```
// Importação de bibliotecas necessárias
using System;
using System.Collections.Generic;
using System.Drawing;
using System.Linq;
using System.Windows.Forms;
using System.Windows.Forms.DataVisualization.Charting;

namespace GraficoConsumoComodo
{
    public partial class Form1 : Form
    {
        private ComboBox comodoComboBox;
        private Dictionary<string, double[]> dadosUsuarioAtual;
        private Dictionary<string, double[]> dadosUsuariosFicticios;
        // Construtor do formulário
        public Form1()
        {
            InitializeComponent();
            InicializarCustomComponents(); // Inicializa componentes personalizados
        }

        // Método que configura o gráfico e a listagem de ranking
        private void InicializarCustomComponents()
        {
            chart = new Chart
            {
                Dock = DockStyle.Left,
                Width = this.ClientSize.Width * 2 / 3,
                BackColor = Color.Black
            };

            var chartArea = new ChartArea();
            chartArea.AxisX.Title = "Mês";
            chartArea.AxisY.Title = "Consumo (kWh)";
            chartArea.AxisX.LabelStyle.ForeColor = Color.White;
            chartArea.AxisY.LabelStyle.ForeColor = Color.White;
            chartArea.BackColor = Color.Black;
            chart.ChartAreas.Add(chartArea);
            this.Controls.Add(chart);

            rankingListBox = new ListBox
            {
                Dock = DockStyle.Right,
```

```
            Width = this.ClientSize.Width / 3,
            Font = new Font("Segoe UI", 12),
            BackColor = Color.Black,
            ForeColor = Color.White
        };
        this.Controls.Add(rankingListBox);

        comodoComboBox = new ComboBox
        {
            Dock = DockStyle.Top,
            DropDownStyle = ComboBoxStyle.DropDownList,
            Font = new Font("Segoe UI", 12)
        };
        comodoComboBox.SelectedIndexChanged +=
ComodoComboBox_SelectedIndexChanged;
        this.Controls.Add(comodoComboBox);

        dadosUsuarioAtual = ObterDadosMensaisUsuarioAtual();
        dadosUsuariosFicticios = ObterDadosUsuariosFicticios();

        foreach (var comodo in dadosUsuarioAtual.Keys)
        {
            comodoComboBox.Items.Add(comodo);
        }

        comodoComboBox.SelectedIndex = 0;
    }

    // Método que carrega os dados e adiciona ao gráfico
    private void CarregarDados()
    {
        var dadosUsuarioAtual = ObterDadosMensaisUsuarioAtual();
        var dadosUsuariosFicticios = ObterDadosUsuariosFicticios();

        foreach (var comodo in dadosUsuarioAtual.Keys)
        {
            // Cria uma nova série no gráfico para cada cômodo
            var serie = new Series(comodo)
            {
                ChartType = SeriesChartType.Line,
                BorderWidth = 2,
                Color = ObterCorPorComodo(comodo),
                MarkerStyle = MarkerStyle.Circle,
                MarkerSize = 6
```

```
            };

            var consumo = dadosUsuarioAtual[comodo];

            // Adiciona pontos à série
            for (int i = 0; i < consumo.Length; i++)
            {
                serie.Points.AddXY(ObterNomeMes(i), consumo[i]);
            }

            chart.Series.Add(serie);
        }

        // Atualiza o ranking com os dados carregados
        AtualizarRanking(dadosUsuarioAtual, dadosUsuariosFicticios);
    }

    // Método que calcula e exibe o ranking de consumo total
    private void AtualizarRanking(Dictionary<string, double[]> usuarioAtual,
                                  Dictionary<string, double[]> usuariosFicticios)
    {
        var ranking = new Dictionary<string, double>
        {
            { "Você", usuarioAtual.Values.SelectMany(x => x).Sum() }
        };

        foreach (var entry in usuariosFicticios)
        {
            ranking[entry.Key] = entry.Value.Sum();
        }

        var ordenado = ranking.OrderByDescending(x => x.Value).ToList();

        rankingListBox.Items.Clear();
        rankingListBox.Items.Add("Ranking de Consumo Total (Mensal):");

        int posicao = 1;
        foreach (var entry in ordenado)
        {
            rankingListBox.Items.Add($"{posicao}. {entry.Key} - {entry.Value:F1}
kWh");
            posicao++;
        }
    }
```

```
        // Dados fictícios de consumo do usuário atual
