# Домашнее задание №2

* Deadline:
  + мягкий 24.10, 23:59
  + жёсткий 31.10, 23:59
* Ссылка на контест: <https://contest.yandex.ru/contest/29941> .

Предлагается 2 задачи на выбор. 2я задача более сложная и поэтому оценивается в 2 раза выше.

## Технические подробности

Для сдачи задания в контест требуется загрузить не код, а собранный Jar-файл. В этом Jar должен также иметься файл MANIFEST.MF, в котором должен быть указан класс с методом main(). Например:

Manifest-Version: 1.0

Main-Class: org.atp.lesson1.Main

Для подготовки Maven-проекта используйте [данный шаблон](https://github.com/atp-mipt/homework-quickstart). Он позволит частично автоматизировать code review и избежать основных ошибок.

## Задача. Русские шашки.

Дана композиция начала или середины партии [русских шашек](http://www.gambiter.ru/checkers/item/5-russkie-shashki-pravila.html). Далее даётся список из 1 или нескольких ходов. На выходе нужно вывести итоговую композицию или сообщение об ошибке если какой-то из ходов списка невозможен.

В данной реализации шашки могут не только бить, но и ***ходить назад***.

Типы сообщений об ошибке:

* **busy cell** - целевая клетка занята
* **white cell** - целевая клетка белая (шашки расставляются только на чёрные и, в силу правил, оказаться на белых не могут),
* **invalid move** - в шашках бить обязательно. Причём, бить надо всю цепочку до конца. Ошибка выводится в том случае если у игрока есть вариант побить шашку, но он его не использует, а идёт на другую клетку. Если вариантов боя несколько, можно взять любой.
* **error** - другие ошибки.

Все исходные позиции - валидные.

### Требования к коду

* При написании кода для ошибок нужно создавать свои классы исключений. Исключения отлавливаются на самом высшем уровне (метод main()) и выводится соответствующий error message. Ловля checked-исключений “по месту” (e.printStackTrace) запрещена.
* Нужно реализовать JavaDoc для нетривиальных публичных методов и классов.
* Также нужно сделать хотя бы 5 unit-тестов.
* (Можно создать Checkstyle Rule) запрещено использовать устаревшие API в Java, заменённые в новых версиях. Прежде всего это касается File и FileInputStream (эти классы чаще всего продолжают использовать новички, хотя есть Java NIO API), также: Date, Calendar, DateFormat, StringBuffer, Vector, HashTable.
* Парсинг входных данных осуществлять с помощью регулярных выражений, конечных автоматов или чего-то подобного. “Наивный” парсинг с помощью split, indexOf, replace не приветствуется.

### Формат входных данных

Координаты шашек на доске и ходы записываются в [стандартной шашечной нотации](http://www.c3b4.narod.ru/alphabet/notation.html). Характеристики ходов (“!” - хороший ход, “?” - плохой ход и т.д.) в записи не применяются. Если шашка является дамкой, то её координата обозначается с большой буквы (например, не d4, а D4). Если за 1 ход происходит несколько взятий, то они записываются через “:” так: e5:c3:a1 .

1. Строка с координатами белых шашек
2. Строка с координатами чёрных шашек
3. Список ходов. Пара ходов (белые + черные) в строке.

### Формат выходных данных

1. Строка с координатами белых шашек
2. Строка с координатами чёрных шашек

Либо сообщение об ошибке

### Примеры входных данных

| in | out |
| --- | --- |
| a1 a3 b2 c1 c3 d2 e1 e3 f2 g1 g3 h2  a7 b6 b8 c7 d6 d8 e7 f6 f8 g7 h6 h8  g3-f4 f6-e5  c3-d4 e5:c3  b2:d4 d6-c5  d2-c3 g7-f6  h2-g3 h8-g7  c1-b2 f6-g5  g3-h4 g7-f6  f4-e5 f8-g7 | a1 a3 b2 c3 d4 e1 e3 e5 f2 g1 h4  a7 b6 b8 c5 c7 d8 e7 f6 g5 g7 h6 |

## Задача. Столбовые русские шашки (“Башни”). Бонус +100%

Постановка задачи такая же как и для обычных шашек. При разработке можно брать за основу код предыдущей задачи. Отличия в правилах игры:

1. *Побитая шашка соперника не снимается с доски, а забирается* ***под*** *бьющую башню. Таким образом, шашка имеет несколько “жизней”.*
2. *Если бьют башню, то с нее снимается только одна верхняя шашка после чего шашка (или дамка), находившаяся под ней вступает в права.*
3. *Если бьются несколько шашек соперника, то они забираются одна за другой под бьющую башню последовательно в процессе боя и на конечном поле образуется “башня”.*
4. *Башня, как и одиночная шашка, может проходить в дамки, причем дамкой становится только верхняя шашка.*
5. *В одном ходе нельзя несколько раз бить одну и ту же “башню”.*

### Формат входных данных

1. Строка с координатами чёрных шашек (и башен с черной крышкой)
2. Строка с координатами белых шашек (и башен с белой крышкой)
3. Список ходов. По паре ходов (черные + белые) в строке.

Координаты “башен” записываются согласно следующему принципу: xn\_aaaaa, где:

* x - буквенная координата. Например, d или D если башня в данный момент играет роль дамки (дамка наверху).
* n - численная.
* aaaa - содержание башни. Шашки в башне нумеруются сверху вниз и помечаются буквами:

| b | чёрная обычная |
| --- | --- |
| w | белая обычная |
| B | чёрная дамка |
| W | белая дамка |

Примеры нотации

|  |  |  |
| --- | --- | --- |
| b8\_bbBbw | d4\_bbBBBb | h6\_wWwWwWbb |

### Формат выходных данных

1. Строка с координатами белых шашек (и башен с белой крышкой)
2. Строка с координатами чёрных шашек (и башен с черной крышкой)

### Примеры входных данных

| in | out |
| --- | --- |
| a1\_w c1\_w e1\_w f2\_ww h2\_w g5\_wbb  a3\_b e3\_b a5\_bww c5\_bwww e7\_b g7\_b b8\_b d8\_b f8\_b h8\_b  f2\_ww:d4\_wwb:b6\_wwbb g7\_b-f6\_b  h2\_w-g3\_w f6\_b:h4\_bw:f2\_bww  e1\_w:g3\_wb g5\_bb-h4\_bb | invalid move |
| a7\_wbb b2\_ww c1\_w e1\_w f2\_w g1\_w  b4\_bwww b8\_b c3\_b c7\_b e5\_bww e7\_b f8\_b g5\_b g7\_b h8\_b  b2\_ww:d4\_wwb:f6\_wwbb:d8\_wwbbb:b6\_wwbbbb b4\_bwww-a3\_bwww | a7\_wbb b6\_Wwbbbb c1\_w e1\_w e5\_ww f2\_w g1\_w  a3\_bwww b8\_b f8\_b g5\_b g7\_b h8\_b |