IPESU - INSTITUTO PERNAMBUCANO DE ENSINO SUPERIOR

CIÊNCIAS DA COMPUTAÇÃO

ATIVIDADES DE PRÁTICAS SUPERVISIONADAS

RECIFE

2019

JONATHA RIHAN R.A: 01600007668
RUAN VICTOR R.A: 01600007668
WILLIAN R.A: 01600007668
GABRIEL DE MORARES R.A: 01600007668

ATIVIDADES DE PRÁTICAS SUPERVISIONADAS

IPESU - INSTITUTO PERNAMBUCANO DE ENSINO SUPERIOR

Atividades Práticas Supervisionadas (APS)
apresentada como exigência para a avaliação do 2º semestre,
do(s) curso(s) de Ciências da Computação da Universidade Paulista,
sob orientação dos professores do semestre.

Orientador: Prof. Fábio Botelho

RECIFE

2019

Sumário

1. Introdução

1.1 Ambiente de pesquisa e produção

Este trabalho tem como objetivo documentar a prática de programação na linguagem C#, voltada à análise de dados em arquivos xml.

Como base de estudos, os arquivos usados serão:

- megasena.xml
- ceps.xml

Os mesmos foram disponibilizados pelo docente.

A plataforma de desenvolvimento será Microsft Visual Studio, para codificação de compilação dos scripts em C#, e o terminal do sistema operacional para execução.

O arquivo disponibilizado sofreu alterações devido a não interpretações dos caracteres especiais, a substituição fora feita no bloco de notas do Windows através da ferramenta de procura e substituição.

1.2 Proposta

Mega Sena:

- Dado um número de sorteio, retornar os números sorteados;

- Dado uma data de sorteio, retornar os números sorteados;

- Retornar a quantidade de vezes que cada número da mega-sena foi sorteado.

Base de CEP:

- A base de cep não pode ter erros de acentuação, corrijam por favor

- A questão 3 do projeto megasena tem que contabilizar os mostrar os números de 1 a 60

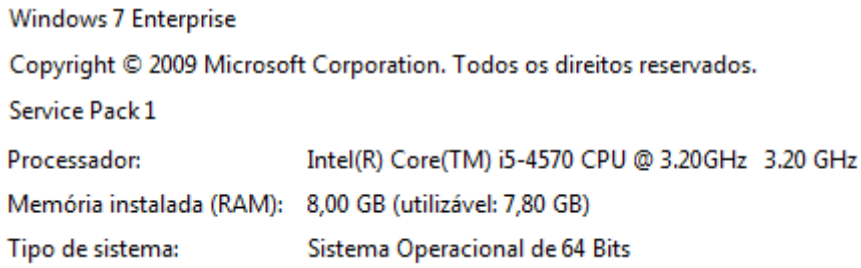
- Usem as questões 1 e 2 da megasena e adaptem para as consultas na base de cep

- Ponham um timer antes e depois da consulta a base de cep para mostrar o tempo que levou para fazer a consulta

- O código que executar de forma mais rápida a questão 3 do projeto megasena terá 1 ponto na np2.

1.3 Plataforma e Especificações

Figura 1 - Máquina



Fonte: Compilação do autor.

2. C#

O trabalho consiste na assimilação conteúdo visto em sala de aula, estruturas lineares, condicionais e de repetição na criação de um programa executável para pesquisa de dados nos arquivos .xml citados anteriormente.

C# é uma linguagem de programação, multiparadigma, de tipagem forte, desenvolvida pela Microsoft como parte da plataforma .NET. A sua sintaxe orientada a objetos foi baseada no C++ mas inclui muitas influências de outras linguagens de programação, como Object Pascal e, principalmente, Java.

Como característica do C#, temos a organização do programa pela estruturação em classes e métodos que serão usados na criação do projeto.

Figura 2 - Exemplo C#

```
using System;

namespace HelloWorld
{
    class Program
    {
        public static void Main(string[] args)
        {
            Console.WriteLine("Hello World");
        }
    }
}
```

Fonte: Compilação do autor.

3. XML

XML, do inglês eXtensible Markup Language, é uma linguagem de marcação recomendada pela W3C para a criação de documentos com dados organizados hierarquicamente, tais como textos, banco de dados ou desenhos vetoriais. A linguagem XML é classificada como extensível porque permite definir os elementos de marcação.

