# Dokumentacja Sprawdzianu Nr 3 Antoni Nowalski 57295

Kod metod obsługi kontrolek:

double anmaksimum(double[,] anTWS)

{

double anmax = anTWS[0, 1];

for (int i = 0; i < anTWS.GetLength(0); i++)

{

if (anTWS[i,1] > anmax)

{

anmax = anTWS[i,1];

}

}

return anmax;

}

double anminimum(double[,] anTWS)

{

double anmin = anTWS[0, 1];

for (int i = 0; i < anTWS.GetLength(0); i++)

{

if (anTWS[i, 1] < anmin)

{

anmin = anTWS[i, 1];

}

}

return anmin;

}

bool anpobieranieX(out double anX)

{

errorProvider1.Dispose();

anX = 0;

if (txtx.Text.Trim()=="")

{

errorProvider1.SetError(txtx, "Brak wartości zmiennej X");

return false;

}

else if (!double.TryParse(txtx.Text, out anX))

{

errorProvider1.SetError(txtx, "Zła wartość zmiennej X");

return false;

}

return true;

}

bool anpobieranieEps(out double aneps)

{

errorProvider1.Dispose();

aneps = 0;

if (txtDokladnoscEps.Text.Trim() == "")

{

errorProvider1.SetError(txtDokladnoscEps, "Brak wartości zmiennej Eps");

return false;

}

else if (!double.TryParse(txtDokladnoscEps.Text, out aneps) || aneps <= 0 || aneps >= 1 )

{

errorProvider1.SetError(txtDokladnoscEps, "Zła wartość zmiennej Eps ");

return false;

}

return true;

}

bool anpobieranieEpsCalki(out double aneps)

{

errorProvider1.Dispose();

aneps = 0;

if (txtDokladnoscEps.Text.Trim() == "")

{

errorProvider1.SetError(txtDokladnoscObliczen, "Brak wartości zmiennej Eps");

return false;

}

else if (!double.TryParse(txtDokladnoscObliczen.Text, out aneps) || aneps <= 0 || aneps > 0.05)

{

errorProvider1.SetError(txtDokladnoscObliczen, "Zła wartość zmiennej Eps ");

return false;

}

return true;

}

bool anpobieranieKroku(out double ank)

{

errorProvider1.Dispose();

ank = 0;

if (txtKrok.Text.Trim() == "")

{

errorProvider1.SetError(txtKrok, "Brak wartości zmiennej Skok");

return false;

}

else if (!double.TryParse(txtKrok.Text, out ank) || ank < 0 || ank > 1)

{

errorProvider1.SetError(txtKrok, "Zła wartość zmiennej Skok ");

return false;

}

return true;

}

bool anpobieranieXdXg(out double anXd, out double anXg)

{

errorProvider1.Dispose();

anXd = anXg = 1;

if (txtXd.Text.Trim() == "")

{

errorProvider1.SetError(txtXd, "Brak wartości zmiennej Xd");

return false;

}

else if (txtXg.Text.Trim() == "")

{

errorProvider1.SetError(txtXg, "Brak wartości zmiennej Xg");

return false;

}

else if (!double.TryParse(txtXd.Text, out anXd))

{

errorProvider1.SetError(txtXd, "Zła wartość zmiennej Xd ");

return false;

}

else if (!double.TryParse(txtXg.Text, out anXg))

{

errorProvider1.SetError(txtXg, "Zła wartość zmiennej Xg ");

return false;

}

else if (anXd > anXg)

{

errorProvider1.SetError(txtXg, "Xd nie może być większa od Xg");

return false;

}

return true;

}

bool anpobieranieXdXgCalki(out double anXd, out double anXg)

{

errorProvider1.Dispose();

anXd = anXg = 1;

if (txtDolnaGranicaCalkowita.Text.Trim() == "")

{

errorProvider1.SetError(txtDolnaGranicaCalkowita, "Brak wartości zmiennej Xd");

return false;

}

else if (txtGornaGranicaCalkowita.Text.Trim() == "")

{

errorProvider1.SetError(txtGornaGranicaCalkowita, "Brak wartości zmiennej Xg");

return false;

}

else if (!double.TryParse(txtDolnaGranicaCalkowita.Text, out anXd))

{

errorProvider1.SetError(txtDolnaGranicaCalkowita, "Zła wartość zmiennej Xd ");

return false;

}

else if (!double.TryParse(txtGornaGranicaCalkowita.Text, out anXg))

{

errorProvider1.SetError(txtGornaGranicaCalkowita, "Zła wartość zmiennej Xg ");

return false;

}

else if (anXd > anXg)

{

errorProvider1.SetError(txtGornaGranicaCalkowita, "Xd nie może być większa od Xg");

return false;

}

return true;

}

double anSumaSzereguPotegowego(double anx, double aneps, out ushort anlicznik)

{

double ansuma = 1, w = 1;

anlicznik = 1;

while (Math.Abs(w) > aneps)

{

anlicznik++;

w \*= (-anx / anlicznik);

ansuma += w;

}

return ansuma;

}

private void anbtnObliczSumeSzereguPotegowego\_Click(object sender, EventArgs e)

