Day 2: Rock Paper Scissors

Aufgabe und Input

- Aufgabe
- Input

(d2e - Beispieleingabe aus dem Aufgabentext, d2i Eingabe für die Lösungen auf dieser Wikiseite)

Ergebnisse

Ergebnis Teil 1 für die Eingabe auf dieser Wikiseite (weicht ab von deiner "echten" Lösung)

11767

Ergebnis Teil 2 für die Eingabe auf dieser Wikiseite (weicht ab von deiner "echten" Lösung)

13886

Eine Lösung mit HashMaps

Man kann beide Teile lösen indem man die Bedingungen des Spiels als assoziatives Array hinterlegt und die Transformationen von Zügen zu Punkten dann einfach aus dem assoziativen Array ausliest.

Informationen zu assoziativen Arrays findest du auf dieser Wiki-Seite. In Java heißt diese Datenstruktur HashMap.

Nun kann man die Spielmechanik für Teil 1 in einer HashMap ablegen und anschließende die Punkte direkt über den Key, den man aus dem Input erhält bestimmen:

```
HashMap<String,Integer> roundScore = new HashMap<String,Integer>();


public day2() throws Exception {
    this.readInput(inputFile,' ');
    this.printInput();

    roundScore.put("AX", 4);
    roundScore.put(       );
    roundScore.put(       );
    roundScore.put(       );
    roundScore.put(       );
    roundScore.put(       );
    roundScore.put(       );
    roundScore.put(       );
    roundScore.put(       );
```

Tipp Teil 1: Codegerüst

```
[...]
    HashMap<String,Integer> roundScore = new HashMap<String,Integer>();

    public day2() throws Exception {
        this.readInput(inputFile,' ');
        this.printInput();

        // Punkte eintragen gemäß Aufgabenstellung
        roundScore.put("AX", 4);
        roundScore.put("AY", );
        roundScore.put("AZ", );
        roundScore.put("BX", );
        roundScore.put("BY", );
        roundScore.put("BZ", );
        roundScore.put("CX", );
        roundScore.put("CY", );
        roundScore.put("CZ", );
    }

    public int partOne() {
        int score=0;
        for(String[] line: input) {
            // Die Zeilen werden am Leerzeichen getrennt eingelesen
            String game=line[0] + line[1];
            System.out.println(game);
            score += // Wie kann man nun den Score aus der HashMap lesen?
            System.out.println(score);
        }

        return score;
```

```
    }
```

Tipp Teil 2. "Doppelte HashMap"

Im Prinzip ändert sich in Teil 2 nichts, man benötigt lediglich eine zweite HashMap, die abbildet, welcher Zug ausgeführt werden muss:

```
HashMap<String,Integer> roundScore = new HashMap<String,Integer>();
HashMap<String,String> rightMove = new HashMap<String,String>();

public day2() throws Exception {
    this.readInput(inputFile,' ');
    this.printInput();

    roundScore.put("AX", 4);
    roundScore.put(       );
    roundScore.put(       );
    roundScore.put(       );
    roundScore.put(       );
    roundScore.put(       );
    roundScore.put(       );
    roundScore.put(       );
    roundScore.put(       );

    rightMove.put("AX","AZ"); // Rock + verlieren -> Rock/Scissors AZ
    rightMove.put("AY","AX"); // Rock + unentschieden ->
    rightMove.put(         ); // Rock + gewinnen ->
    rightMove.put(         ); // Paper + verlieren ->
    rightMove.put(         ); // Paper + unentschieden ->
    rightMove.put(         ); // Paper + gewinnen ->
    rightMove.put(         ); // Scissors + verlieren ->
    rightMove.put(         ); // Scissors + unentschieden ->
    rightMove.put(         ); // Scissors + gewinnen ->
```

From:
https://info-bw.de/ -

Permanent link:
https://info-bw.de/faecher:informatik:oberstufe:java:aoc:aoc2022:day2:start?rev=1670270529

Last update: **05.12.2022 20:02**