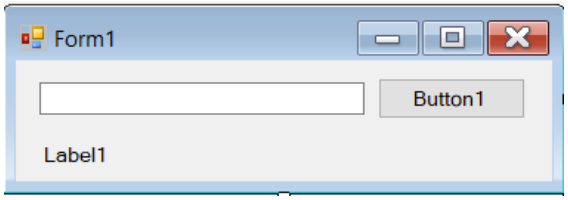


```
// WindowsForm.cpp : main project file.

#include "stdafx.h"
#include "Form1.h"


// kevert kód
// Properties ->General -> Common Language Runtime Support (/clr)
#pragma unmanaged       // kevert kód VS2017-ben már nem is kell kiírni, tudja
// natív cpp osztályok, adattagok
#include <iostream>
#include <iomanip>
#include <string>
#include <list>
using namespace std;

// tantárgy publikus adattagokkal
struct tantargy_n
{
    string nev;
    int kredit;
    // konstruktor alapértelmezett értékekkel
    tantargy_n(string nev = "???", int kredit = 0)
    {
        this->nev = nev;
        this->kredit = kredit;
    }
    ~tantargy_n()   // debug, hogy lássuk mikor szűnik meg
    {
        cout << this->nev << " uritve" << endl;
    }
};

tantargy_n t;   //statikus változó
// STL lista és iterátora a bejáráshoz
list<tantargy_n> szemeszter;
list<tantargy_n>::iterator it;

#pragma managed

using namespace System;
using namespace WindowsForm;        // Form1 ezen a névterületen van

[STAThreadAttribute]

// using namespace esetén a main-ban található array-t
// át kell írni cli::array-ra
int main(cli::array<System::String ^> ^args)
{
    // ismétlés előző félévből:

    //elemek feltöltése
    t.nev = "Modellezes es ..."; t.kredit = 6;
    szemeszter.push_back(t);
    t.nev = "Dinamika"; t.kredit = 5;
    szemeszter.push_back(t);
    szemeszter.push_back(tantargy_n()); //???
    szemeszter.push_back(tantargy_n("Gepelemek",6));

    cout << "==================================" << endl;
    cout << setw(20) << "TANTARGY" << setw(10) << "KREDIT" << endl;
    cout << "==================================" << endl;

    int szum = 0;
    for (it = szemeszter.begin(); it != szemeszter.end(); it++)
    {
        cout << setw(20) << it->nev << setw(10) << it->kredit << endl;
        szum += it->kredit;
    }

    cout << "==================================" << endl;
```

```
    cout << "OSSZESEN: " << szemeszter.size() << " DB TANTARGY" << endl;
    cout << "OSSZESEN: " << szum << " KREDIT" << endl;
    cout << "==================================" << endl;


    // innen idul az idei félév :)
    // ha már nincs szükségünk a konzolra, akkor Windows felület
    // Properties -> Linker -> General -> SubSystem = Windows (/SUBSYSTEM:WINDOWS)
    // belépési pont pedig main
    // Properties -> Linker -> Advanced -> Entry Point = main
    Console::WriteLine(L"Ablak indul");
    Application::Run(gcnew Form1());
    return 0;
}




/*-------------------------------------------------------------------------------------
Form1.h -ba beírandó dolgok
-------------------------------------------------------------------------------------*/

    // CPP CLI referencia osztály, a menedzselt halmon
    ref class tantargy_r
    {
    private:
        String ^ nev;
        int kredit;
    public:
        // konstuktor, itt nincs alapértelmezett érték
        tantargy_r(String ^nev, int kredit)
        {
            this->nev = nev;
            this->kredit = kredit;
        }

        // tagelérő metódus helyett használhatunk property-ket
        // nem kell () a használatukhoz
        // get -rész kiolvasás
        // set -rész értékadás
        property int Kredit
        {
            int get() { return this->kredit; }
            void set(int kredit) { this->kredit = kredit; }
        }

        // csak olvasható property (set hiányzik)
        property String ^ Nev
        {
            String ^ get() { return this->nev; }

        }

    };

    // globális változók
    tantargy_r ^targy;  // managed osztálypéldány esetén kell a ^
    // CLI sablon alapú lista osztály, meg kell adni az elemek típusát <> között
    List<tantargy_r ^> ^felev;

    //ablak betöltődésekor fog lefutni ez az eseménykezelő
    private: System::Void Form1_Load(System::Object^  sender, System::EventArgs^  e) {
        // ablak fejlécének szövege
        this->Text = "Első gyakorlat";
        // gomb felirata
        button1->Text = "Keresés";

        // managed osztálypéldány létrehozása mindig gcnew-val történik
        targy = gcnew tantargy_r("Modellezés és mérésadatgyűjtés szoftverei", 6);
        felev = gcnew List<tantargy_r ^>();     //üres lista létrehozása
        // adott tantárgy hozzáadása a listához
```

```
        felev->Add(targy);

        // új tantárgy létrehozása
        targy = gcnew tantargy_r("Dinamika", 2);
        // elem hozzáadása a listához
        felev->Add(targy);
        // a listába adott elem csak egy referencia, ezért amíg nem hozok létre egy újabb
        példányt erre a változóra,
        // addig a lista eleme is módosulni fog
        targy->Kredit = 5;

        //elem közvetlen hozzáadása a listához
        felev->Add(gcnew tantargy_r("Gepelemek", 6));

        // lista elemeinek száma ->Count property (nem kell zárójel)
        label1->Text = Convert::ToString(felev->Count) + " db tantárgy";
    }


    // gomb megnyomásának eseménye
    private: System::Void button1_Click(System::Object^  sender, System::EventArgs^  e) {

        bool van = false;
        for (int i = 0; i < felev->Count; i++)      // a lista összes elemét végig nézzük
        {
            // a lista bejárásához nem kell iterátor, mert van indexük []
            if (felev[i]->Nev == textBox1->Text)    // teljes egyezés kell
            {
                van = true;
                // ha megvan, akkor kredit értékét konvertálni kell szövegé
                // vagy .ToString() metódussal
                // vagy Convert::ToString(elev[i]->Kredit) metódussal
                label1->Text = felev[i]->Kredit.ToString() + " kredit";
            }
        }
        //ha nem található az adott tantárgy
        if (!van) label1->Text = "Nincs ilyen tárgy!";
    }
```