{

ushort anlicznik = 0;

double anX, aneps;

if (!anpobieranieX(out anX))

return;

if (!anpobieranieEps(out aneps))

return;

txtObliczonaSumaSz.Text = string.Format("{0:F4}", anSumaSzereguPotegowego(anX, aneps, out anlicznik));

txtx.Enabled = false;

anbtnObliczSumeSzereguPotegowego.Enabled = false;

}

private void exitToolStripMenuItem\_Click(object sender, EventArgs e)

{

this.Close();

}

void anTablicowanieSzereguPotegowego(double anXd, double anXg, double aneps, double ank, out double[,] anTWS)

{

int ann = (int)(Math.Abs(anXg - anXd) / ank);

anTWS = new double[ann+1, 3];

double anx;

ushort anlicznik;

int ani;

for (ani = 0, anx = anXd; ani < anTWS.GetLength(0); ani++, anx = anXd + ani \* ank)//lub x+=k

{

anTWS[ani, 0] = anx;

anTWS[ani, 1] = anSumaSzereguPotegowego(anx, aneps, out anlicznik);

anTWS[ani, 2] = anlicznik;

}

}

private void btnTabelarycznaWizualizacja\_Click(object sender, EventArgs e)

{

double aneps, anxd, anxg, ank;

if (!anpobieranieEps(out aneps))

{

return;

}

if (!anpobieranieKroku(out ank))

{

return;

}

if (!anpobieranieXdXg(out anxd, out anxg))

{

return;

}

txtDokladnoscEps.Enabled = false;

txtXd.Enabled = false;

txtXg.Enabled = false;

txtKrok.Enabled = false;

double[,] anTWS;

anTablicowanieSzereguPotegowego(anxd, anxg, aneps, ank, out anTWS);

dgvTabela.Visible = true;

pbZdjecie.Visible = false;

chtGraficznaWizualizacja.Visible = false;

dgvTabela.Rows.Clear();

dgvTabela.Columns[0].DefaultCellStyle.Alignment = DataGridViewContentAlignment.MiddleCenter;

dgvTabela.Columns[1].DefaultCellStyle.Alignment = DataGridViewContentAlignment.MiddleCenter;

dgvTabela.Columns[2].DefaultCellStyle.Alignment = DataGridViewContentAlignment.MiddleCenter;

for (int i = 0; i < anTWS.GetLength(0); i++)

{

dgvTabela.Rows.Add();

dgvTabela.Rows[i].Cells[0].Value = string.Format("{0:F2}", anTWS[i, 0]);

dgvTabela.Rows[i].Cells[1].Value = string.Format("{0:F3}", anTWS[i, 1]);

dgvTabela.Rows[i].Cells[2].Value = string.Format("{0}", anTWS[i, 2]);

}

btnTabelarycznaWizualizacja.Enabled = false;

}

void TablicowanieWartosciSzeregu(ref double[,] anTWS, double anxd, double anxg, double ank, double aneps)

{

double anx;

int ani;

ushort anlicznik;

for (ani = 0, anx = anxd; ani < anTWS.GetLength(0); ani++, anx += ank)

{

anTWS[ani, 0] = anx;

anTWS[ani, 1] = anSumaSzereguPotegowego(anx, aneps, out anlicznik);

}

}

private void btnGraficznaWizualizacja\_Click(object sender, EventArgs e)

{

//odsłonięcie kontrolek

btnWybierzkolorlinii.Visible = true;

lblWziernikKoloruLinii.Visible = true;

lblWziernikKoloruLinii.Visible = true;

btnWybierzKolorTla.Visible = true;

lblWziernikKoloruTla.Visible = true;

txtWziernikKoloruTla.Visible = true;

lblStylLiniiWykresu.Visible = true;

comboxStylLiniiWykresu.Visible = true;

lblGruboscLiniiWykresu.Visible = true;

tbarGruboscLiniWykresu.Visible = true;

lblGruboscLiniWykresu.Visible = true;

txtGruboscLiniiWykr.Visible = true;

lblWykreslanieOsiUkladu.Visible = true;

rbtnLinieOsiUkladyBezOpisu.Visible = true;

rbtnLinieUkladuWspolrzednychZOpisem.Visible = true;

rbtnGrotyStrzalek.Visible = true;

rbtnGrotyStrzalek.Visible = true;

tsddbKoloryWykresu.Enabled = true;

tsddbStylLinii.Enabled = true;

tsddbWyborGrubosciLiniiMenu.Enabled = true;

tsddbTypWykresu.Enabled = true;

tsddbStylCzcionki.Enabled = true;

txtWziernikKoloruLinii.Visible = true;

double aneps, anxd, anxg, ank;

if (!anpobieranieEps(out aneps))

{

return;

}

if (!anpobieranieKroku(out ank))

{

return;

}

if (!anpobieranieXdXg(out anxd, out anxg))

{

return;

}

txtDokladnoscEps.Enabled = false;

txtXd.Enabled = false;

txtXg.Enabled = false;

txtKrok.Enabled = false;

int ann = (int)((anxg - anxd) / ank);

double[,] TWS = new double[ann + 1, 2];

