Лабораторная 10

Сдать до 16.11

Задание 1. Используем черный ящик (4 балла)

У вас есть следующий класс BlackBox:

```
public class BlackBox
{
   private int innerValue;

   private BlackBox(int innerValue)
   {
      this.innerValue = 0;
   }

   private BlackBox()
   {
      this.innerValue = DefaultValue;
   }

   private void Add(int addend)
   {
      this.innerValue += addend;
   }

   private void Subtract(int subtrahend)
   {
      this.innerValue -= subtrahend;
   }

   private void Multiply(int multiplier)
   {
      this.innerValue *= multiplier;
   }

   private void Divide(int divider)
   {
      this.innerValue /= divider;
   }
}
```

Обратите внимание, что все методы помечены модификатором private, поэтому вы не сможете воспользоваться ими из своего кода. Но сможете через рефлексию (reflection, отражение). В классе BlackBox менять что-либо нельзя.

Реализуйте консольное приложение, которое будет считывать операции и аргументы в виде текста и выполнять их над объектом BlackBox, выводя результат вычислений на консоль.

Например:

“Add(100)” => 100

“Subtract(20)” => 80

“Divide(5)” => 16

“Multiply(2)” => 32

Задание 2. Атрибуты для отслеживания важных классов и методов (4 балла)

Реализуйте свой класс для атрибутов CustomAttribute. Ваш атрибут может быть использован как для класса, так и для метода. Он должен принимать параметры: автора, номер ревизии, краткое описание и список ревьюверов (список может быть произвольным, поэтому используйте params string[]).

Далее создайте какой-либо класс с методами и свойствами, для которых задан ваш атрибут Custom. Например, класс HealthScore:

```
[Custom("Joe", 2, "Class to work with health data.", "Arnold", "Bernard")]
public class HealthScore
{
   [Custom("Andrew", 3, "Method to collect health data.", "Sam", "Alex")]
   public static long CalcScoreData()
   {
      ...
   }
}
```

Из главного метода вашего консольного приложения выведите всю полезную информацию, какую только можно получить про ваш класс и каждый из его методов на основе информации из вашего кастомного атрибута (обратите внимание, что не все методы обязательно должны иметь атрибут).

Задание 3. Конверт-матрешка (4 балла)

Вам дан массив конвертов envelopes. Каждый конверт имеет два параметра: ширину (w) и высоту (h). Один конверт может поместиться в другой тогда и только тогда, когда ширина и высота одного конверта меньше ширины и высоты другого конверта.
Вам необходимо вернуть максимальное количество конвертов, которые можно положить один в другой. Вы можете поворачивать конверты.

Пример 1

Вход: конверты = [[5,4],[6,4],[6,7],[2,3]]
Ответ: 3
Пояснение: Максимальное количество вложенных конвертов равно трем: ([2,3] => [5,4] => [6,7]).

Пример 2

Вход: конверты = [[1,1],[1,1],[1,1]]
Ответ: 1

Задание 4*. Продвинутый калькулятор (10 баллов)

Вам необходимо реализовать калькулятор, который может вычислять сложные арифметические выражения, используя определенные функции и параметры.
Поддерживаемые операции: +, -, *, /

Пусть калькулятор поддерживает три «продвинутые» операции. Например:
MyExp(x) – вычисляет экспоненту e^x и заносит информацию об операции в лог
MyLog(x) – вычисляет логарифм log(x) и заносит информацию об операции в лог
MyAbs(x) – вычисляет модуль числа |x| и заносит информацию об операции в лог

Выражения могут быть сколь угодно сложными. Примеры выражений, которые может считать калькулятор:
MyExp(x)* MyExp(x) - MyLog(x)
(MyAbs(x) + MyLog(x)/2) * MyExp(x) - MyLog(x)/ 12 + 12.5
((12-3* MyExp(x) / (MyLog(x) +5)) * (MyAbs(x)/ MyLog(x)) + 0.5* MyExp(x)
В выражениях могут встречаться скобки и арифметические операции. А значение параметра x может меняться.

Требования

- Функции MyExp(x), MyLog(x) и MyAbs(x) определены в отдельной сборке (например, CalcLib.dll)
- Выражение, которое необходимо вычислить, компилируется во время исполнения (runtime) в отдельную временную сборку (например, Temp.dll) и сохраняется во временном файле. В сборке определен класс и метод, вызов которого приведет к вычислению операции. Например,

```
class CalcHolder
{
   public static double MakeCalc(double x)
   {
         // выражение, которое необходимо вычислить
   }
}
```

- Для вычисления результата выражения вам необходимо создать отдельный домен приложения (AppDomain), загрузить в него временную сборку Temp.dll, вызвать метод CalcHolder.MakeCalc с заданным аргументом x и вернуть результат в основное приложение (вывести результат на консоль)

Сценарий использования

Пользователь вводит текстовое выражение, которое он хотел бы посчитать. Параметр x задается отдельно. Текстовое выражение подставляется в текст метода CalcHolder.MakeCalc и класс CalcHolder компилируется в отдельную сборку. Если при компиляции возникают ошибки, пользователь узнает об этом и получает соответствующее сообщение (например, выражение некорректно и пропущена закрывающая скобка). Далее в отдельном app domain происходит вычисление результата, который возвращается пользователю.