**Лабораторная работа 2 часть 2. Рефлексия**

В данной лабораторной работе нужно разобраться со сторонней библиотекой и добавить ее к проекту Калькулятор. Библиотека находится в файле BinCalc.dll.

При создании приложения для него определяется набор сборок, которые будут использоваться. В проекте указываются ссылки на эти сборки. Когда приложение выполняется, при обращении к функционалу этих сборок они автоматически подгружаются. Кроме того можно динамически подгружать другие сборки, на которые в проекте нет ссылок.

Для управления сборками в пространстве имен System.Reflection есть класс Assembly. С помощью него можно загружать сборку и исследовать ее.

Сначала изучите, что находится в библиотеке BinCalc. Для этого создайте консольный проект в общем решении. Положите файл BinCalc.dll в папку Debug созданного проекта. В методе main создайте объект класса Assembly:

Assembly SampleAssembly = Assembly.LoadFrom("BinCalc.dll");

Метод LoadFrom() принимает в качестве параметра путь к сборке.

Получите все типы в сборке

Type[] types = SampleAssembly.GetTypes();

foreach (Type type in types)

{

Console.WriteLine(type.Name+ "\t"+type.FullName);

}

**Запишите в отчет**, какие типы используются в библиотеке.

Получите массив всех полей

FieldInfo[] ts = types[0].GetFields();

Узнайте и **запишите в отчет** названия всех полей и их типов

Получите массив всех методов

MethodInfo[] ms = types[0].GetMethods();

Узнайте и **запишите в отчет** названия всех методов и значения свойства DeclaringType.

Узнайте информацию о методах класса. Под информацией понимается имя методы, параметры и их тип, тип возвращаемый методом. Например, для метода в списке получите информацию:

MethodInfo method = types[0].GetMethods()[0];

Определите и **запишите в отчет** сигнатуру первого метода и тип им возвращаемый.

Для получения списка параметров используйте метод GetParameters. Для получения названия возвращаемого типа используйте свойство ReturnType.

Вся необходимая информация о методе была получена. Сейчас необходимо вызвать его. Для этого используется метод Invoke:

public object Invoke (object obj, object[] parameters);

Первый параметр представляет объект, для которого вызывается метод, а второй – набор параметров в виде массива object[].

Сначала получим первый параметр obj – объект класса BCalc:

object obj = Activator.CreateInstance(types[0]);

Массив передаваемых параметров – new object[] { "23|4" }

Если бы метод не принимал параметров, то вместо массива объектов использовалось бы значение null: method.Invoke(obj, null)

Соедините все в вызов метода и проверьте его работоспособность:

string s = (string)method.Invoke(obj, new object[] { "23|4" });

Разобрались с тем, как вызвать метод. С тем, что еще может делать этот метод можно познакомиться в документации – файле BinCalc.xml.

Задание.

1. Добавьте полученную библиотеку к своей и реализуйте с помощью нее в классе Calc двоичную арифметику.
2. Доработайте интерфейс приложения для работы с двоичной арифметикой. Добавьте перевод чисел из одной системы счисления в другую.
3. Добавьте документацию для своей библиотеки. Правила составления документации можно посмотреть по ссылке <https://docs.microsoft.com/ru-ru/dotnet/csharp/codedoc>.

Для создания документации нужно добавить XML-комментарии к классу и методам. А затем создать xml-файл. Для создания файла с помощью Visual Studio, щелкните правой кнопкой мыши проект и выберите Свойства. В диалоговом окне свойств откройте вкладку Сборка и выберите XML-файл документации. Запомните, где будет располагаться файл документации. Файл обновляется при каждой сборке проекта.