Сериализация данных.

Первой строкой на вход участнику подается тип сериализации Json или Xml.
Следующей строкой посылается сериализованный объект.

```
public class Input
    {
        public int K { get; set; }
        public decimal[] Sums { get; set; }
        public int[] Muls { get; set; }
    }
```

Для уобства пример показан в несколько строк. Пример ниже

Xml

```
<Input>
    <K>10</K>
    <Sums>
        <decimal>1.01</decimal>
        <decimal>2.02</decimal>
    </Sums>
    <Muls>
        <int>1</int>
        <int>4</int>
    </Muls>
</Input>
```

Json

```
{"K":10,"Sums":[1.01,2.02],"Muls":[1,4]}
```

На выход участник должен послать сериализованный объект ответа.

```
public class Output
    {
        public decimal SumResult { get; set; }
        public int MulResult { get; set; }
        public decimal[] SortedInputs { get; set; }
    }
```

где SumResult сумма всех чисел из массива Sums входного объекта, умноженная на коэффициент K
MulResult произведение всех чисел из массива Muls входного обекта
SortedInputs отсортированные числа из полей Sums, Muls входного объекта

Для удобства пример показан в несколько строк. Пример ответа ниже

Xml

```
<Output>
    <SumResult>30.30</SumResult>
    <MulResult>4</MulResult>
    <SortedInputs>
        <decimal>1</decimal>
        <decimal>1.01</decimal>
        <decimal>2.02</decimal>
        <decimal>4</decimal>
    </SortedInputs>
</Output>
```

Json

```
{"SumResult":30.30,"MulResult":4,"SortedInputs":[1.0,1.01,2.02,4.0]}
```