TablicowanieWartosciSzeregu(ref TWS, anxd, anxg, ank, aneps);

chtGraficznaWizualizacja.Visible = true;

dgvTabela.Visible = false;

pbZdjecie.Visible = false;

//Tytuł

//chtGraficznaWizualizacja.Titles.Clear();

//chtGraficznaWizualizacja.Titles.Add("Wykres zmian wartości szeregu");

//chtGraficznaWizualizacja.Font = new Font("Tahoma", 14, FontStyle.Bold);

//chtGraficznaWizualizacja.Series[0].Font = new Font("Arial", 10, FontStyle.Bold);

chtGraficznaWizualizacja.Legends.FindByName("Legend1").Docking = Docking.Bottom;//umieszczenie legendy

chtGraficznaWizualizacja.BackColor = Color.GreenYellow;

//marginesy

chtGraficznaWizualizacja.ChartAreas[0].Position.X = 10;

chtGraficznaWizualizacja.ChartAreas[0].Position.Y = 10;

chtGraficznaWizualizacja.ChartAreas[0].Position.Width = 80;

chtGraficznaWizualizacja.ChartAreas[0].Position.Height = 80;

//opis osi X i Y

chtGraficznaWizualizacja.ChartAreas[0].AxisX.Title = "Wartość zmiennej x";

chtGraficznaWizualizacja.ChartAreas[0].AxisY.Title = "Wartość szeregu S(x)";

//chtGraficznaWizualizacja.ChartAreas[0].AxisX.LabelStyle.Font = new Font("Arial", 14, FontStyle.Bold);

//chtGraficznaWizualizacja.ChartAreas[0].AxisY.LabelStyle.Font = new Font("Arial", 14, FontStyle.Bold);

//chtGraficznaWizualizacja.ChartAreas[0].AxisY.TitleFont = new Font("Arial", 10, FontStyle.Bold);

//sformatowanie osi X i Y

chtGraficznaWizualizacja.ChartAreas[0].AxisX.LabelStyle.Format = "{0:F2}";

chtGraficznaWizualizacja.ChartAreas[0].AxisY.LabelStyle.Format = "{0:F2}";

//chtGraficznaWizualizacja.ChartAreas[0].AxisY.TitleFont = fontDialog1.Font;

//chtGraficznaWizualizacja.ChartAreas[0].AxisX.TitleFont = fontDialog1.Font;

//wyzerowanie danych

chtGraficznaWizualizacja.Series.Clear();

//dodanie nowej serii 0

chtGraficznaWizualizacja.Series.Add("Seria 0");

//ustawienie nazw dla dodanej serii danych

chtGraficznaWizualizacja.Series[0].Name = "Wartość szeregu potęgowego";

//ustawienie typu wykresu

chtGraficznaWizualizacja.Series[0].ChartType = SeriesChartType.Line;

//ustawienie koloru linii wykresu

chtGraficznaWizualizacja.Series[0].Color = Color.Black;

//ustwienie stylu linii wykresu

chtGraficznaWizualizacja.Series[0].BorderDashStyle = ChartDashStyle.Solid;

//ustawienie grubości linii wykresu

chtGraficznaWizualizacja.Series[0].BorderWidth = int.Parse(txtGruboscLiniiWykr.Text);

//ustawienia osi X i Y

//chtGraficznaWizualizacja.ChartAreas[0].AxisX.Minimum = xd;

//chtGraficznaWizualizacja.ChartAreas[0].AxisX.Maximum = xg;

//eksperyment

//chtGraficznaWizualizacja.ChartAreas[0].AxisY.Minimum = minimum(TWS)-1;

//chtGraficznaWizualizacja.ChartAreas[0].AxisY.Maximum = maksimum(TWS)+1;

comboxStylLiniiWykresu.SelectedIndex = 3;

for (int i = 0; i < TWS.GetLength(0); i++)

chtGraficznaWizualizacja.Series[0].Points.AddXY(TWS[i, 0], TWS[i, 1]);

btnGraficznaWizualizacja.Enabled = false;

}

private void btnWybierzKolorTla\_Click(object sender, EventArgs e)

{

if (KolorWybor.ShowDialog() == DialogResult.OK)

{

chtGraficznaWizualizacja.BackColor = KolorWybor.Color;

txtWziernikKoloruTla.BackColor = KolorWybor.Color;

}

}

private void btnWybierzkolorlinii\_Click(object sender, EventArgs e)

{

if (KolorWybor.ShowDialog() == DialogResult.OK)

{

chtGraficznaWizualizacja.Series[0].Color = KolorWybor.Color;

txtWziernikKoloruLinii.BackColor = KolorWybor.Color;

}

}

private void btnResetuj\_Click(object sender, EventArgs e)

