```
using namespace System::IO;     // fájlkezelés

-----------------------------------------------------------------

// Globális változók
const int maxpsz = 100;          // maximális pontszám
// a fájlokat tegyük a Form1.h mellé
array<int> ^hiszt;               // tömb a gyakoriság számára
String ^filename = "infrsz.txt"; // adatfájl, szeparáló elem: tab
String ^logo = "mogi.png";       // logo
Bitmap ^bm;
Image ^kep;
StreamReader ^sr;
Graphics ^gr;
Pen ^toll;
SolidBrush ^ecset;
System::Drawing::Font ^font;
Color szin;

// ablak betöltése
private: System::Void Form1_Load(System::Object^  sender, System::EventArgs^  e) {
    this->Text = "Grafika";
    button1->Text = "Adatok betöltése";

    hiszt = gcnew array<int>(maxpsz + 1);   // hisztogram adatainak tárolója

    if (File::Exists(logo))                 // kép meglétének ellenőrzése
    {
        bm = gcnew Bitmap(logo);            // kép betöltése egy Bitmap-ba
        pictureBox1->Image = bm;            // kép betöltése a pictureboxba
        pictureBox1->Width = bm->Width;
        pictureBox1->Height = bm->Height;
    }

    toll = gcnew Pen(Color::Black);           // fekete toll
    ecset = gcnew SolidBrush(Color::Red);     // vörös ecset
    font = gcnew System::Drawing::Font("Arial", 10);    // betűstílus: Arial, 10-es fontméret
    szin = Color::Azure;                      // háttérszín
}

// Szövegfájl feldolgozása
private: void feldolgoz()
{
    if (File::Exists(filename))                                 // ha létezik a fájl
    {
        for (int i = 0; i < maxpsz + 1; i++) hiszt[i] = 0;   // előző hisztogram törlése

        sr = gcnew StreamReader(filename);                      // fájl megnyítása olvasásra
        while (!sr->EndOfStream)                                // a fájl végéig olvas
        {
            String ^sor = sr->ReadLine();                       // az egész sor kiolvasása
            if (sor->Contains("\t"))                            // mérés pontszámát tartalmazó
            sor
            {
                // kiszűröm a szövegből a neptunkódot
                String ^neptunkod = sor->Substring(0, sor->IndexOf('\t'));
                // kiszedem a pontszámot
                String ^pontszam = sor->Substring(sor->IndexOf('\t') + 1);
                int szam = Convert::ToInt32(pontszam);  // szöveg -> szám konverzió
                hiszt[szam]++;   //a pontszám szerinti találatot növelem eggyel
            }
        }
        sr->Close();                                            // fájl bezárása
    }
}
```

```
    //  Rajzolás - külön függvény, hogy többször is hívható legyen
    private: void grafikon()
    {
        // Graphics panel1-hez rendelése
        // koordináta rendszer origója a panel1 bal felső sarka
        // x vizszíntesen növekszik
        // y lefelé növekszik
        gr = panel1->CreateGraphics();

        //gr->Clear(System->Drawing->SystemColors->Control);      //törlés az eredeti színnel

        int w = panel1->Width - 1;  // képpontok 0-tól
        int h = panel1->Height - 1;
        float dx = (float)w / (maxpsz + 1); // oszlopok szélessége

        // Háttér (kitöltés + keret)
        ecset->Color = szin;
        gr->FillRectangle(ecset, 0, 0, w, h);
        gr->DrawRectangle(toll, 0, 0, w, h);

        // Maximum keresés a normalizáláshoz
        int hmax = 0;
        for (int i = 0; i < maxpsz + 1; i++)
            if (hiszt[i] > hmax) hmax = hiszt[i];

        // gyakorisághoz tartozó oszlopok kirajzolása
        for (int i = 0; i < maxpsz + 1; i++)
        {
            float szam = hiszt[i];              // eredeti
            szam = (float)h / hmax * hiszt[i];  // normalizált

            // Színezés
            if (i < 40) ecset->Color = Color::Silver;
            else if (i < 55) ecset->Color = Color::Red;
            else if (i < 70) ecset->Color = Color::Yellow;
            else if (i < 85) ecset->Color = Color::GreenYellow;
            else  ecset->Color = Color::Green;

            // színes oszlopok
            gr->FillRectangle(ecset, i * dx, h - szam, dx, szam);
            gr->DrawRectangle(toll, i * dx, h - szam + 1, dx - 1, szam);
            gr->DrawRectangle(toll, i * dx, h - szam, dx, szam);
        }

        // Szöveg megjelenítése
        gr->DrawString("Informatikai rendszerek", font, ecset, 10, 10);

        // Logo megjelenítése
        kep = gcnew Bitmap(logo);
        gr->DrawImage(kep, w - kep->Width - 10, 10);

    }

    // gombnyomásra szövegfájl beolvasása és hisztogram megjelenítése
    private: System::Void button1_Click(System::Object^  sender, System::EventArgs^  e) {
        // szövegfájl feldolgozása
        feldolgoz();
        // grafikon kirajzolása
        grafikon();
    }

    // adjuk hozzá a panel1-hez a MouseDown eseményt
    // egérgombok lekezelése
    private: System::Void panel1_MouseDown(System::Object^  sender,
    System::Windows::Forms::MouseEventArgs^  e) {
        if (e->Button == System::Windows::Forms::MouseButtons::Left)    // Bal egérgomb
```

```
        {
            // font dialógus ablak
            FontDialog ^ftg = gcnew FontDialog();
            ftg->Font = font;
            if (ftg->ShowDialog() == System::Windows::Forms::DialogResult::OK)
                font = ftg->Font;
        }
        if (e->Button == System::Windows::Forms::MouseButtons::Right)   // Jobb egérgomb
        {
            // szín dialógus ablak
            ColorDialog ^clg = gcnew ColorDialog();
            clg->Color = szin;
            if (clg->ShowDialog() == System::Windows::Forms::DialogResult::OK)
                szin = clg->Color;
        }
        // grafikon megjelenítése az új színnel vagy fonttal
        grafikon();
    }

    // adjuk hozzá a Form1-hez a Resize eseményt
    // ablak átméretezésekor fog lefutni
    private: System::Void Form1_Resize(System::Object^  sender, System::EventArgs^  e) {
        grafikon(); // grafikon újra rajzolása
    }
```