        private Dictionary<string, double[]> ObterDadosMensaisUsuarioAtual()
        {
            return new Dictionary<string, double[]>
            {
                { "Sala de Estar", new double[] { 10, 15, 25, 20, 32, 33, 34, 41, 30, 51, 50,
52 } },
                { "Quarto", new double[] { 8, 12, 20, 18, 24, 26, 28, 35, 25, 40, 39, 41 } },
                { "Cozinha", new double[] { 12, 18, 30, 28, 35, 38, 40, 45, 32, 55, 53, 56 } }
            };
        }

        // Dados fictícios de outros usuários para ranking
        private Dictionary<string, double[]> ObterDadosUsuariosFicticios()
        {
            return new Dictionary<string, double[]>
            {
                { "Usuário A", new double[] { 310, 315, 280, 290, 275, 300, 295, 310, 330,
310, 315, 320 } },
                { "Usuário B", new double[] { 200, 210, 190, 180, 220, 230, 240, 250, 230,
210, 220, 225 } },
                { "Usuário C", new double[] { 450, 460, 470, 440, 430, 420, 460, 470, 480,
460, 455, 465 } },
                { "Usuário D", new double[] { 150, 160, 170, 165, 175, 180, 190, 185, 190,
200, 195, 205 } }
            };
        }

        // Define cores diferentes para cada cômodo
        private Color ObterCorPorComodo(string comodo)
        {
            return comodo switch
            {
                "Sala de Estar" => Color.LimeGreen,
                "Quarto" => Color.Orange,
                "Cozinha" => Color.Cyan,
                _ => Color.White
            };
        }

        // Retorna o nome do mês com base no índice
        private string ObterNomeMes(int indice)
        {
```

```
            string[] meses =
            {
                "Jan", "Feb", "Mar", "Apr", "May", "Jun",
                "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"
            };
            return meses[indice];
        }

        // Evento de carregamento do formulário (vazio no momento)
        private void Form1_Load(object sender, EventArgs e)
        {
        }
        private void ComodoComboBox_SelectedIndexChanged(object sender,
EventArgs e)
        {
            string comodoSelecionado = comodoComboBox.SelectedItem.ToString();
            AtualizarGraficoPorComodo(comodoSelecionado);
        }

        private void AtualizarGraficoPorComodo(string comodo)
        {
            chart.Series.Clear();

            var serieUsuario = new Series("Você")
            {
                ChartType = SeriesChartType.Line,
                BorderWidth = 2,
                Color = ObterCorPorComodo(comodo),
                MarkerStyle = MarkerStyle.Circle,
                MarkerSize = 6
            };

            var consumoUsuario = dadosUsuarioAtual[comodo];
            for (int i = 0; i < consumoUsuario.Length; i++)
            {
                serieUsuario.Points.AddXY(ObterNomeMes(i), consumoUsuario[i]);
            }

            chart.Series.Add(serieUsuario);

            foreach (var entry in dadosUsuariosFicticios)
            {
                var serie = new Series(entry.Key)
                {
```

```
                ChartType = SeriesChartType.Line,
                BorderDashStyle = ChartDashStyle.Dash,
                BorderWidth = 2,
                MarkerStyle = MarkerStyle.Square,
                MarkerSize = 5,
                Color = Color.Gray
            };

            for (int i = 0; i < entry.Value.Length; i++)
            {
                serie.Points.AddXY(ObterNomeMes(i), entry.Value[i]);
            }

            chart.Series.Add(serie);
        }

        AtualizarRankingPorComodo(comodo);
    }

    private void AtualizarRankingPorComodo(string comodo)
    {
        var ranking = new Dictionary<string, double>
  {
    { "Você", dadosUsuarioAtual[comodo].Sum() }
  };

        foreach (var entry in dadosUsuariosFicticios)
        {
            ranking[entry.Key] = entry.Value.Sum();
        }

        var ordenado = ranking.OrderBy(x => x.Value).ToList();

        rankingListBox.Items.Clear();
        rankingListBox.Items.Add($"Ranking por Consumo - {comodo}:");

        int posicao = 1;
        foreach (var entry in ordenado)
        {
            string marcador = entry.Key == "Você" ? " ← Você (linha colorida)" : "";
            rankingListBox.Items.Add($"{posicao}. {entry.Key} - {entry.Value:F1}
kWh{marcador}");
            posicao++;
        }
```

```
        }

    }

}
```

message.txt10 KB