{

anbtnObliczSumeSzereguPotegowego.Enabled = true;

txtx.Enabled = true;

txtDokladnoscEps.Enabled = true;

txtXd.Enabled = true;

txtXg.Enabled = true;

txtKrok.Enabled = true;

chtGraficznaWizualizacja.Visible = false;

dgvTabela.Visible = false;

pbZdjecie.Visible = true;

btnGraficznaWizualizacja.Enabled = true;

btnTabelarycznaWizualizacja.Enabled = true;

txtx.Text = "";

txtDokladnoscEps.Text = "";

txtXd.Text = "";

txtXg.Text = "";

txtKrok.Text = "";

txtGruboscLiniiWykr.Text = "2";

txtObliczonaSumaSz.Text = "";

tbarGruboscLiniWykresu.Value = 2;

txtWziernikKoloruTla.BackColor = Color.GreenYellow;

txtWziernikKoloruLinii.BackColor = Color.Black;

txtWziernikKoloruLinii.Visible = false;

btnWybierzkolorlinii.Visible = false ;

lblWziernikKoloruLinii.Visible = false;

lblWziernikKoloruLinii.Visible = false;

btnWybierzKolorTla.Visible = false;

lblWziernikKoloruTla.Visible = false;

txtWziernikKoloruTla.Visible = false;

lblStylLiniiWykresu.Visible = false;

comboxStylLiniiWykresu.Visible = false;

lblGruboscLiniiWykresu.Visible = false;

tbarGruboscLiniWykresu.Visible = false;

lblGruboscLiniWykresu.Visible = false;

txtGruboscLiniiWykr.Visible = false;

lblWykreslanieOsiUkladu.Visible = false;

//radiobuttons

rbtnLinieOsiUkladyBezOpisu.Visible = false;

rbtnLinieUkladuWspolrzednychZOpisem.Visible = false;

rbtnGrotyStrzalek.Visible = false;

rbtnLinieOsiUkladyBezOpisu.Checked = false;

rbtnLinieUkladuWspolrzednychZOpisem.Checked = true;

rbtnGrotyStrzalek.Checked = false;

//menu

tsddbKoloryWykresu.Enabled = false;

tsddbStylLinii.Enabled = false;

tsddbWyborGrubosciLiniiMenu.Enabled = false;

tsddbTypWykresu.Enabled = false;

tsddbStylCzcionki.Enabled = false;

//czcionka opisów osi wykresu

}

private void tssbGruboscLinii1\_Click(object sender, EventArgs e)

{

chtGraficznaWizualizacja.Series[0].BorderWidth = 1;

txtGruboscLiniiWykr.Text = "1";

tbarGruboscLiniWykresu.Value = 1;

}

private void tssbGruboscLinii2\_Click(object sender, EventArgs e)

{

chtGraficznaWizualizacja.Series[0].BorderWidth = 2;

txtGruboscLiniiWykr.Text = "2";

tbarGruboscLiniWykresu.Value = 2;

}

private void tssbGruboscLinii3\_Click(object sender, EventArgs e)

{

chtGraficznaWizualizacja.Series[0].BorderWidth = 3;

txtGruboscLiniiWykr.Text = "3";

tbarGruboscLiniWykresu.Value = 3;

}

private void tssbGruboscLinii4\_Click(object sender, EventArgs e)

{

chtGraficznaWizualizacja.Series[0].BorderWidth = 4;

txtGruboscLiniiWykr.Text = "4";

tbarGruboscLiniWykresu.Value = 4;

}

private void tssbGruboscLinii5\_Click(object sender, EventArgs e)

{

chtGraficznaWizualizacja.Series[0].BorderWidth = 5;

txtGruboscLiniiWykr.Text = "5";

tbarGruboscLiniWykresu.Value = 5;

}

private void tbarGruboscLiniWykresu\_Scroll(object sender, EventArgs e)

{

chtGraficznaWizualizacja.Series[0].BorderWidth = tbarGruboscLiniWykresu.Value;

txtGruboscLiniiWykr.Text = tbarGruboscLiniWykresu.Value.ToString();

}

private void kolorTłaWykresuToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (KolorWybor.ShowDialog() == DialogResult.OK)

{

chtGraficznaWizualizacja.BackColor = KolorWybor.Color;

txtWziernikKoloruTla.BackColor = KolorWybor.Color;

}

}

private void kolorLiniiWykresuToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (KolorWybor.ShowDialog() == DialogResult.OK)

{

chtGraficznaWizualizacja.Series[0].Color = KolorWybor.Color;

txtWziernikKoloruLinii.BackColor = KolorWybor.Color;

}

}

private void rbtnLinieOsiUkladyBezOpisu\_CheckedChanged(object sender, EventArgs e)

{

chtGraficznaWizualizacja.ChartAreas[0].AxisX.Title = "";

chtGraficznaWizualizacja.ChartAreas[0].AxisY.Title = "";

chtGraficznaWizualizacja.ChartAreas[0].AxisX.ArrowStyle = AxisArrowStyle.None;

chtGraficznaWizualizacja.ChartAreas[0].AxisY.ArrowStyle = AxisArrowStyle.None;

}

private void rbtnLinieUkladuWspolrzednychZOpisem\_CheckedChanged(object sender, EventArgs e)

{

chtGraficznaWizualizacja.ChartAreas[0].AxisX.Title = "Wartość zmiennej x";

chtGraficznaWizualizacja.ChartAreas[0].AxisY.Title = "Wartość szeregu S(x)";

chtGraficznaWizualizacja.ChartAreas[0].AxisX.ArrowStyle = AxisArrowStyle.None;

chtGraficznaWizualizacja.ChartAreas[0].AxisY.ArrowStyle = AxisArrowStyle.None;

