Задание

Результат на выходе:

- Репозиторий (можно отправить в архиве) с проектом

Введение:
Представьте, что у вас есть новый пустой проект. Задача этого проекта - конвертировать специфичный формат JSON в HTML, однако со веремнем этот проект дополняется новыми требованиями и вам необходимо их реализовать как архитектору и исполнителю в одном лице.
Ниже будет описана послежовательность требований от заказчика (читай - руководителя, коллеги, etc). Каждое требование с родни оконченной задаче, результат работы над которой будет отправлен в продакшн. Если коммитов будет больше, чем задач, то необходимо удостовериться, что коммит с полностью выполненой очередной задачей будет помечен (например номером задачи, etc). На основе этих коммитов можно будет увидеть ход мысли, принятие решений о том, как должна (или не должна) изменяться архитектура проекта. На работе с git сосредотачиваться не нужно. Сообщения в коммиты можно писать на русском языке - это не принципиально. Тесты ко всему коду можно не писать. необходимо написать только пару тест-кейсов на функции или методы по своему усмотрению только для того, что бы был понятен подход к написанию и организации тестов. Будет плюсом, если вы укажите на места, которые хотели бы проверить перед выпуском в продакшн просто словами - без тестов, что бы не тратить ваше время.
Заметки:

- Программа должна читать файл на диске с названием `source.json` , а вывод работы печатать в `stdout`
- Если в ходе выполнения задачи у вас возникнут проблемы - можете описать какие именно проблемы, как это влияет на бизнес (соовтетсвует/не соовтетствует требованиям), почему не получилось решить задачу и какие есть идеи по ее решению.

Первая задача
Необходимо написать конвертер, которому на вход подается JSON в формате, описанном ниже, а на выходе должен быть HTML для рендеринга в браузере. Формат рассчитан на создание списка параграфов с заголовками.
Пример на входе:

```
[
  {
    "title": "Title #1",
    "body": "Hello, World 1!"
  },
  {
    "title": "Title #2",
    "body": "Hello, World 2!"
  }
]
```

Выход:

```
<h1>Title #1</h1><p>Hello, World 1!</p><h1>Title #2</h1><p>Hello, World 2!</p>
```

Вторая задача

Некоторым проектам требуется более точно указывать какие теги будут использоваться для отображения заголовка и тела, поэтому теперь в ключе будет указано название тега. вместо h3 и div могут быть указаны любые теги. ответсвенность за то, что вместо h3 будет написано table на себя брать нельзя. можно полностью доверять источнику

Пример на входе:

```
[
  {
    "h3": "Title #1",
    "div": "Hello, World 1!"
  }
]
```

Выход:

```
<h3>Title #1</h3><div>Hello, World 1!</div>
```

Третья задача

Потребовались списки. Теперь, если JSON содержит тип list - то все элементы, которые он содержит должны быть обернуты в ul, а каждый конкретный элемент в списке в тег li.

Пример на входе:

```
[
  {
    "h3": "Title #1",
    "div": "Hello, World 1!"
  },
  {
    "h3": "Title #2",
    "div": "Hello, World 2!"
  }
]
```

Выход:

```
<ul><li><h3>Title #1</h3><div>Hello, World 1!</div></li><li><h3>Title #2</h3><div>Hello, World 2!</div></li></ul>
```

Четвертая задача

Подход с рендерингом всем так понравился, что еще несколько проектов решили использовать ваш проект для рендеринга. Однако они предложили усовершенствовать формат. Теперь список может появиться в любом месте а элементы могут быть вложены друг в друга.

Пример на вход 1:

```
[
  {
    "span": "Title #1",
```

```
    "content": [
      {
        "p": "Example 1",
        "header": "header 1"
      }
    ]
  },
  {"div": "div 1"}
]
```

Выход 1:

```
<ul><li><span>Title #1</span><content><ul><li><p>Example 1</p><header>hea
der 1</header></li></ul></content></li><li><div>div 1</div></li></ul>
```

Пример на вход 2:

```
{
  "p":"hello1"
}
```

Выход 2:

```
<p>hello1</p>
```

Пятая задача

Верстка поплыла - необхоимо добавлять класс и идентификаторы к тегам, а содержимое не должно рассматриваться как html

Пример на вход:

```
{
  "p.my-class#my-id": "hello",
  "p.my-class1.my-class2":"example<a>asd</a>"
}
```

Выход:

```
<p id="my-id" class="my-class">hello</p><p class="my-class1 my-class2">ex
ample&lt;a&gt;asd&lt;/a&gt;</p>
```