dois atributos privados do tipo RepositorioDePostagens e RepositorioDePerfis;
 - b. Os métodos:
 - i. incluirPerfil(perfil: Perfil): void - chama o método incluir do atributo repositorioDePerfis validando se já há algum com mesmo id ou nome ou e-mail e que todos os atributos estejam preenchidos;
 - ii. consultarPerfil(id: number, nome: string, email: string): Perfil -chama o método da classe repositorioDePerfis;
 - iii. incluirPostagem(postagem: Postagem): void chama o método incluir do repositorioDePostagens desde não exista uma postagem com mesmo id e que todos os atributos estejam preenchidos;
 - iv. consultarPostagens(id: number, texto: string, hashtag: string, perfil: Perfil): Postagem[] - chama o método da classe repositorioDePostagens;
 - v. curtir(idPostagem: number): void - pesquisa uma postagem pelo id e chama o método curtir da classe postagem;
 - vi. descurtir(idPostagem: number): void - pesquisa uma postagem pelo id e chama o método descurtir da classe postagem;
 - vii. decrementarVisualizacoes(postagem: PostagemAvancada): void – chama o método decrementarVisualizacoes() da classe PostagemAvancada, não deixando o contador ficar negativo;
 - viii. exibirPostagensPorPerfil(id: number): Postagem[] – consulta todas as postagens de um perfil, filtra as postagens que ainda podem ser

exibidas, decrementa as visualizações postagens avançadas chamando o método da própria classe (`this.decrementarPostagem(postagem)`) e retorna um array de postagens;

- ix. `exibirPostagensPorHashtag(hashtag: string): PostagemAvancada[]` – consulta todas as postagens avançadas por hashtag, decrementa as visualizações postagens avançadas chamando o método da própria classe (`this.decrementarPostagem(postagem)`), decrementa as visualizações e retorna um array de postagens.

- 6. (1,5) Crie uma classe App que possua uma classe RedeSocial como atributo privado e exiba todas as opções da rede social em menus com um loop. Uma das opções deve ser a de sair. Permita a entrada de dados e retorne feedbacks.

Para as questões a seguir, implemente além da opção na classe App, os métodos necessários nas classes básicas, nos repositórios e na classe rede social.

- 7. (1,0) Adicione persistência em arquivos para perfis e postagens e permita a carga e salvamento ao inicializar e finalizar a aplicação.

- 8. (1,5) Implemente as funcionalidades:

- a. Exibir as postagens populares que ainda podem ser exibidas;
- b. Exibir as hashtags mais populares, ou seja, as que estão presentes em mais postagens;

Uma possibilidade é refatorar toda a aplicação a classe para ter um array de classes Hashtag na classe PostagemAvancada, onde ao cadastrar uma postagem pela rede social, as hashtags sejam cadastradas e um contador de cada uma seja atualizado.

- c. 4 funcionalidades na classe App além das solicitadas.

Questão adicionada em função do adiamento solicitado para alguns alunos:

- 9. Implementar um relacionamento entre perfis e criar as funcionalidades:

- a. Seguir perfil, não permitindo que siga mais de uma vez o mesmo perfil;
- b. Deixar de seguir perfil;
- c. Listar seguidores;
- d. Exibir postagens ou timeline de um seguidor em ordem decrescente de data de postagem.

Boa prova