}

private void rbtnGrotyStrzalek\_CheckedChanged(object sender, EventArgs e)

{

chtGraficznaWizualizacja.ChartAreas[0].AxisX.ArrowStyle = AxisArrowStyle.Triangle;

chtGraficznaWizualizacja.ChartAreas[0].AxisY.ArrowStyle = AxisArrowStyle.Triangle;

}

private void txtGruboscLiniiWykr\_TextChanged(object sender, EventArgs e)

{

errorProvider1.Dispose();

int angruboscliniiwykresu;

if(!int.TryParse(txtGruboscLiniiWykr.Text, out angruboscliniiwykresu))

{

errorProvider1.SetError(txtGruboscLiniiWykr, "Nieprawidłowy znak");

return;

}

else if(angruboscliniiwykresu > 10)

{

errorProvider1.SetError(txtGruboscLiniiWykr, "Podana wartość jest za duża (maksymalna wartość wynosi 10)");

return;

}

else if (angruboscliniiwykresu < 1)

{

errorProvider1.SetError(txtGruboscLiniiWykr, "Podana wartość jest za mała (minimalna wartość wynosi 1)");

return;

}

tbarGruboscLiniWykresu.Value = angruboscliniiwykresu;

chtGraficznaWizualizacja.Series[0].BorderWidth = angruboscliniiwykresu;

}

private void rozmiarCzcionkiToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (fontDialog1.ShowDialog() == DialogResult.OK)

{

dgvTabela.Font = fontDialog1.Font;

}

}

private void punktowyToolStripMenuItem\_Click(object sender, EventArgs e)

{

chtGraficznaWizualizacja.Series[0].ChartType = SeriesChartType.Point;

}

private void comboxStylLiniiWykresu\_SelectedIndexChanged(object sender, EventArgs e)

{

if (comboxStylLiniiWykresu.SelectedIndex == 0)

{

chtGraficznaWizualizacja.Series[0].BorderDashStyle = ChartDashStyle.Dot;

}

else if (comboxStylLiniiWykresu.SelectedIndex == 1)

{

chtGraficznaWizualizacja.Series[0].BorderDashStyle = ChartDashStyle.Dash;

}

else if (comboxStylLiniiWykresu.SelectedIndex == 2)

{

chtGraficznaWizualizacja.Series[0].BorderDashStyle = ChartDashStyle.DashDot;

}

else if (comboxStylLiniiWykresu.SelectedIndex == 3)

{

chtGraficznaWizualizacja.Series[0].BorderDashStyle = ChartDashStyle.Solid;

}

else if (comboxStylLiniiWykresu.SelectedIndex == 4)

{

chtGraficznaWizualizacja.Series[0].BorderDashStyle = ChartDashStyle.DashDotDot;

}

}

private void kropkowaToolStripMenuItem\_Click(object sender, EventArgs e)

{

chtGraficznaWizualizacja.Series[0].BorderDashStyle = ChartDashStyle.Dot;

comboxStylLiniiWykresu.SelectedIndex = 0;

}

private void kreskowaToolStripMenuItem\_Click(object sender, EventArgs e)

{

chtGraficznaWizualizacja.Series[0].BorderDashStyle = ChartDashStyle.Dash;

comboxStylLiniiWykresu.SelectedIndex = 1;

}

private void kreskowokropkowwaToolStripMenuItem\_Click(object sender, EventArgs e)

{

chtGraficznaWizualizacja.Series[0].BorderDashStyle = ChartDashStyle.DashDot;

comboxStylLiniiWykresu.SelectedIndex = 2;

}

private void ciągłaToolStripMenuItem\_Click(object sender, EventArgs e)

{

chtGraficznaWizualizacja.Series[0].BorderDashStyle = ChartDashStyle.Solid;

comboxStylLiniiWykresu.SelectedIndex = 3;

}

private void kreskowakropkowaKropkowaToolStripMenuItem\_Click(object sender, EventArgs e)

{

chtGraficznaWizualizacja.Series[0].BorderDashStyle = ChartDashStyle.DashDotDot;

comboxStylLiniiWykresu.SelectedIndex = 4;

}

private void krójPismaToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (fontDialog1.ShowDialog() == DialogResult.OK)

{

chtGraficznaWizualizacja.ChartAreas[0].AxisX.TitleFont = fontDialog1.Font;

chtGraficznaWizualizacja.ChartAreas[0].AxisY.TitleFont = fontDialog1.Font;

chtGraficznaWizualizacja.Series[0].Font = fontDialog1.Font;

}

}

bool anPobieranieDanychDoCalkowania(out double anepsSzeregu, out double anXdCalki, out double anXgCalki, out double anepsCalki)

{

anepsSzeregu = anXdCalki = anXgCalki = anepsCalki = 0;

errorProvider1.Dispose();

if (anpobieranieEps(out anepsSzeregu) && anpobieranieEpsCalki(out anepsCalki) && anpobieranieXdXgCalki(out anXdCalki, out anXgCalki))

return true;

return false;

}

double anObliczanieCalkiMetodaProstokatow(double anepsSzeregu, double ana, double anb, double anepsCalki,

out int anLicznik, out double anSzerokoscPrzedzialu)

