Software Engineering Hausaufgabe 10

Youran Wang (719511, RAS), Yannick Fuchs (723866, ITS)

Januar 2022

1 Git

Kleine Anmerkung an dieser Stelle. Ich (Yannick) hatte bereits einen Git-Account, weshalb nach dem initial commit plötzlich ein anderer Account aufgelistet wird und mein Name nur ein einziges Mal vorkommt.

2 Testen mit JUnit

2.1

2.1.1

1,1,1
Hier testen wir ob das Programm gleichseitige Dreiecke erkennt.

2.1.2

1,2,1
Hier wird geprüft ob gleichschenklige Dreiecke erkannt werden.

2.1.3

1,2,3
Hier wird geschaut ob ungleichseitige Dreiecke erkannt werden.

2.1.4

-1,2,1
Hier wollen wir testen ob das Programm mit negativen Werten umgehen kann.

2.2

Siehe zip-Archiv, genauer gesagt die Dateien: TriangleType (hinzufügen eines weiteren Typen), Triangle (Funktion getType wurde um einen Fall erweitert) und TriangleTest.

Anmerkungen:
1.) Da unser Programm beim letzten Mal nicht ausführbar war, haben wir an dieser Stelle mit der Musterlösung der letzten Aufgabe weiter gearbeitet, welche im Moodle zur Verfügung stand.

2.) Leider lassen sich die Tests nicht mittels *mvn test* ausführen, da die Main-Klasse, sowie einige Bestandteile von Javafx nicht erkannt werden. Jedoch laufen die Testfälle fehlerfrei durch, wenn man die Klasse TriangleTest innerhalb der IDE Intellij ausführen lässt.

3

3.1

Diese Variante von LTL ist daher nicht eingeschränkt, da wir uns jene Operationen, welche hier fehlen aus den vorhandenen ableiten können.

Zu Beginn können wir $\varphi_1 \wedge \varphi_2$ darstellen als $\neg(\neg\varphi_1 \vee \neg\varphi_2)$. Daraus folgen die nächsten Operationen: $\varphi_1 \rightarrow \varphi_2$ ist darstellbar als $\neg\varphi_1 \vee \varphi_2$, daraus können wir ableiten, dass $\varphi_1 \Leftrightarrow \varphi_2$ zu $\neg(\neg(\neg\varphi_1 \vee \varphi_2) \vee \neg(\neg\varphi_2 \vee \varphi_1))$.

Wenn wir nun die temporallogischen Operationen betrachten, so lassen auch diese sich aus den vorhandenen formen. $\mathcal{F}\varphi_1$ ist nämlich equivalent zu $\neg\varphi_1\mathcal{U}\varphi_1$. Wir können $\mathcal{G}\varphi_1$ als $\varphi_1\mathcal{U}\neg\varphi_1$ und $\varphi_1\mathcal{R}\varphi_2$ als $\varphi_2\mathcal{U}(\neg(\neg\varphi_1 \vee \neg\varphi_2))$ darstellen.

3.2

Auch hier lassen sich die aussagenlogischen Operationen äquivalent zu der vorherigen Teilaufgabe herleiten. Jedoch wird bei $\varphi_1\mathcal{U}^+\varphi_2$ vorausgesetzt, dass φ_2 nicht auf ω_0 gesetzt werden kann. Daraus können wir recht leicht unser $\mathcal{F}\varphi_1$ definieren: $\neg\varphi_1\mathcal{U}^+\varphi_1$.
Es fehlen nun also noch unser $\mathcal{G}$, $\mathcal{R}$ und unser $\mathcal{X}$. Die ersten beiden Operationen können wir ebenfalls analog zur vorherigen Teilaufgabe definieren, aber $\mathcal{X}$ müssen wir hier neu definieren. Dies machen wir wie folgt: $(\neg\varphi_1\mathcal{U}^+\varphi_1)\mathcal{U}^+\neg\varphi_1$.

3.3

Alle der genannten Operationen dürfen sich ja nicht auf die aktuelle Position beziehen und das erreichen wir dadurch, dass wir zur Definition von $\mathcal{G}^+$ und $\mathcal{F}^+$, $\mathcal{U}^+$ verwenden. Somit beziehen wir uns sowieso nicht auf die aktuelle Position. $\mathcal{F}^+$ wäre dann also als $\neg\varphi_1\mathcal{U}^+\varphi_1$ und $\mathcal{G}^+$ als $\varphi_1\mathcal{U}^+\neg\varphi_1$ definiert.