Wydział Elektroniki
i Technik Informacyjnych
POLITECHNIKA WARSZAWSKA

*Praca inżynierska*

# Edytor multigrafów

*Jan Kędra*

Promotor*: mgr inż. Waldemar Grabski*

Politechnika
Warszawska

# Cel pracy



Praca miała na celu zaprojektowanie i stworzenie aplikacji która pozwala na tworzenie i edycję multigrafów.

Wykonana aplikacja miała z założenia wspierać tworzenie i edycję multigrafów, co może być wykorzystywane przy projektowaniu sieci kolejowych.

# Wymagania



- Operacje tworzenia i edytowania multigrafu
- Operacje zapisu i otwierania stworzonych struktur
- Łatwość rozszerzania systemu
- Prosty w obsłudze interfejs użytkownika

# Zaimplementowane funkcjonalności

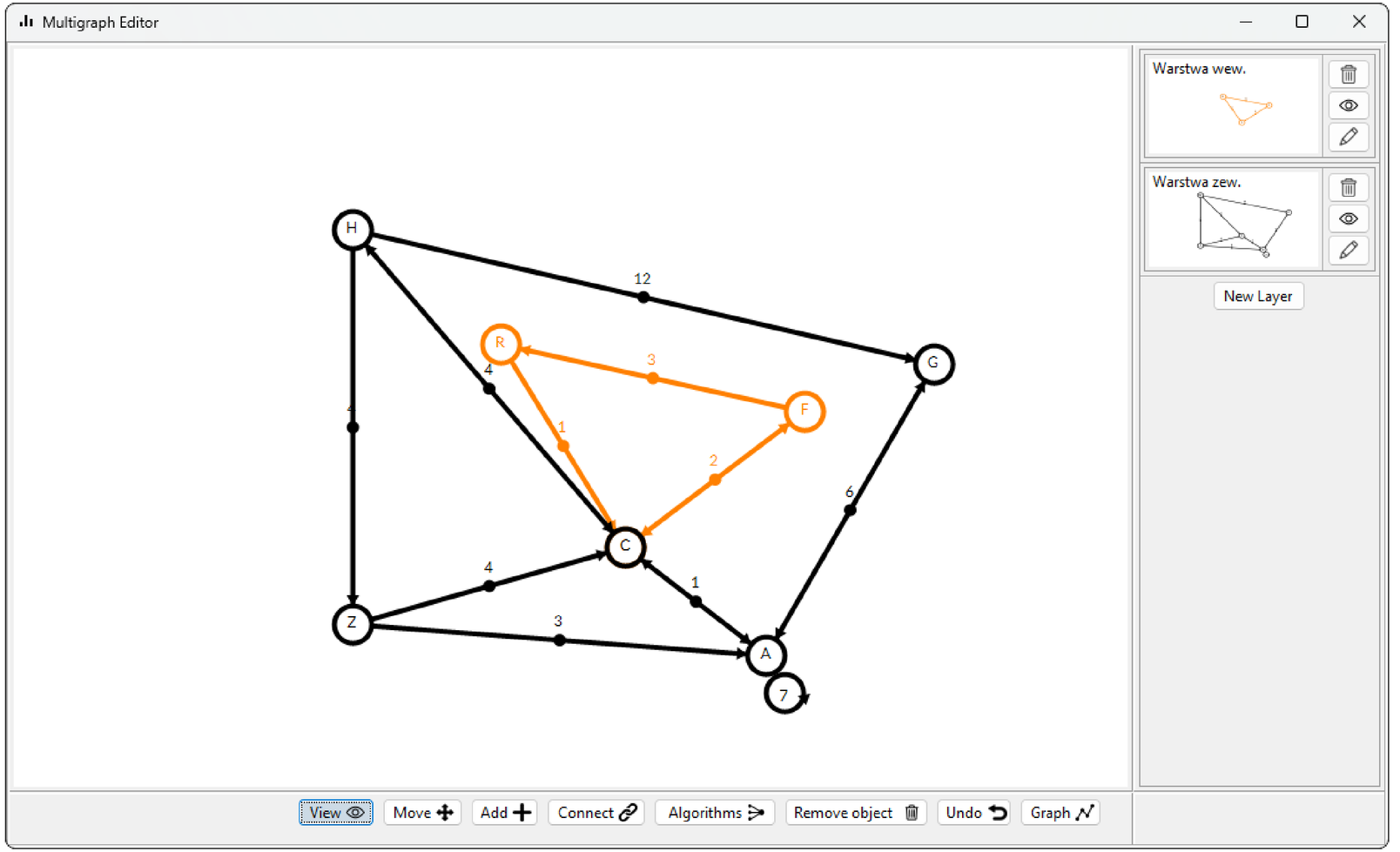


- Tworzenie i edycja multigrafów
- Zapisywanie, otwieranie i eksport stworzonych multigrafów
- System warstw wspierający stworzenie skomplikowanych struktur
- Zestaw algorytmów pozwalających na analizę multigrafów

# Wykorzystane technologie



- C# 12.0 z użyciem .NET 8.0
- Windows Forms
- MSTest
- TestStack.White




# Prezentacja aplikacji (1/3)

Program.cs

```
namespace MultigraphEditor
{
    internal static class Program
    {
        [STAThread]
        static void Main()
        {
            Type NodeType = typeof(MGraphEditorNode);
            Type EdgeType = typeof(MGraphEditorEdge);
            Type LayerType = typeof(MGraphLayer);
            List<Type> AlgorithmList = new List<Type>();
            AlgorithmList.Add(typeof(DijkstraAlgorithm));
            AlgorithmList.Add(typeof(HamiltonCycleAlgorithm));

            ApplicationConfiguration.Initialize();
            Application.Run(new MainForm(NodeType, EdgeType, LayerType,
AlgorithmList));
        }
    }
}
```

Prezentajca aplikacji (2/3)

MGraphEditorNodeSquare.cs

```
namespace MultigraphEditor.src.graph.example
{
    [Serializable]
    public class MGraphEditorNodeSquare : MGraphEditorNode
    {
        public override void Draw(Graphics g, INodeLayer l)
        {
            g.SmoothingMode = SmoothingMode.AntiAlias;

            Pen p = new Pen(l.Color, l.nodeWidth);
            using (p)
            {
                float x = GetDrawingCoordinates().Item1;
                float y = GetDrawingCoordinates().Item2;

                g.DrawRectangle(p, x, y, Diameter, Diameter);
                if (Label != null)
                {
                    DrawLabel(g, l);
                }
            }
        }
    }
}
```

# Prezentacja aplikacji (3/3)