Linguagem de marcação é um agregado de códigos que podem ser aplicados a dados ou textos para serem lidos por computadores ou pessoas. Por exemplo, o HTML é uma linguagem de marcação para organizar e formatar um website, já o XML tem o mesmo conceito, mas para padronizar uma sequência de dados com o objetivo de organizar, separar o conteúdo e integrá-lo com outras linguagens.

Figura 3 - Exemplo XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<quiz>
    <record>
        <roll>10001</roll>
        <name>Roberto</name>
        <age>26</age>
    </record>
    <record>
        <roll>10002</roll>
        <name>Carlos</name>
        <age>13</age>
    </record>
    <record>
</quiz>
```

Fonte: Compilação do autor.

4. Software

O projeto iniciará com a importação das bibliotecas necessárias e a criação das classes e métodos para o menu da aplicação. O método Main será o menu que irá interligar todo o projeto para a interação do usuário, e classes secundarias para a realização da pesquisa propriamente dita.

4.1 Menu e Métodos

O método Main contém uma apresentação amigável para o usuário, contendo ASCII Arts, Sons e suas funções bem explicadas. As entradas de dados possuem tratamentos de exceções e estruturas condicionais para chamar o menus dos outros métodos:

Figura 5 – Método Main

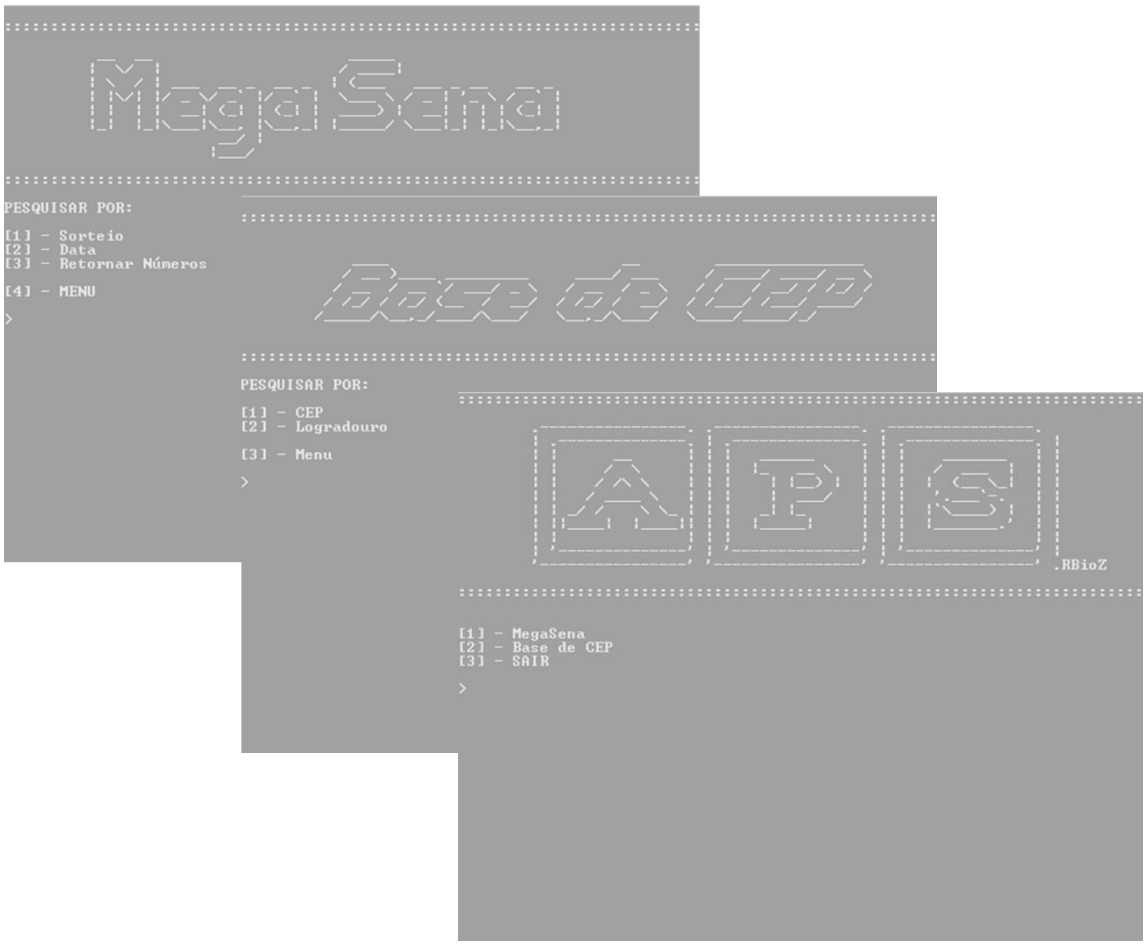
```
//Classe do projeto
class Program
{
    //Menu
    static void Main()
    {
        Console.Clear();
        int choice = 0;