{

double anH =0, anCi, anCi\_1, anSuma, anX =0;

ushort anp;

anLicznik = 1;

anCi = (anb - ana) \* anSumaSzereguPotegowego((ana + anb) / 2, anepsSzeregu, out anp);

do

{

anCi\_1 = anCi;

anLicznik++;

anH = (anb-ana)/anLicznik;

anX = ana + anH / 2;

anSuma = 0;

for (ushort ani = 0; ani < anLicznik; ani++)

{

anSuma += anSumaSzereguPotegowego(anX + ani \* anH , anepsSzeregu, out anp);

}

anCi = anH \* anSuma;

} while (Math.Abs(anCi - anCi\_1) > anepsCalki);

anSzerokoscPrzedzialu = anH ;

return anCi;

}

private void btnObliczCalke\_Click(object sender, EventArgs e)

{

int anLicznik;

double anepsSzeregu, anXdCalki, anXgCalki, anepsCalki, anSzerokoscPrzedzialu;

if (!anPobieranieDanychDoCalkowania(out anepsSzeregu, out anXdCalki, out anXgCalki, out anepsCalki))

{

return;

}

txtWartoscCalki.Text = anObliczanieCalkiMetodaProstokatow(anepsSzeregu, anXdCalki, anXgCalki, anepsCalki,

out anLicznik, out anSzerokoscPrzedzialu).ToString();

}

}

}

Ci = H \* Suma;

} while (Math.Abs(Ci - Ci\_1) > epsCalki);

SzerokoscPrzedzialu = H;

return Ci;

}

private void btnObliczCalke\_Click(object sender, EventArgs e)

{

int Licznik;

double epsSzeregu, XdCalki, XgCalki, epsCalki, SzerokoscPrzedzialu;

if (!PobieranieDanychDoCalkowania(out epsSzeregu, out XdCalki, out XgCalki, out epsCalki))

{

return;

}

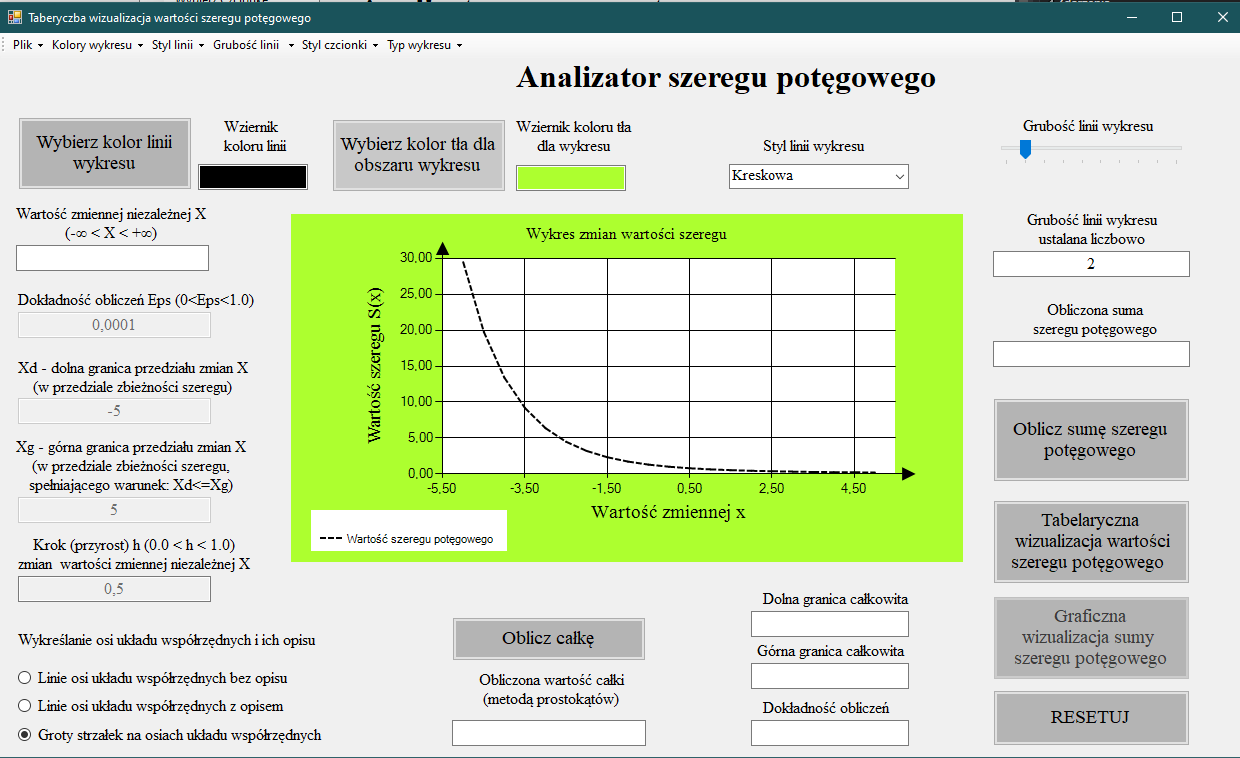
txtWartoscCalki.Text = ObliczanieCalkiMetodaProstokatow(epsSzeregu, XdCalki, XgCalki, epsCalki,

