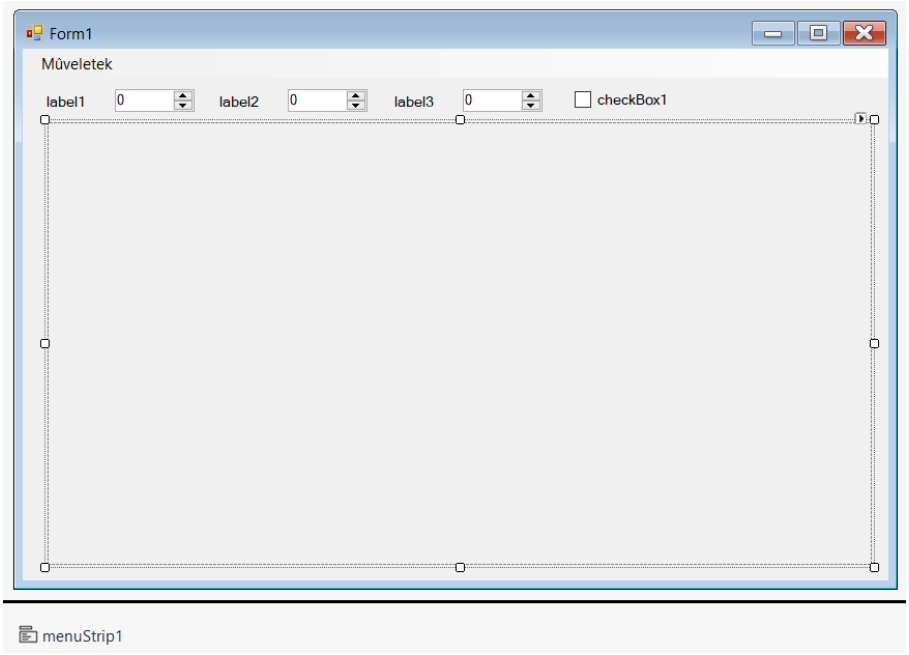
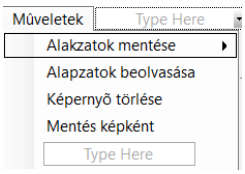
```
    using namespace System::Collections::Generic;
    using namespace System::IO;

    ref class Alakzat
    {
        //Alakzat adatai
    public: // private kellene, de akkor kell property is!
        int x, y;
        int csucs;
        int meret;
        int szog;
        Color szin;
    public:
        Alakzat(int x, int y, int cs, int k, int elf, Color c)
        {
            this->x = x;
            this->y = y;
            csucs = cs;
            meret = k;
            szog = elf;
            szin = c;
        }

        // Alakzat rajzolása átadott Graphics felületre
        void DrawAlakzat(Graphics ^gr)
        {
            array<PointF>^ sikidom = gcnew array<PointF>(csucs);
            for (int j = 0; j < csucs; j++)
            {
                sikidom[j].X = meret * Math::Sin(j * 2 * Math::PI / csucs);
                sikidom[j].Y = meret * Math::Cos(j * 2 * Math::PI / csucs);
            }
            gr->ResetTransform();
            gr->TranslateTransform(x, y);
            gr->RotateTransform(szog);
            gr->FillPolygon(gcnew SolidBrush(szin), sikidom);
        }

        // fájl mentéshez
        String ^ ToString() override
        {
            return x + "\t" + y + "\t" + csucs + "\t" + meret + "\t" + szog + "\t" +
            szin.ToArgb();
        }
    };

    //Globális változók
    Graphics^ gr;
    Bitmap^ bm;
    Random^ randi = gcnew Random();
    // ebben a listában kerülnek tárolásra az Alakzat objektumok
    List<Alakzat^>^ elemek = gcnew List<Alakzat ^>();

// ablak indul
private: System::Void Form1_Load(System::Object^  sender, System::EventArgs^  e) {
    bm = gcnew Bitmap(pictureBox1->Width, pictureBox1->Height);
    gr = Graphics::FromImage(bm);
    numericUpDown1->Minimum = 3;
    numericUpDown2->Minimum = 10;
    numericUpDown3->Maximum = 360;
    label1->Text = "Sokszög";
    label2->Text = "Méret";
    label3->Text = "Irány";
    checkBox1->Text = "Random";
    Text = "Alakzatok";
}
```

```
// egér klikkelés eseménye
// e->Button visszaadja, hogy melyik gommal történt a klikkelés
// e->X és e->Y megadja a klikkelés helyét
private: System::Void pictureBox1_MouseClick(System::Object^  sender,
System::Windows::Forms::MouseEventArgs^  e) {
    if (e->Button == System::Windows::Forms::MouseButtons::Left)    //bal gomb
    {
        // alakzat méretei a vezérlőkről
        int n = (int)numericUpDown1->Value;
        int l = (int)numericUpDown2->Value;
        int f = (int)numericUpDown3->Value;
        if (checkBox1->Checked) //ha be van kapcsolva a Random
        {   //véletlen méretek
            n = randi->Next(3, 10);
            l = randi->Next(10, 100);
            f = randi->Next(361);
        }
        //mindig változó színek
        Color szin = Color::FromArgb(randi->Next(256), randi->Next(256), randi->Next(256));
        //alakzat hozzáadása a listához
        elemek->Add(gcnew Alakzat(e->X, e->Y, n, l, f, szin));
    }
    else if (e->Button == System::Windows::Forms::MouseButtons::Right)  //jobb gomb
    {   //utolsó elem törlése
        if (elemek->Count > 0)
            elemek->RemoveAt(elemek->Count - 1);
    }
    // új lista kirajzoltatása
    Rajz();
}

//A listában található elemek kirajzolása
public: void Rajz(){
    //koordináta rendszer visszaállítása
    gr->ResetTransform();
    gr->Clear(Color::Transparent);

    int numb = elemek->Count;
    //összekötő vonalak rajzolása
    if (numb > 2) //vonalak kirajzolása
    {   //a pontok számára létrehozott tömb
        array<Point>^ vegpontok = gcnew  array<Point>(numb);
