A1

a)

Parameter	Äquivalenzklasse	Gültig?	Repräsentanten
LaufNr	1	G	1
LaufNr	2	G	2
LaufNr	3,4	G	3
LaufNr	5,6	G	5
LaufNr	< 1	U	0
LaufNr	> 6	U	7
tageBisZumWettkampf	> 32	G	33
tageBisZumWettkampf	32 - 7	G	32 / 7
tageBisZumWettkampf	6 - 4	G	6 / 4
tageBisZumWettkampf	< 4	U	3

b)

Gültig?	laufNr	tageBisZumWettkampf	Ergebnis
G	1	33	20
G	1	32	25
G	1	6	28
U	1	3	-
G	2	33	15
G	2	32	20
G	2	6	23
U	2	3	-
G	3	33	10
G	3	32	15
G	3	6	18
U	3	3	-
G	5	33	5
G	5	32	5
G	5	6	8
U	5	3	-
U	0	33	-
U	7	33	-

A2

```java
public int berechneGebuehr(int laufNr, int tageBisZumWettkampf) {
    if(laufNr < 1 || laufNr > 6 || tageBisZumWettkampf < 4) return -1;
    int gebuehr = switch (laufNr) {
       case 1 -> tageBisZumWettkampf > 32 ? 20 : 25;
       case 2 -> tageBisZumWettkampf > 32 ? 15 : 20;
       case 3, 4 -> tageBisZumWettkampf > 32 ? 10 : 15;
       case 5, 6 -> 5;
       default -> -2; // Sollte nicht erreicht werden
    };
    if(tageBisZumWettkampf <= 6) gebuehr += 3;
    return gebuehr;
}
```

```java
@ParameterizedTest(name = "laufNr: {0}, Anmeldung {1} Tage vor Wettkampf kostet {2}€")
@CsvSource({
       "1, 33, 20",
       "1, 32, 25",
       "1,  6, 28",
       "1,  3, -1",
       "2, 33, 15",
       "2, 32, 20",
       "2,  6, 23",
       "2,  3, -1",
       "3, 33, 10",
       "3, 32, 15",
       "3,  6, 18",
       "3,  3, -1",
       "5, 33,  5",
       "5, 32,  5",
       "5,  6,  8",
       "5,  3, -1",
       "0, 33, -1",
       "7, 33, -1"
})
void berechneGebuehrTest(int laufNr, int tageBisWettkampf, int expectedResult) {
    Calculator calculator = new Calculator();
    assertEquals(expectedResult, calculator.berechneGebuehr(laufNr, tageBisWettkampf),
          "Laufnummer " + laufNr + " mit Anmeldung " + tageBisWettkampf +
                " vor dem Wettkampf sollte " + expectedResult + " kosten");
}
```