        Console.Title = "Atividade Prática Supervisionada 2019";
        Console.SetWindowSize(75, 40);
        Console.Beep(440, 300);
        Console.Beep(520, 200);
        Console.Beep(900, 200);
        string title1 = @"
    .-----------------. .-----------------. .-----------------.
    | .---------------. | .---------------. | .---------------. |
    | |      __       | | |    _______    | | |    ________   | |
    | |     /  \      | | |   |_    __ \   | | |   /  ___   |  | |
    | |    / /\ \     | | |     | |__) |  | | |  |  (__ \_|  | |
    | |   / ____ \    | | |     |  ___/   | | |   '.___`-.   | |
    | | _/ /    \ \_  | | |    _| |_      | | |  |`\____) |  | |
    | ||____|  |____|| | |   |_____|     | | |  |_______.'  | |
    | |              | | |               | | |              | |
    | '--------------' | '---------------' | '--------------' |
    '-----------------' '-----------------' '-----------------' .RBioZ
    ";
        Console.ForegroundColor = ConsoleColor.Gray;
        Console.WriteLine(new string(':', 75));
        Console.WriteLine(title1);
        Console.WriteLine(new string(':', 75));
        Console.ForegroundColor = ConsoleColor.Gray;
        Console.WriteLine("\n\n[1] - MegaSena");
        Console.WriteLine("[2] - Base de CEP");
        Console.WriteLine("[3] - SAIR");
        Console.Write("\n> ");
        try
        {
        catch
        {

        if (choice == 1)
        {
            MegaSena_Menu();
        }
        else if (choice == 2)
        {
```

Fonte: Compilação do autor.

Figura 6 – Programa no Console



Fonte: Compilação do autor.

Os demais métodos são semelhantes, contanto, retornam chamadas de classes passando como parâmetro os dados descritos previamente, e retornam um array do tipo *string* com os dados.

Figura 7 – Instancia da função

```
if (option == 1)
{
    Console.Write("\nSorteio: ");
    Console.ForegroundColor = ConsoleColor.Green;
    search = Console.ReadLine();
    Console.ForegroundColor = ConsoleColor.Gray;

    var ms_return_1 = Mega.Pesquisa_Mega_1(search);

    Console.WriteLine(new string('_', 75));
    Console.Write("\nNumeros Sorteados: ");
    Console.ForegroundColor = ConsoleColor.Green;
    Console.WriteLine(ms_return_1[0]);
    Console.ForegroundColor = ConsoleColor.Gray;
    Console.Write("\nData: ");
    Console.ForegroundColor = ConsoleColor.Green;
    Console.WriteLine(ms_return_1[1]);
    Console.ForegroundColor = ConsoleColor.Gray;
    Console.WriteLine(new string('_', 75));
    Console.WriteLine("\n\nPress any button to continue...");
    Console.ReadKey();
}
```

Fonte: Compilação do autor.

4.2 Código

Com o menu estruturado, poderemos chamar as classes para realizar as pesquisas no arquivo. Iremos declarar uma variável do tipo *XmlTextReader* da classe *System.Xml*, e passar como parâmetro o caminho do arquivo a ser aberto, como padrão, deixaremos no mesmo diretório do executável. O xml funciona através de Nós, cada Nó possui seus marcadores e seus respectivos valores:

Figura 8 – Classes

```
public class CEP
{
    public static string[] Pesquisa_Cep_1(string search)
    {
        // Usando a classe XmlTextReader
        XmlTextReader ceps = new XmlTextReader(".\\ceps.xml");

        string cep = "", cidade = "", uf = "", bairro = "", logradouro = "", complemento = "";

