Тестовое задание

0. Общие условия

- Приложение должно быть написано на языке **Java 1.6** без использования дополнительных библиотек – только стандартные классы + интерфейсы, относящиеся к заданию.
- Вместе с текстом заданий высылаются интерфейсы – по одному на каждое задание. Они должны быть реализованы в классах с определенными именами, которые, в свою очередь, должны иметь публичные конструкторы без аргументов.
- Исходный код необходимо прокомментировать и упаковать в zip-архив.
- Плюсом будет stateless реализация.

Пример

Задание	Calculator
Название интерфейса	`com.tsystems.javaschool.tasks.Calculator`
Имя класса	`com.tsystems.javaschool.tasks.CalculatorImpl`
Имя архива	`calculator.zip`

Содержимое архива	`com`
	`  + tsystems`
	`  + javaschool`
	`  + tasks`
	`  + Calculator.java`
	`  + CalculatorImpl.java`
	`  + ...`
	`  + ...`

1. Calculator

Написать калькулятор для вычисления простейших арифметических выражений.

Арифметическим выражением считается выражение, включающее:

- Цифры
- Точка в качестве десятичного разделителя
- Круглые скобки
- Знаки операций («+», «-», «*», «/»)

Приоритет операций: скобки, умножение-деление, сумма-вычитание. Округление производится до 4-го знака после запятой, округляется только конечный результат.

В качестве входного параметра в метод передается строка с арифметическим выражением, в результате ожидается строка с вычисленным значением либо `null`, если в выражение не может быть вычислено.

Название интерфейса	`com.tsystems.javaschool.tasks.Calculator`
Имя класса	`com.tsystems.javaschool.tasks.CalculatorImpl`
Имя архива	`calculator.zip`

Пример

```
Calculator c = new CalculatorImpl();

System.out.println(c.evaluate("(1+38)*4-5")); // Результат: 151

c = new CalculatorImpl();
System.out.println(c.evaluate("7*6/2+8")); // Результат: 29

c = new CalculatorImpl();
System.out.println(c.evaluate("-12)1//(")); // Результат: null
```

2. Subsequence

Заданы две последовательности X_1, X_2,..., X_n и Y_1, Y_2,..., Y_k произвольных элементов (`java.lang.Object`). Определить, можно ли получить последовательность X путем вычеркивания некоторых элементов из Y?

В качестве входных параметра в метод передаются два списка: первый – список X_i, второй – список Y_i.

Название интерфейса	`com.tsystems.javaschool.tasks.Subsequence`
Имя класса	`com.tsystems.javaschool.tasks.SubsequenceImpl`
Имя архива	`subsequense.zip`

Пример

```
Subsequence s = new SubsequenceImpl();
boolean b = s.find(Arrays.asList("A", "B", "C", "D"),
Arrays.asList("BD", "A", "ABC", "B", "M", "D", "M", "C", "DC", "D"));
System.out.println(b); // Результат: true
```

3. Duplicates

Составить программу для обработки файла по следующему алгоритму. Задается входной файл, содержащий текстовые строки. Программа обрабатывает его и создает в указанном месте выходной файл, содержащий отсортированные по алфавиту неповторяющиеся строки исходного файла. В конце каждой строки в квадратных скобках указывается количество повторений данной строки во входном файле.

В качестве входных параметра в метод передаются два файла: первый – входной, второй – выходной. Метод возвращает true тогда и только тогда, когда обработка файла прошла успешно. В случае возникновения ошибок программа должна вернуть false.

Не гарантируется, что данные файлы существуют. В случае, если выходной файл не существует, он должен быть создан. Если он существует, необходимо дописать результат выполнения программы, без перезаписи уже содержащейся там информации.

Название интерфейса	`com.tsystems.javaschool.tasks.DuplicateFinder`
Имя класса	`com.tsystems.javaschool.tasks.DuplicateFinderImpl`
Имя архива	`duplicates.zip`

Пример

```
DuplicateFinder d = new DuplicateFinderImpl();
d.process(new File("a.txt"), new File("b.txt"));
```

a.txt

```
ccc
ddd
bbb
ddd
ddd
aaa
```

b.txt

```
aaa[1]
bbb[1]
ccc[1]
ddd[3]
```