```java
public static double calculateOverallRating(Movie m) {
    List<Performer> performers = storage.getLinkedPerformersOfMovie(m);
    double movieRating = m.rating();
    double result = m.rating();
    if(!performers.isEmpty() && m.isRated()) {
        DoubleSummaryStatistics stats = new DoubleSummaryStatistics();
        for (Performer p : performers) {
            if (p.isRated()) {
                double rating = p.rating();
                stats.accept(rating);
            }
        }
        if(stats.getCount() > 0) {
            double performersRating = stats.getAverage();
            result = (movieRating + performersRating) / 2.0;
        }
    }
    return result;
}
```

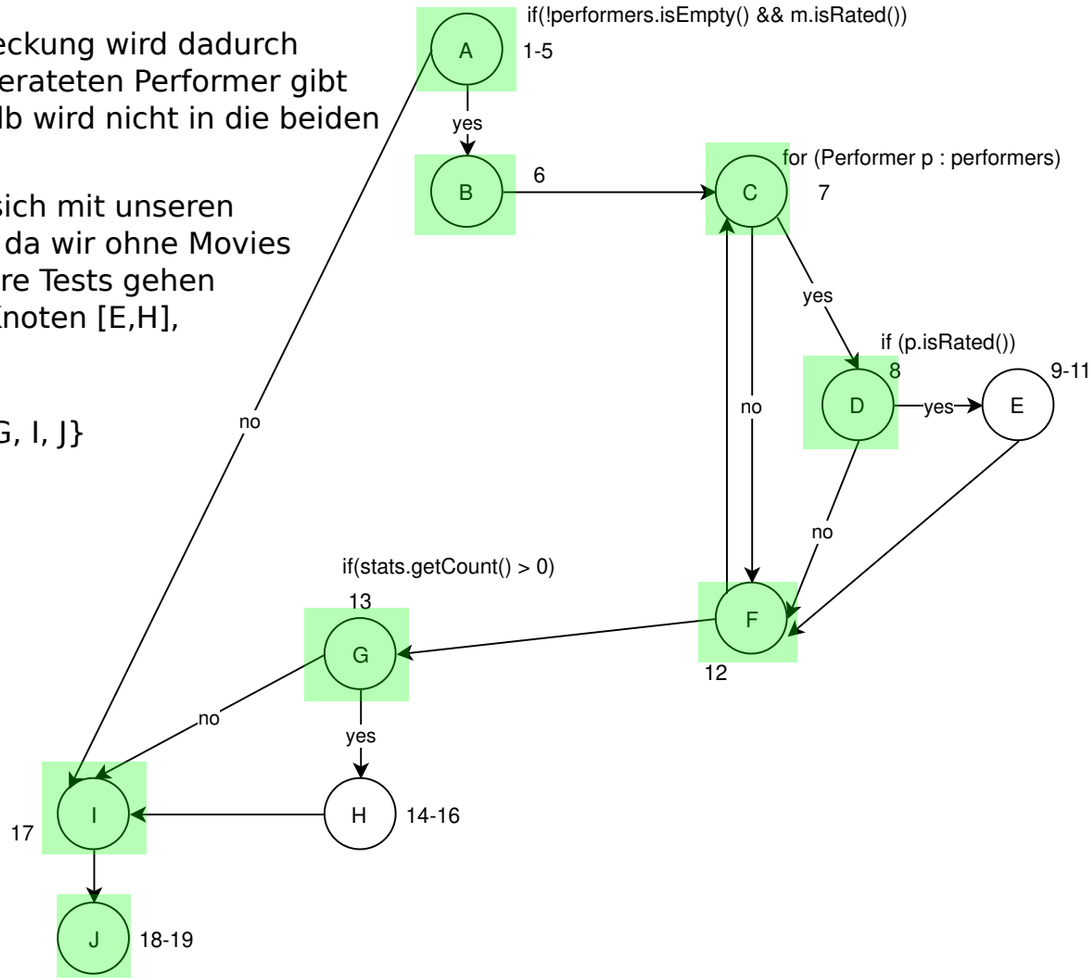
A B C D E F G H I J

9.2

Die Anweisungsüberdeckung wird dadurch erreicht, da es keine gerateten Performer gibt in diesen Tests. Deshalb wird nicht in die beiden if scopes gegangen.

Die Menge KB ändert sich mit unseren gewählten Tests nicht, da wir ohne Movies Testen. Das heißt unsere Tests gehen ebenfalls nicht in die Knoten [E,H],

KB= {A, B, C, D, F, G, I, J}



1.3

100% Anweisungsüberdeckung:
A->B: Performers existieren, Movie is rated
B->C: X
C->D: Multiple Performers exist
D->E: Performer is rated
E->F: X
F->G: Performer loop ended
G->H: Rated Performer exist
H->I: X
I->J: X
Für eine 100% Anweisungsüberdeckung wird ein Test benötigt mit mindestens einem rated Movie und mindestens einem rated Performer.
100% Zweigüberdeckung
A->I: No performers or no rated movie
D->F: Performers are not rated
G->I: Performers are not rated
Für eome 100% Zweigüberdeckung wird der Test für 100% Anweisungsüberdeckung benötigt, sowie:
Ein Test mit keinen Performern oder keine rated Movies.
Ein Test mit rated Movies und unrated Performers