Домашнее задание 1. Разминочное.

Срок сдачи: 15 февраля, 2012

Прежде чем приступить к написанию, прочтите раздел **Замечания**.

1 Условие

1. Создайте класс сообщения *Message*. Внутри объект сообщения должен хранить список строк содержания (List<String>).

2. Добавьте в класс *Message* метод *void append(Message)*, принимающий в качестве аргумента другое сообщение и присоединяющий его содержимое к содержимому первого.

3. Добавьте в класс *Message* метод *List<String> getLines()*, предоставляющий доступ к содержимому сообщения.

4. Реализуйте класс *FileMessageReader*, содержащий метод *Message readMessage()*, считывающий сообщения из файла по одному.

   Формат файла с сообщениями следующий:

   кол-во строк первого сообщения
   строки первого сообщения
   кол-во строк второго сообщения
   строки второго сообщения
   ...

5. Создайте класс проверяемого исключения *IllegalMessageFormatException* и используйте его по назначению.

6. Создйте интерфейс *MessageWriter*, содержащий метод *void writeMessage(Message)* и реализуйте трех его наследников: *ConsoleMessageWriter*, *FileMessageWriter* и *CompressingMessageW*

7. *FileMessageWriter* должен выводить сообщения в файл в указанном выше формате.

8. *ConsoleMessageWriter* должен выводить сообщения в консоль в следующем формате:

   "Message 1"
   "1.1. "строка 1
   "1.2. "строка 2
   ...
   "Message 2"
   "2.1. "строка 1
   "2.2. "строка 2
   ...

9. *CompressingMessageWriter* "оборачивает" другой *MessageWriter* и соединяет два последовательных сообщения в одно: 1 с 2, 3 с 4 и т.д. Если последнему сообщению нет пары, значит такова его судьба и нужно просто записать его как есть.

   Напоминание (на всякий случай): метод *append* к этому моменту уже реализован.

10. Класс *Main* интерпретирует первый аргумент командной строки как имя входного файла с сообщениями.

    Если второй аргумент присутствует, то он интерпретируется как имя выходного файла, в противном случае вывод происходит в консоль.

    Нужно всего лишь "сжать" сообщения, находящиеся во входном файле с помощью *CompressingMessageWriter*.

2 Замечания

1. Некоторым вашим классам может понадобиться больше public методов, чем те, которые перечислены в условии.
2. Добавление дополнительных уровней абстракции приветствуется в том случае, если вы готовы обосновать принятые вами решения.
3. Метод *getLines* не должен создавать копии содержимого, а значение, которое он возвращает должен быть неизменяемым. (Подсказка: покопайтесь в классе *Collections*)
4. Для чтения из файла используйте *BufferedReader*, а не *Scanner*
5. Количество сообщений, записанных в файле в нем не указывается.
6. Считывать все сообщения сразу нельзя (представьте, что они не влезают в оперативную память).