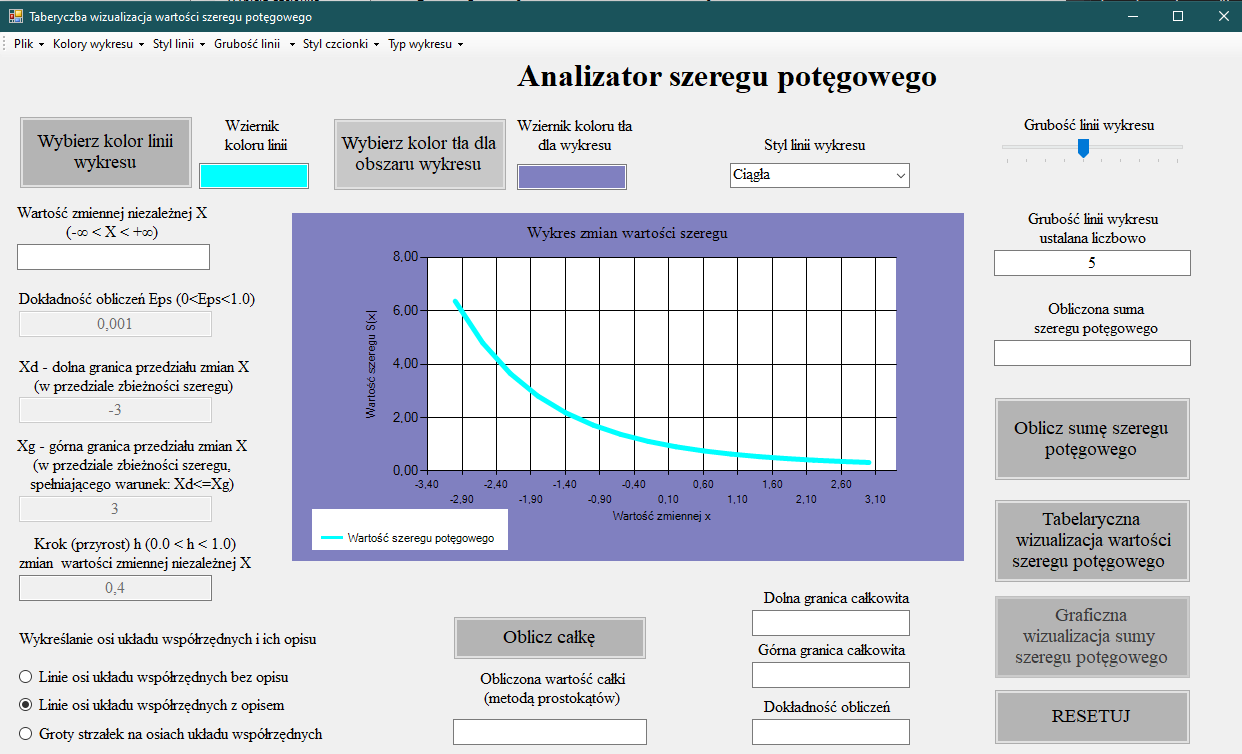
out Licznik, out SzerokoscPrzedzialu).ToString();

