МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное

образовательное учреждение высшего образования

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

ФАКУЛЬТЕТ АВТОМАТИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

Кафедра Автоматизированных систем управления

**ОТЧЕТ О ВЫПОЛНЕНИИ**

**Лабораторной работы № 3**

Дисциплина «*Технологии и методы программирования*»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Выполнили:** |  | **Проверил:** |  | |
| Студенты | *Бузмаков А.И.,*  *Шестаков К.Д.* | Преподаватель | *Астапчук В. А., Пустовских Д. А.* | |
| Факультет | *АВТФ* |  |  | |
| Направление (специальность) подготовки | *09.03.01 – Информатика и вычислительная техника* | Балл: \_\_\_\_\_\_\_\_\_ | |  |
| Группа | *АП-227* | Оценка *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_* | | |
| Шифр |  |  |  | |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | |
| подпись | | подпись | | |
| Дата сдачи: | «13» апреля 2024г. | Дата защиты: | «\_\_»\_\_\_\_\_\_\_\_\_\_20\_\_г. | |

Новосибирск 2024

**Текст задания**

Описание класса и все модули, созданные при выполнении работы 2, оформить как DLL. Тестирующая программа должна повторить задание работы 2, используя созданную DLL и разные варианты связывания.

**Тексты разработанных программ**

Таблица 1 – Код тестирующей программы

|  |
| --- |
| Program.cs |
| using System;  using System.Windows.Forms;  using System.Reflection;  namespace lab3  {  internal static class Program  {  /// <summary>  /// The main entry point for the application.  /// </summary>  [STAThread]  static void Main()  {  Application.EnableVisualStyles();  Application.SetCompatibleTextRenderingDefault(false);  // Позднее связывание  Assembly asm = Assembly.LoadFrom("laba2.dll");  dynamic t = asm.GetType("laba2.AuthForm");  dynamic AuthForm = Activator.CreateInstance(t);  Application.Run(AuthForm);  // Раннее связывание  //Form AuthForm = new laba2.AuthForm();  //Application.Run(AuthForm);  }  }  } |

