

## Aufgabe 1 – Regressionstest

-> Was versteht man unter einem Regressionstest?

Der Regressionstest beschreibt einfach gesagt ein Verfahren, bei dem Programme nach Änderungen erneut getestet werden, um sicherzustellen dass auch bei kleinsten Programmänderungen keine Rückschritte aufgetreten sind.

-> Wozu ein Regressionstest?

Der Regressionstest ist vor allem dann hilfreich, wenn Software gewartet oder überarbeitet wird, da diese Vorkommen schnell zu Fehlern führen können. Zudem liefert der Test einen Nachweis zur Programmkorrektheit, wenn die Software nachträglich modifiziert wird. Bleibt die Funktionalität in der Software nach Modifikation erhalten, ist der Regressionstest erfolgreich gewesen.

-> Probleme von Regressionstests

Der Test kann nicht zu voller Korrektheit funktionieren, da die Testergebnisse von der verwendeten Hardware abhängen und dadurch unterschiedliche Ergebnisse erzielt werden können. Zudem gibt es einige Szenarien, in denen automatisch nicht getestet werden kann – beispielsweise ein Flugüberwachungssystem. Das Zeitverhalten eines Menschen kann so nicht „Regressionsgerecht“ erfolgen.

-> Welche Voraussetzung muss erfüllt sein, um einen Regressionstest durchführen zu können?

Die einzige Voraussetzung die für einen Regressionstest von größter Wichtigkeit ist, ist die Grundfunktionalität des Programms. Das heißt, dass der Regressionstest erst dann Anwendung finden kann, wenn ein Programm vorliegt, welches vollständig und ohne Probleme die erwünschte Funktionalität erzielt. Der Regressionstest bezieht sich damit logischerweise auf Modifikationen des Programms.

-> Wie können Testwerkzeuge unterstützend eingesetzt werden?

Testwerkzeuge sind vor allem dann hilfreich, wenn ein so großes Programm vorliegt, dass der Aufwand des Tests schnell den der Änderung von Programmzeilen übersteigt. Die Lösung heißt Automatisierung. Zur Verfügung stehen uns hier die Automatisierte **Testdurchführung** und **Auswertung der Testergebnisse** um den Entwickler/Tester zu entlasten.

## Aufgabe 2 – Testmodelle

-> Wie ordnet sich der Regressionstest in die 2 in der VL vorgestellten Modelle ein?

In der Vorlesung wurden das *Aktivitäten-Modell* nach Wegener sowie das *V-Modell* erläutert. Der Regressionstest gliedert sich im Aktivitäten-Modell in die Phasen „*Test-Data Selection*“, „*Test Execution*“ und „*Test Evaluation*“ ein. Im V-Modell wiederum gliedert sich der Regressionstest in die Phase „*Unit Test*“ auf der rechten Seite des V ein.

-> Wie werden die Testfälle und die Testdaten ermittelt?

Bei der Testfallermittlung bedarf es zunächst einer Ermittlung der zu testenden Objekte und einer Bewertung der Testpriorität, die anhand manuell bereits erfolgreich ausgeführter Tests erfolgen. Testfälle, für die eine Automatisierung erstellt werden soll, werden also anhand verschiedener Kriterien aus bereits erstellten manuellen Testfällen ermittelt und näher beschrieben. Beim Regressionstest werden Testdaten entweder zufällig oder per Aufzeichnung in einer Benutzersitzung ermittelt.

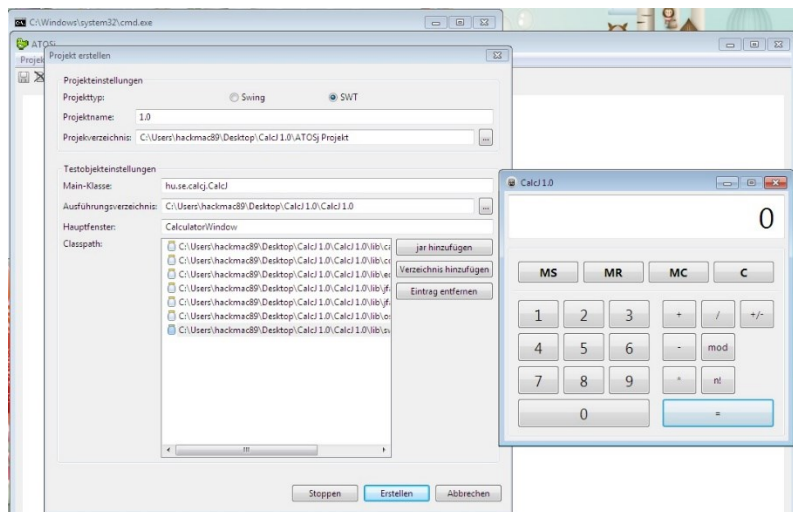
## Aufgabe 3 – ATOSj/CalcJ

-> Projekt einrichten

Durch starten der BAT und klicken auf „*Projekt erstellen*“ kommt man in das Menü, deren Daten ähnlich wie in der Abbildung im Aufgabenblatt „*V5.pdf*“ zu befüllen sind.