        for (int i = 0; i < numb; i++)
            vegpontok[i] = Point(elemek[i]->x, elemek[i]->y);
        // tömbben szereplő végpontok vonalakkal történő kirajzolása
        gr->DrawPolygon(Pens::Black, vegpontok);
    }
    //alakzatok rajzolása
    for (int i = 0; i < numb; i++)
    {   //az osztályon belül létrehozott rajzoló függvény használatával
        elemek[i]->DrawAlakzat(gr);
    }
    //megjelenítés a picturebox-on
    pictureBox1->Image = bm;
}

// képernyő törlése
private: System::Void képernyõTörléseToolStripMenuItem_Click(System::Object^  sender,
System::EventArgs^  e) {
    elemek->Clear();
    gr->Clear(Color::Transparent);
    pictureBox1->Image = bm;
}

// alakzatok adatainak elmentése
private: System::Void alakzatokElmentéseToolStripMenuItem_Click(System::Object^  sender,
System::EventArgs^  e) {
```

```
    SaveFileDialog^ sv = gcnew SaveFileDialog();        //mentés dialógus ablak
    sv->FileName = "";
    sv->Filter = "Pontok|*.txt";
    if (sv->ShowDialog() == System::Windows::Forms::DialogResult::OK)   // ha megadtunk fájlt
    {
        StreamWriter^ sw = gcnew StreamWriter(sv->FileName);    //fájl megnyitása  írásra
        for each (Alakzat^ elem in elemek)                      //elemek lista bejárása
        {
            //a lista elemeinek konvertálása szöveggé és soronkénti kiírása szövegfájlba
            sw->WriteLine(elem->ToString());
        }
        sw->Flush();
        sw->Close();    //bezárás
    }
}

// alakzat adatainak visszatöltése fájlból
private: System::Void alakzatokBetöltéseToolStripMenuItem_Click(System::Object^  sender,
System::EventArgs^  e) {
    OpenFileDialog^ ofd = gcnew OpenFileDialog();       // megnyitás dialógus ablak
    ofd->FileName = "";
    ofd->Filter = "Pontok|*.txt";
    if (ofd->ShowDialog() == System::Windows::Forms::DialogResult::OK)  // ha megadtunk fájlt
    {
        StreamReader^ sr = gcnew StreamReader(ofd->FileName);   //fájl megnyitása  olvasása

        elemek->Clear();                                        //lista ürítése
        while (!sr->EndOfStream)
        {
            String^ sor = sr->ReadLine();                       //soronkénti olvasás
            if (sor->Contains("\t"))                            // van benne adat
            {
                // elemek kigyűjtése és konvertálása
                array<String^>^ resz = sor->Split('\t');
                int x = Convert::ToInt32(resz[0]);
                int y = Convert::ToInt32(resz[1]);
                int n = Convert::ToInt32(resz[2]);
                int l = Convert::ToInt32(resz[3]);
                int f = Convert::ToInt32(resz[4]);
                Color szin = Color::FromArgb(Convert::ToInt32(resz[5]));
                //alakzat hozzáadása a listához
                elemek->Add(gcnew Alakzat(x, y, n, l, f, szin));
            }
        }
        sr->Close();    //bezárás
        // új lista kirajzolása
        Rajz();
    }
}

// ablak átméretezésekor újra számítás és megjelenítés
private: System::Void Form1_Resize(System::Object^  sender, System::EventArgs^  e) {
    bm = gcnew Bitmap(pictureBox1->Width, pictureBox1->Height);
    gr = Graphics::FromImage(bm);
    Rajz();
}

// kép mentése átlátszó png-ként
private: System::Void mentésKépkéntToolStripMenuItem_Click(System::Object^  sender,
System::EventArgs^  e) {
    bm->Save("background.png", System::Drawing::Imaging::ImageFormat::Png);
}
```


Form1
Műveletek
label1
0
label2
0
label3
0
checkBox1
menuStrip1


Műveletek
Type Here
Alakzatok mentése
Alapzatok beolvasása
Képernyő törlése
Mentés képként
Type Here