Таблица 2 – Код библиотеки

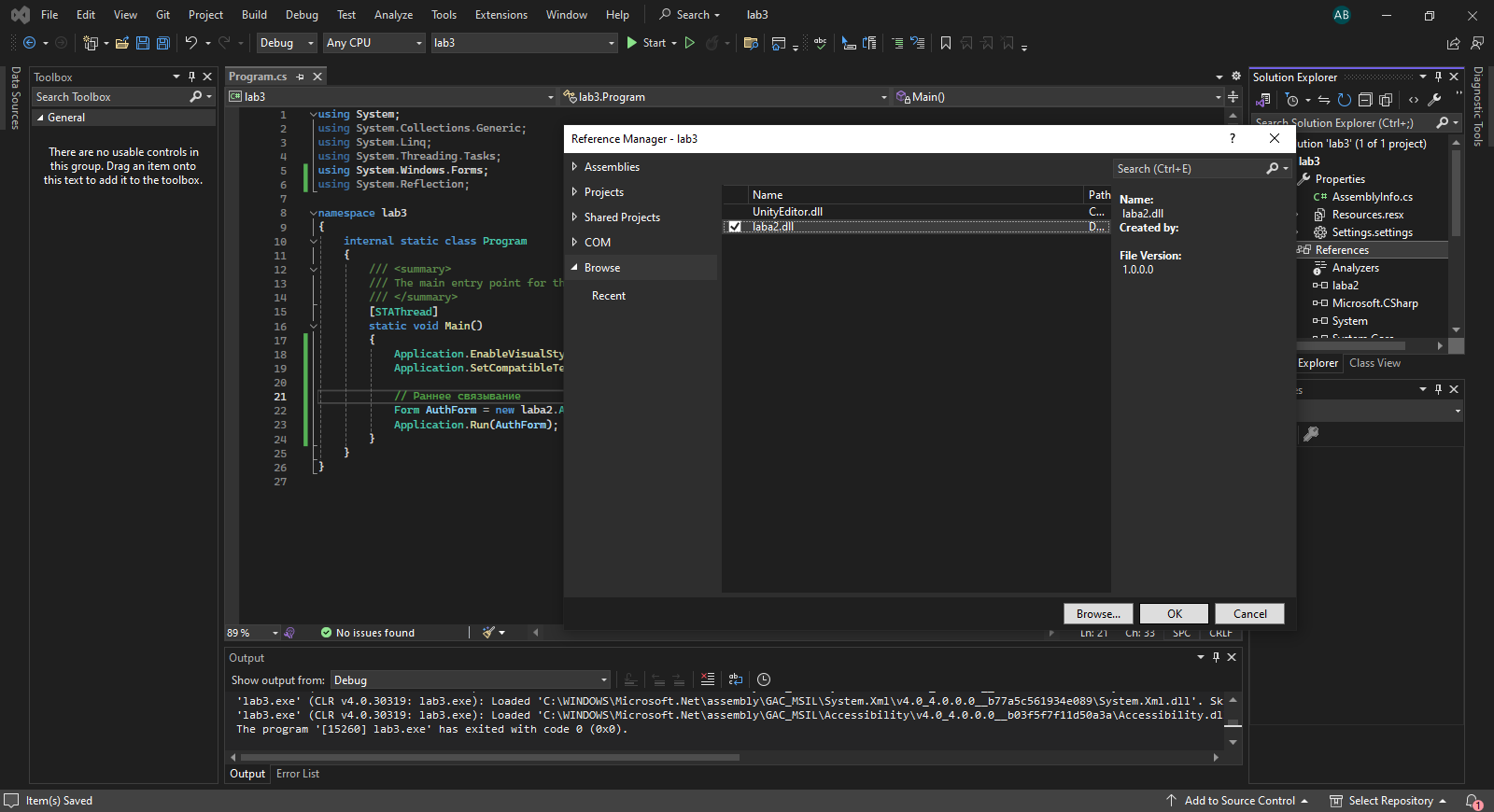
|  |
| --- |
| MainForm.cs |
| using System;  using System.Collections.Generic;  using System.IO;  using System.Windows.Forms;  namespace laba2  {  public partial class MainForm : Form  {  private string \_username;  public MainForm(string username)  {  this.\_username = username;  InitializeComponent();  LoadUserRoles(\_username);  }  // Метод для загрузки ролей пользователя из файла  private void LoadUserRoles(string username)  {  // Словарь для хранения ролей пользователя  Dictionary<string, string> userRoles = new Dictionary<string, string>();  // Чтение строк из файла пользователей  string[] lines = File.ReadAllLines("users.txt");  // Перебор строк файла  for (int i = 0; i < lines.Length; i++)  {  string[] parts = lines[i].Split(' ');  if ("#" + username == parts[0])  {  for (int j = i + 1; j < lines.Length; j++)  {  if (!lines[j].StartsWith("#"))  {  string[] roleParts = lines[j].Split(' ');  string name = "";  string status = "";  for (int k = 0; k < roleParts.Length; k++)  {  bool isNumeric = int.TryParse(roleParts[k], out \_);  if (!isNumeric)  {  name += roleParts[k] + " ";  }  else  {  status = roleParts[k];  }  }  name = name.Trim();  status = status.Trim();  userRoles[name] = status;  }  else  {  break;  }  }  break;  }  }  LoadMenu("menu.txt", menuStrip1, userRoles);  }  // Метод для загрузки меню из файла  private void LoadMenu(string fileName, MenuStrip menuStrip, Dictionary<string, string> userRoles)  {  if (File.Exists(fileName))  {  string[] lines = File.ReadAllLines(fileName);  Dictionary<int, List<ToolStripMenuItem>> menuItemsByLevel = new Dictionary<int, List<ToolStripMenuItem>>();  foreach (string line in lines)  {  string[] parts = line.Split(' ');  if (parts.Length >= 4)  {  int level = int.Parse(parts[0]);  string title = "";  int status = 0;  string methodName = "";  for (int i = 1; i < parts.Length; i++)  {  bool isNumeric = int.TryParse(parts[i], out \_);  if (!isNumeric)  {  title += parts[i] + " ";  }  else  {  status = int.Parse(parts[i]);  methodName = parts[i + 1];  break;  }  }  title = title.Trim();  if (userRoles.ContainsKey(title))  {  status = Int32.Parse(userRoles[title]);  }  ToolStripMenuItem menuItem = new ToolStripMenuItem(title, null, MenuItem\_Click);  menuItem.Tag = methodName;  if (status == 2)  {  menuItem.Visible = false;  }  else if (status == 1)  {  menuItem.Enabled = false;  }  if (!menuItemsByLevel.ContainsKey(level))  {  menuItemsByLevel[level] = new List<ToolStripMenuItem>();  }  menuItemsByLevel[level].Add(menuItem);  if (level == 0)  {  menuStrip.Items.Add(menuItem);  }  else  {  int parentLevel = level - 1;  if (menuItemsByLevel.ContainsKey(parentLevel))  {  foreach (var parentMenuItem in menuItemsByLevel[parentLevel])  {  parentMenuItem.DropDownItems.Add(menuItem);  }  }  }  }  }  }  }  // Метод для поиска родительского пункта меню по уровню  private ToolStripMenuItem FindParentMenuItem(ToolStripItemCollection items, int level)  {  foreach (ToolStripMenuItem item in items)  {  if (item.Tag != null && (int)item.Tag == level)  {  return item;  }  if (item.DropDownItems.Count > 0)  {  ToolStripMenuItem foundItem = FindParentMenuItem(item.DropDownItems, level);  if (foundItem != null)  return foundItem;  }  }  return null;  }  // Обработчик события нажатия на элемент меню  private void MenuItem\_Click(object sender, EventArgs e)  {  ToolStripMenuItem menuItem = (ToolStripMenuItem)sender;  string methodName = menuItem.Tag as string;  if (!string.IsNullOrEmpty(methodName))  {  InvokeMethodByName(methodName);  }  }  // Метод для вызова метода по имени  private void InvokeMethodByName(string methodName)  {  MessageBox.Show($"{methodName}");  }  }  } |