*ATOSj*

Projekt erstellen  
Projekt öffnen



(Abbildung 1: ATOSj Hauptmenü)



(Abbildung 2: ATOSj Projekt erstellen)

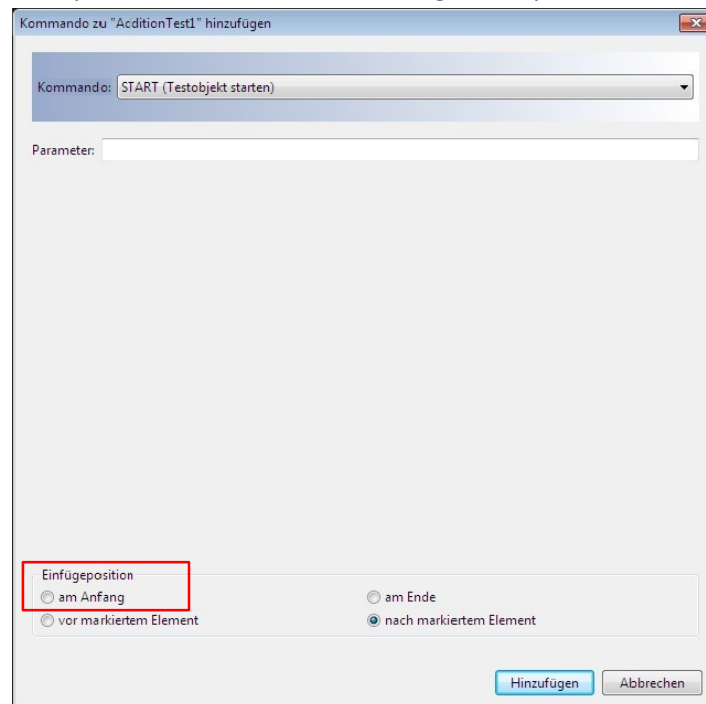
-> Testsequenz aufzeichnen

Im nun erstellten APF-Projekt rechtsklickt man im linken Reiter auf „Testsequenzen“, anschließend wählt man „Testsequenz erstellen“



(Abbildung 3: Neue Testsequenz „AdditionTest1“ erstellen)

Ist die Testsequenz erzeugt, so startet man das Aufzeichnen einer Testsequenz mit einem Klick auf das -Zeichen am rechten Bildrand. Als Beispiel-Testfall kann man nun das aus „V5.pdf“ bekannte  $-3 + -2 = -5$  verwenden, indem man die Rechnung in den Taschenrechner eingibt und anschließend das Programm wieder beendet. Um ein Testszenario starten zu können benötigt eine Testsequenz noch das START Kommando. Dieses fügt man (und das hätte man auch vor dem Aufzeichnen einer Testsequenz bereits tun können) über einen Klick auf das -Zeichen am rechten Bildrand. Um ein START-Kommando an den Anfang einer Testsequenz zu setzen, wählt man folgende Optionen.



(Abbildung 4: START Kommando einfügen)

Nummer	Kommando
1	START
2	ACTION, MAIN, BUTTON, "=", SELECT
3	ACTION, MAIN, BUTTON, "3", SELECT
4	ACTION, MAIN, BUTTON, "+/-", SELECT
5	ACTION, MAIN, BUTTON, "+", SELECT
6	ACTION, MAIN, BUTTON, "2", SELECT
7	ACTION, MAIN, BUTTON, "+/-", SELECT

(Abbildung 5: START Kommando wurde am Anfang der Sequenz eingefügt)

Nun gilt es noch, die Testsequenz zu starten. Dies erfolgt durch einen Rechtsklick auf die zu startende Testsequenz gefolgt von der Auswahl des Menüpunktes „starten“. Im sich öffnenden Fenster kann man noch zusätzliche Optionen wie eine zeitliche Verzögerung oder die Protokollierung einstellen. Durch einen Klick auf „Starten“ beginnt die Testsequenz und man erhält wie in der folgenden Abbildung ersichtlich farblich gekennzeichnete Ausgaben der verschiedenen Schritte der Sequenz.