        DateTime time_af = DateTime.Now;
        while (ceps.Read())
        {
            if (ceps.NodeType == XmlNodeType.Element && ceps.Name == "cep")
                cep = (ceps.ReadString());
            if (ceps.NodeType == XmlNodeType.Element && ceps.Name == "cidade")
                cidade = (ceps.ReadString());
            if (ceps.NodeType == XmlNodeType.Element && ceps.Name == "uf")
                uf = (ceps.ReadString());
            if (ceps.NodeType == XmlNodeType.Element && ceps.Name == "bairro")
                bairro = (ceps.ReadString());
            if (ceps.NodeType == XmlNodeType.Element && ceps.Name == "logradouro")
                logradouro = (ceps.ReadString());
            if (ceps.NodeType == XmlNodeType.Element && ceps.Name == "complemento")
            {
                complemento = (ceps.ReadString());

                if (search == cep)
                {
                    DateTime time_bf = DateTime.Now;
                    TimeSpan time_t = time_bf.Subtract(time_af);
                    ceps.Close();
                    string[] ret = new string[7] {cep,cidade, uf, bairro, logradouro, complemento, Convert.ToString(time_t)};
                    return ret;
                }
            }
        } // Fim While
        return new string[7];
    }
```

Fonte: Compilação do autor.

Antes do início do laço de repetição *While*, declaramos as variáveis para guardar os valores dos elementos do xml, e uma variável para capturar o tempo de execução, até encontrar o resultado compatível ou terminar de ler todos os nós do arquivo. Dentro do While, existem várias condicionais para averiguar os elementos do nó atual e o valor nele contido, caso o valor for compatível com o recebido do método, irá retornar dos dados em formato de uma matriz do tipo *String* e fechar o arquivo.

4.3 Fast Search

Uma das propostas para o desenvolvimento do software era a possibilidade de pesquisa rápida, a criação foi livre para o desenvolvimento do algoritmo de busca mais eficiente. A ideia a ser desenvolvida é a de carregar os dados na memória RAM para fácil acesso,

Figura 8 – Classes

```
public static void Fast_Search()
{
    // Usando a classe XmlTextReader
    XmlTextReader ceps = new XmlTextReader(".\\ceps.xml");

    //variáveis
    string cep = "", cidade = "", uf = "", bairro = "", logradouro = "", complemento = "";
    string search = "";

    //Variaveis de acesso rápido
    List<string> l_cep = new List<string>();
    List<string> l_cidade = new List<string>();
    List<string> l_uf = new List<string>();
    List<string> l_bairro = new List<string>();
    List<string> l_logradouro = new List<string>();
    List<string> l_complemento = new List<string>();

    string title4 = @"

    ______           __     _____                      __
   / ____/___ ______/ /_   / ___/___  ____ ___________/ /_
  / /_  / __ `/ ___/ __/   \__ \/ _ \/ __ `/ ___/ ___/ __ \
 / __/ / /_/ (__  ) /_    ___/ /  __/ /_/ / /  / /__/ / / /
/_/    \__,_/____/\__/   /____/\___/\__,_/_/   \___/_/ /_/


    ";


    Console.WriteLine("\nCarregando arquivos na memória...");

    //Carregar arquivos na memoria
    while (ceps.Read())
    {
        if (ceps.NodeType == XmlNodeType.Element && ceps.Name == "cep")
            cep = (ceps.ReadString());
            l_cep.Add(cep);
        if (ceps.NodeType == XmlNodeType.Element && ceps.Name == "cidade")
            cidade = (ceps.ReadString());
            l_cidade.Add(cidade);
        if (ceps.NodeType == XmlNodeType.Element && ceps.Name == "uf")
            uf = (ceps.ReadString());
            l_uf.Add(uf);
        if (ceps.NodeType == XmlNodeType.Element && ceps.Name == "bairro")
            bairro = (ceps.ReadString());
            l_bairro.Add(bairro);
        if (ceps.NodeType == XmlNodeType.Element && ceps.Name == "logradouro")
            logradouro = (ceps.ReadString());
            l_logradouro.Add(logradouro);
        if (ceps.NodeType == XmlNodeType.Element && ceps.Name == "complemento")
            complemento = (ceps.ReadString());
            l_complemento.Add(complemento);

    } // Fim While
```

Foram declaradas variáveis do tipo *List<string>* para armazenar os valores a cada nó.

5. Referências

Ana Paula Pereira. Tecmundo, 2009. Disponivel em:
https://www.tecmundo.com.br/programacao/1762-o-que-e-xml-.htm

Wikipédia, 2019. Disponivel em:
https://www.tecmundo.com.br/programacao/1762-o-que-e-xml-.htm