Таблица 3 – Код библиотеки

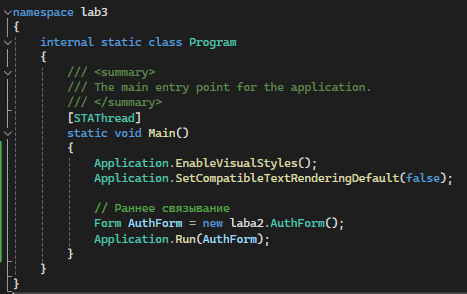
|  |
| --- |
| AuthForm.cs |
| using System;  using System.Collections.Generic;  using System.IO;  using System.Windows.Forms;  namespace laba2  {  public partial class AuthForm : Form  {  public AuthForm()  {  InitializeComponent();  }  // Обработчик нажатия кнопки "Отмена"  private void cancelButton\_Click(object sender, EventArgs e)  {  Application.Exit();  }  // Обработчик нажатия кнопки "Вход"  private void enterButton\_Click(object sender, EventArgs e)  {  // Чтение всех строк из файла пользователей  string[] lines = File.ReadAllLines("users.txt");  // Создание словаря для хранения пар логин-пароль  Dictionary<string, string> users = new Dictionary<string, string>();  // Перебор каждой строки файла  foreach (string line in lines)  {  string[] parts = line.Split(' ');  // Проверка, является ли строка логином и паролем  if (line.StartsWith("#"))  {  // Добавление пары логин-пароль в словарь  users[parts[0]] = parts[1].Trim();  }  }  // Проверка наличия введенного логина в словаре  if (users.ContainsKey("#" + loginBox.Text))  {  // Проверка правильности введенного пароля  if (passBox.Text == users["#" + loginBox.Text])  {  MessageBox.Show("Успешно!"); // Оповещение об успешном входе  MainForm mainForm = new MainForm(loginBox.Text); // Создание главной формы с передачей логина  mainForm.Show(); // Отображение главной формы  }  else  {  MessageBox.Show("Неправильный пароль!"); // Оповещение о неправильном пароле  }  }  else  {  MessageBox.Show("Неправильный логин!"); // Оповещение о неправильном логине  }  }  }  } |

