Exercícios de Revisão - Parte 3
Unidade 1 - Linguagem C#

Resolva todos os exercícios de revisão dentro de uma mesma solução (*solution*) no Visual Studio e crie um projeto para cada um deles.

1. Reestruture o exercício 1 da lista 2 para que os dados dos clientes sejam lidos do arquivo **clientes.json** com o seguinte formato:

```
[ {"nome":"Pedro de Almeida", "cpf":"29585098008",
   "dt_nascimento":"15/02/1996", "renda_mensal":"3584,12", "estado_civil":"S",
   "dependentes":"0"},
  {"nome":"Ana Beatriz", "cpf":"34645846078",
   "dt_nascimento":"28/10/2000", "renda_mensal":"2810,25", "estado_civil":"S",
     "dependentes":"0"},
  {"nome":"Andre Rocha", "cpf":"69024641039",
   "dt_nascimento":"07/05/1984", "renda_mensal":"4110,39", "estado_civil":"S",
   "dependentes":"0"},
  {"nome":"Fernanda Pinheiro", "cpf":"21386403024",
   "dt_nascimento":"04/12/1972", "renda_mensal":"5850,27", "estado_civil":"C",
   "dependentes":"1"},
  {"nome":"Joice Medeiros", "cpf":"60904708055",
   "dt_nascimento":"21/07/1991", "renda_mensal":"10254,48", "estado_civil":"C",
   "dependentes":"2"},
  {"nome":"Carlos Roberto", "cpf":"96154144091",
   "dt_nascimento":"14/09/1966", "renda_mensal":"8387,92", "estado_civil":"C",
   "dependentes":"3"} ]
```

   Os dados dos clientes devem ser validados usando as mesmas regras do exercício 1 da lista 2. Caso haja um ou mais erros em um ou mais clientes, deve ser gerado um arquivo JSON chamado **erros.json** com o seguinte formato:

```
[ { "dados": { colocar aqui todos os dados do cliente com erro },
    "erros": [ {"nome do campo": "mensagem de erro"},
               {"nome do campo": "mensagem de erro"}, … ]
  },
  { "dados": { colocar aqui todos os dados do cliente com erro },
    "erros": [ {"nome do campo": "mensagem de erro"},
               {"nome do campo": "mensagem de erro"}, … ]
  },
  …
]
```

   Links sobre JSON: https://www.treinaweb.com.br/blog/o-que-e-json
   https://www.hostinger.com.br/tutoriais/o-que-e-json
   https://zetcode.com/csharp/json/
   https://code-maze.com/csharp-deserialize-complex-json-object/

2. Crie a classe **IndiceRemissivo** com os seguintes métodos

   - **IndiceRemissivo(string pathTXT, string pathIgnore)**: carrega as palavras do arquivo texto definido em **pathTXT** e as palavras do arquivo **pathIgnore**. O arquivo **pathIgnore** não é obrigatório (pode ser nulo), somente o arquivo **pathTXT**. Gere uma exceção caso haja algum problema na manipulação dos arquivos.
   - **Imprime()**: imprime, em ordem alfabética e em caixa alta, as palavras contidas no arquivo texto. Para cada palavra devem ser apresentados: a quantidade de vezes em que a palavra aparece no texto (entre parênteses) e os números das linhas em que ela aparece no texto (sem repetição e separados por vírgula) . Exemplo:

     JULGAMENTO (5) 35, 98, 548, 704

   Regras:

   1) Considere **palavra** qualquer sequência de um ou mais caracteres iniciados com uma letra (A a Z sem acentos) e separadas por espaço em branco ou um dos símbolos a seguir:

      . , ; < > : \ / | ~ ^ ´ ` [ ] { } ' " ! @ # $ % & * ( ) _ + =

   2) Devem ser desconsideradas as palavras contidas no arquivo **pathIgnore**.

   Sugestão: para os testes use os arquivos **texto.txt** e **ignore.txt** enviados em anexo.

3. Pesquise sobre **números de Armstrong** e implemente um **método de extensão** denominado **IsArmstrong** que retorna um valor booleano indicando se um número inteiro positivo é ou não um número de Armstrong. Por fim, usando esse método, imprima todos os números de Armstrong de 1 a 10000.

   Link sobre métodos de extensão: https://www.macoratti.net/18/04/c_extmet1.htm

4. Implemente um **método de extensão** denominado **RemoveRepetidos** que remove os valores repetidos em uma lista. A lista pode ser uma lista de inteiros, strings, caracteres ou qualquer outro tipo de objeto, por isso use **generics** para implementar o método de extensão. Verifique se o método funciona corretamente para:

   - Uma lista de inteiros.
   - Uma lista de strings.
   - Uma lista de Clientes com CPF e nome (dois clientes são iguais se possuem o mesmo CPF).

   Links sobre generics e método de extensão com generics:

   https://docs.microsoft.com/en-us/dotnet/csharp/fundamentals/types/generics

   https://www.codeproject.com/Articles/29079/Using-Generic-Extension-Methods