}

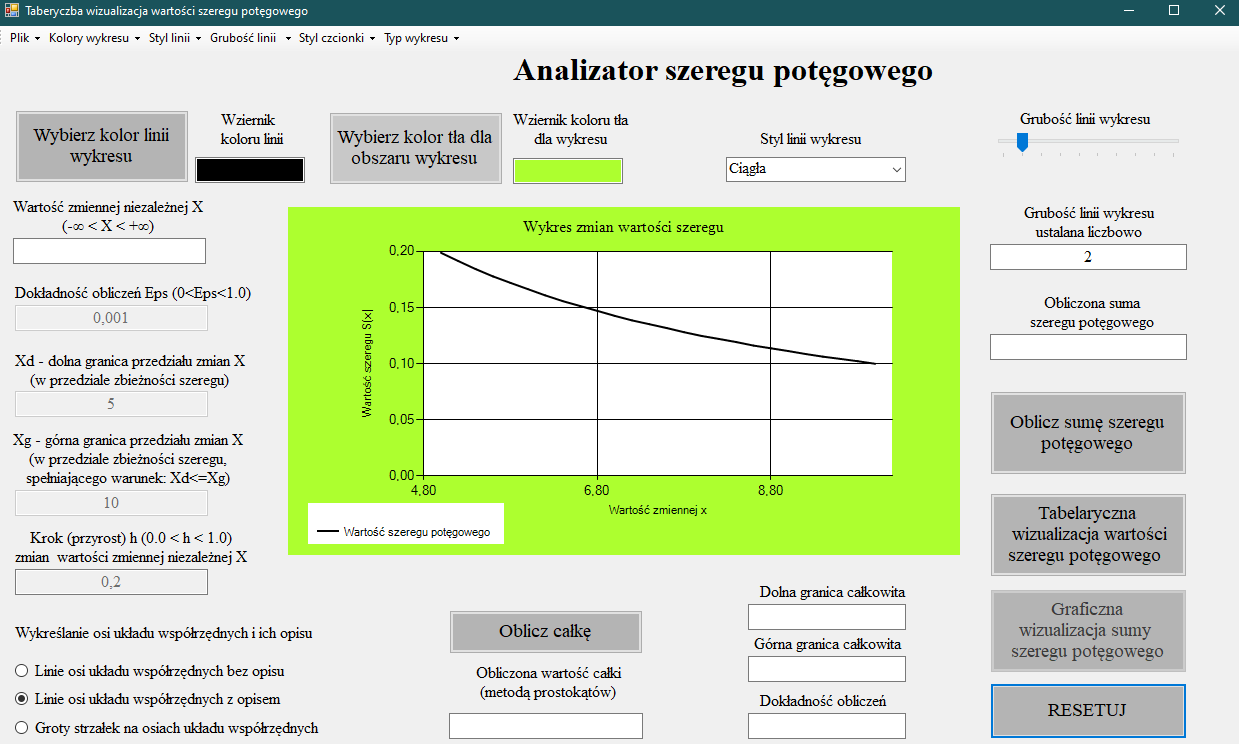
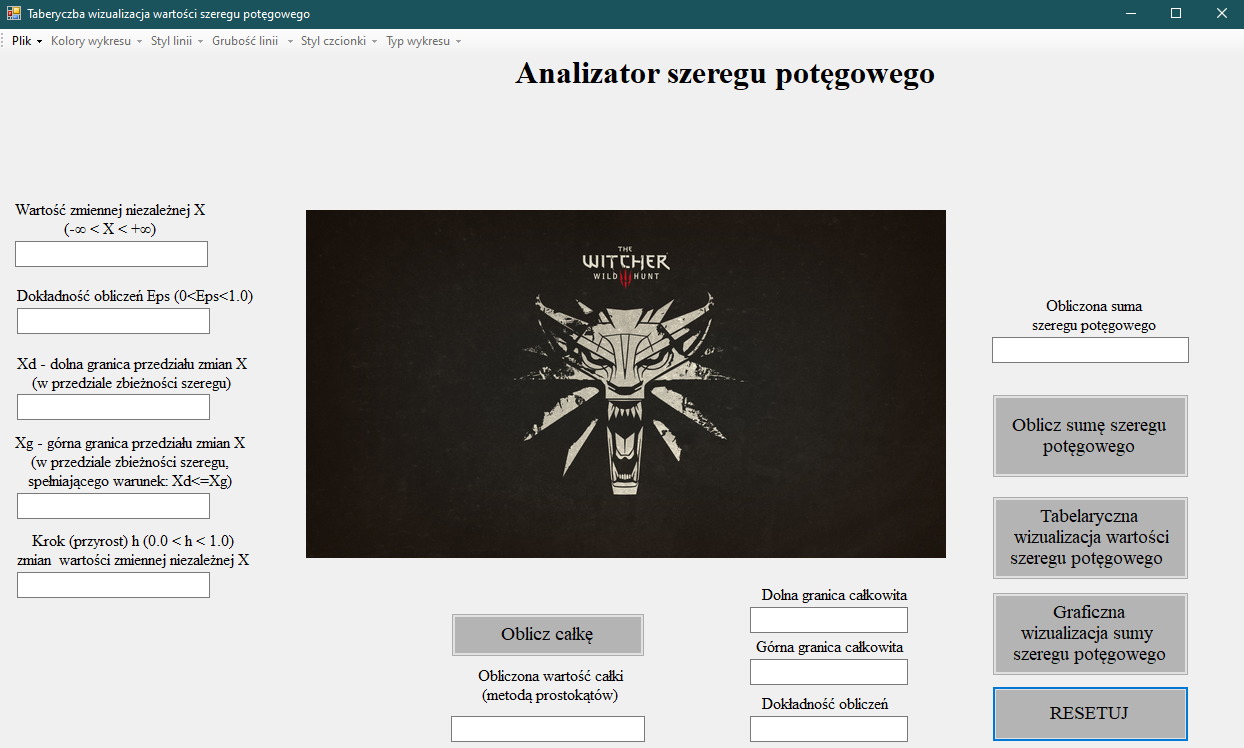
Testy:

a)Zmiana typu wykresu na kreskowy, zmiana rozmiaru czcionki opisów osi oraz dodanie grotów strzałek na końcach osi.



b)Zmiana koloru tła wykresu, zmiana koloru linii wykresu, zmiana grubości linii wykresu.

d)Sprawdzenie przycisku RESETUJ polegające na: wyświetleniu graficznej wizualizacji sumy szeregu potęgowego zmieniając jej właściwości takie jak kolor tła wykresu, kolor linii, naciśnięciu przycisku RESETUJ, wyświetleniu wizualizacji sumy szeregu potęgowego sumy szeregu potęgowego z innymi danymi aby zobaczyć czy właściwości wykresu wrócą do bazowych.



Wnioski:

-Programowanie obiektowe jest skomplikowane, ale jednocześnie ciekawe.

Samoocena:

-Cały pasek menu działa bez zastrzeżeń, wszystkie kontrolki też. Oceniłbym się na 5.