**Ход работы**

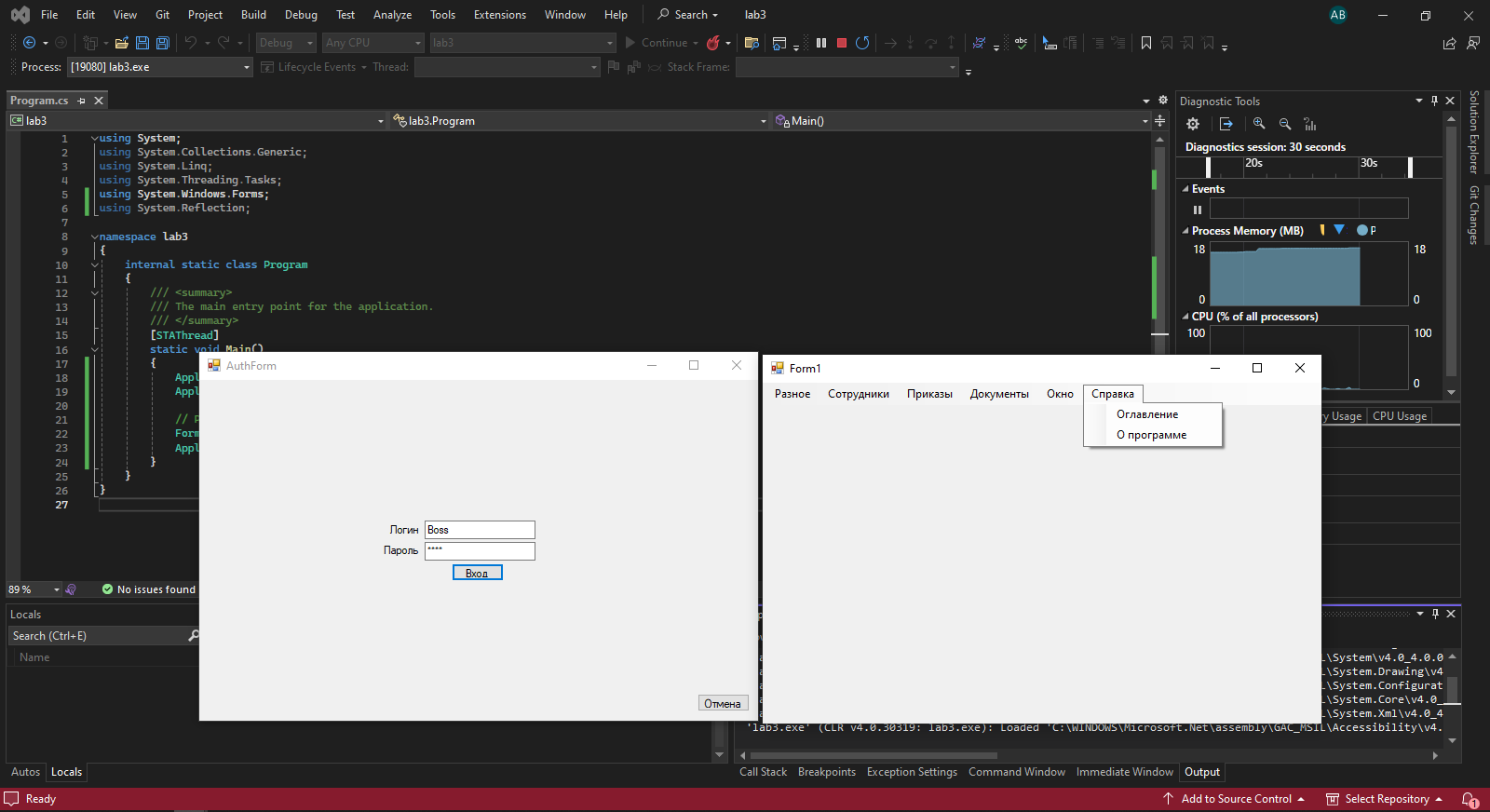
Для раннего связывания добавляем ссылку на DLL в проект:



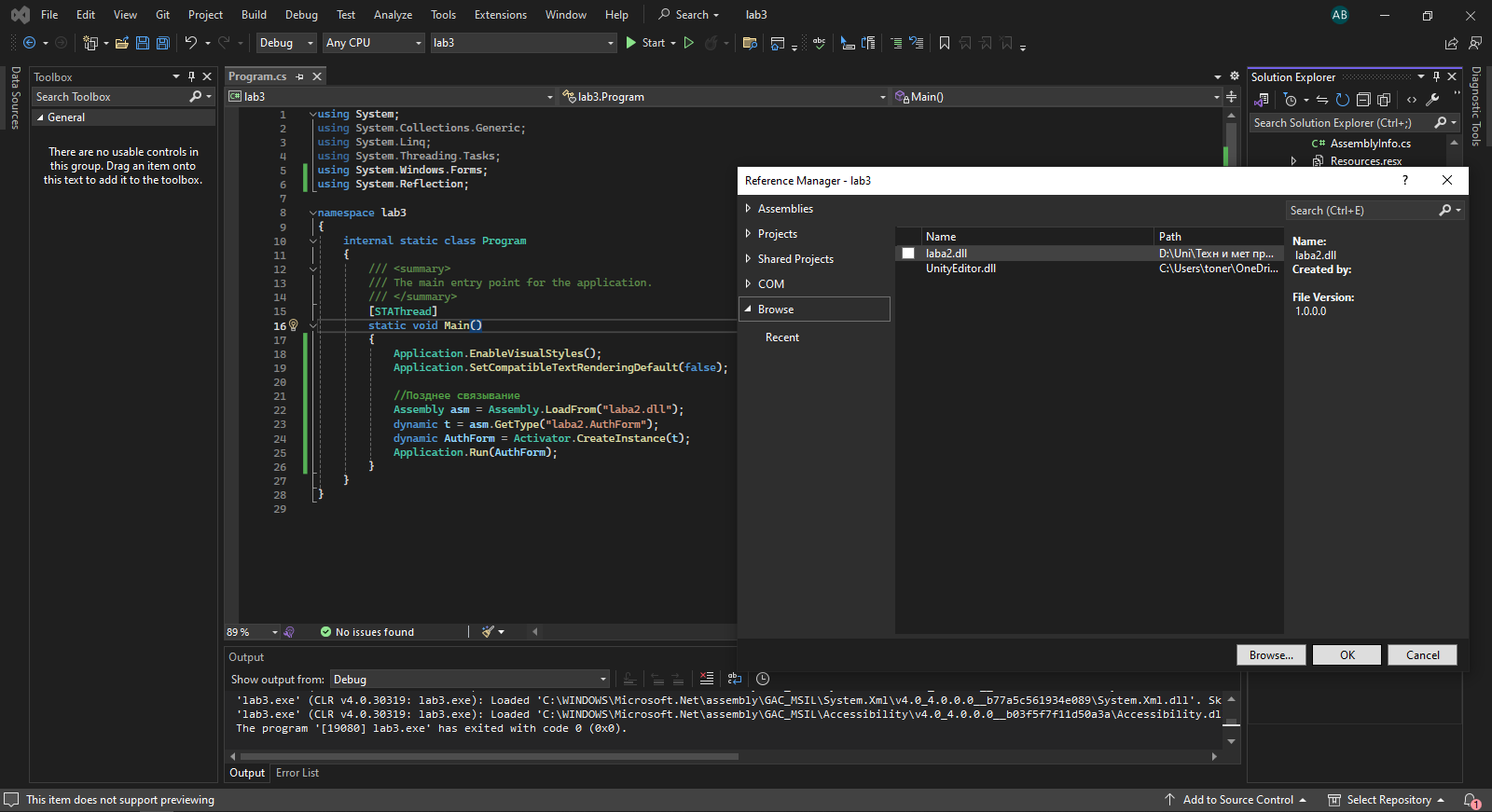
При раннем связывании информация о типах и классах заранее известна, иначе программа просто не будет работать.



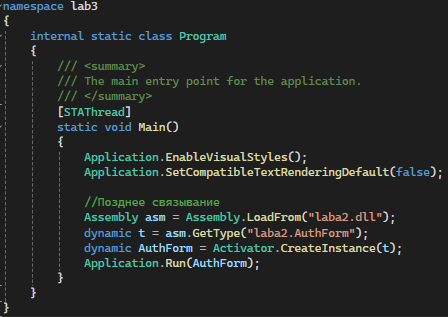
Запускаем программу, все как и планировалось:



Для позднего связывания убираем ссылку на DLL, модифицируем программу для правильной работы:



При позднем связывании мы не знаем о содержимом библиотеки и должны определить необходимый нам класс или метод в ходе работы программы:



Чтобы программа работала корректно, необходимо чтобы динамическая библиотека находилась точно по указанному в программе пути.

