Descrição de Exercício
Strings e Arrays

Objetivo: Este exercício, dividido em três módulos, tem como objetivo a familiarização com objetos do tipo *String* e com *Arrays*.
Tempo Estimado: 2h 30min

Strings

1. Utilizar a IDE para importar o projeto `ExercicioStringsArrays`, que está dentro da pasta `exercicios\StringsArrays\startupkit`.
2. Criar dentro deste projeto uma nova classe, chamada `StringTest`. Esta classe conterá apenas um método `main`.
3. Declarar no método `main` duas Strings (`s1` e `s2`). Inicializar cada uma delas utilizando o construtor explícito (usando o operador `new`) da classe String que recebe como parâmetro apenas uma String ("André" ou seu nome, por exemplo). Colocar o mesmo parâmetro para a inicialização de ambas as Strings.
4. Imprimir na tela a soma dos tamanhos de `s1` e `s2` (dica: utilizar o método `length()` da classe String).
5. Imprimir na tela se o conteúdo de `s1` e `s2` é o mesmo. Para isso, utilizar o método `equals()`.
6. Imprimir na tela se `s1` e `s2` são o mesmo objeto. Para isso, utilizar o operador de igualdade (==). Você consegue compreender a diferença entre o resultado do item 5 e o deste item?
7. Declarar agora duas novas Strings (`s3` e `s4`). Inicialize-as não utilizando um construtor como no item 3, e sim atribuindo diretamente a elas um literal String, como você faria normalmente com uma variável de tipo primitivo.
8. Imprimir na tela se `s3` e `s4` possuem mesmo conteúdo (`equals`) e se são o mesmo objeto (==). Analisar os resultados.
9. Atribuir um novo valor a `s3` e repetir o item 8. Analisar os resultados.

Enum (Projeto)

Parte 1

1. Criar um tipo enum intitulado `TipoCliente`. Este enum deve classificar o cliente como VIP, CLASS, ESPECIAL.
2. Modificar a classe cliente para que ele contemple o tipo do cliente (Modificar o construtor e adicionar os métodos get e set).

Arrays (Projeto)

Parte 1

1. Criar uma nova classe intitulada `RepositorioClientesArray`. Esta classe deve conter como atributos um *array* de `Clientes`, um índice (inteiro) que identifica o tamanho atual do *array* e uma variável estática do tipo inteiro, intitulada

(`tamanhoCache`), que armazena a capacidade máxima do *array*. Inicialize-a com 100.
2. Criar um construtor para `RepositorioClientesArray`, que não recebe nenhum parâmetro e inicializa o índice e o *array* de clientes.
3. Criar o método `inserir`, que recebe um cliente como parâmetro e o adiciona ao *array* de clientes. Observe que a variável índice (que identifica o tamanho atual do *array*) também deve ser atualizada.
4. Criar o método `procurarIndice`, que recebe como parâmetro o cpf de um cliente e retorna a posição desse cliente no *array* de clientes. Este método deve retornar (-1) caso o cliente não seja encontrado.
5. Criar o método `existe`, que recebe como parâmetro o cpf de um cliente e retorna um `boolean` identificando se o cliente existe ou não no *array* de clientes. Dica: utilize o método `procurarIndice` aqui.
6. Criar o método `procurar`, que recebe como parâmetro o cpf de um cliente e retorna o cliente correspondente. Caso o cliente não exista, imprimir uma mensagem de erro na tela. Dica: utilize os métodos `existe` e/ou `procurarIndice` aqui.
7. Criar o método `atualizar`, que recebe como parâmetro um cliente (que substituirá o cliente que possui o mesmo cpf no *array* de clientes). Caso o cliente a ser substituído não exista, imprimir uma mensagem de erro na tela. Dica: utilize os métodos `existe` e/ou `procurarIndice` aqui.
8. Criar o método `remover`, que recebe como parâmetro o cpf do cliente a ser removido do *array*. Caso o cliente a ser removido não exista, imprima uma mensagem de erro na tela. Dica: utilize os métodos `existe` e/ou `procurarIndice` aqui.

Crie uma classe chamada `TesteArrayClientes` com um método `main` para testar seu programa.

Parte 2

1. Para que a aplicação fique completa, os mesmos procedimentos realizados para a criação do repositório de clientes devem ser feitos para a criação do repositório de contas. Como os métodos a serem declarados neste repositório são muito similares aos métodos declarados no `RepositorioClientesArray`, para não ficar repetitivo, esta classe já está pronta e pode ser obtida na pasta de respostas do exercício.
   A diferença entre `RepositorioClientesArray` e `RepositorioContasArray` é que para a nova classe intitulada `RepositorioContasArray`, que obviamente lidará com contas e não com clientes, no lugar do cpf os métodos de procura/atualização/remoção desta classe utilizarão o <u>número</u> da conta a ser localizada/atualizada/removida.

Você pode criar uma outra classe de teste (`TesteArrayContas`) com um método `main`, desta vez para testar seu programa o seu programa com Contas.