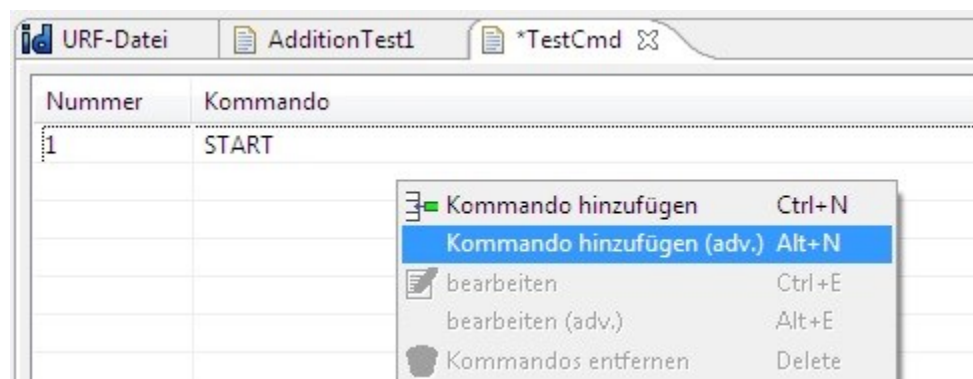
Nummer	Kommando	Status	Durchläufe	Dauer
1	START	Erfolg	1	5520
2	ACTION, MAIN, BUTTON, "=", SELECT	Erfolg	1	659
3	ACTION, MAIN, BUTTON, "3", SELECT	Erfolg	1	514
4	ACTION, MAIN, BUTTON, "+/-", SELECT	Erfolg	1	531
5	ACTION, MAIN, BUTTON, "+", SELECT	Erfolg	1	520
6	ACTION, MAIN, BUTTON, "2", SELECT	Erfolg	1	537
7	ACTION, MAIN, BUTTON, "+/-", SELECT	Erfolg	1	522

(Abbildung 6: Ein Durchlauf einer Testsequenz)

-> TEST-Kommando einfügen

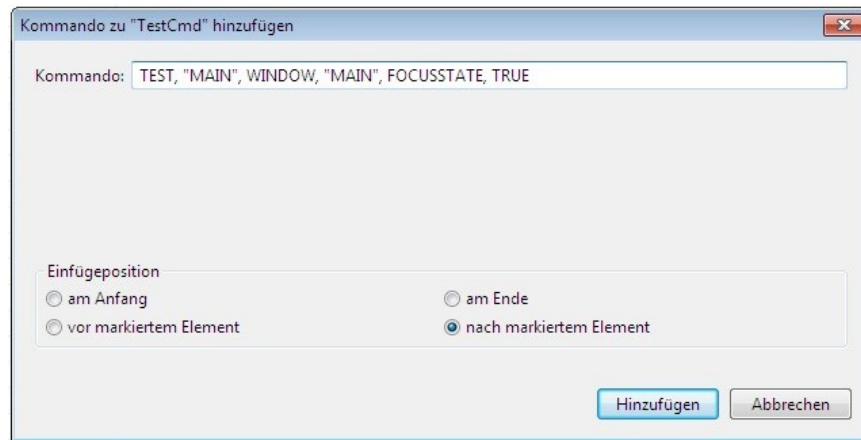
Ein Setzen eines TEST-Kommandos erfolgt analog zu dem oben beschriebenen Vorgang zum Setzen eines START-Kommandos, nur das beim TEST-Kommando mehrere Parameter (Variable, Sollwert, Istwert...) anzugeben sind.

Ein manuelles Setzen erfolgt mit einem Rechtsklick innerhalb des Listenfelds einer Testsequenz und der Auswahl des Menüeintrages „Kommando hinzufügen (adv.)“ (siehe Abbildung 7).



(Abbildung 7: Ein Kommando manuell einfügen)

Kommandos unterliegen einer Syntax, welche versch. Parameter erwartet. Ein Beispiel ist in Abbildung 8 zu sehen.



(Abbildung 8: Ein Beispiel für ein manuelles Kommando)

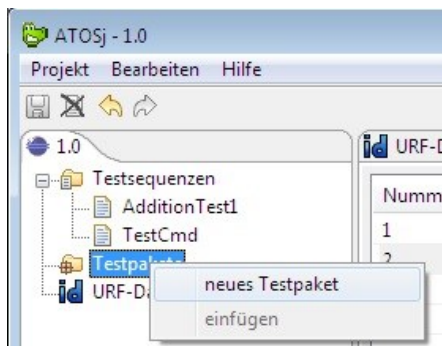
Das obige (etwas sinnfreie) TEST-Kommando testet zum Beispiel, ob das Programmfenster nach dem starten der Testsequenz den Fokus erhalten hat.

Die Eingabe wird folgendermaßen erwartet:

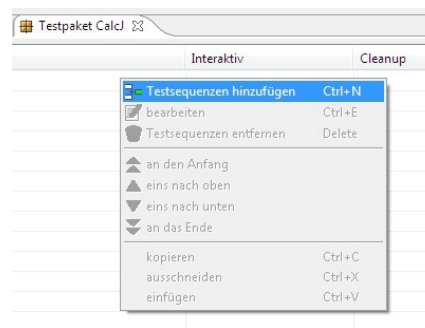
`<KOMMANDOART>, „KLASSE“, OBJEKT, „KLASSE“, <EVENT>, <WERT>`

-> Zwei oder mehr Testsequenzen zu Testpaket bündeln

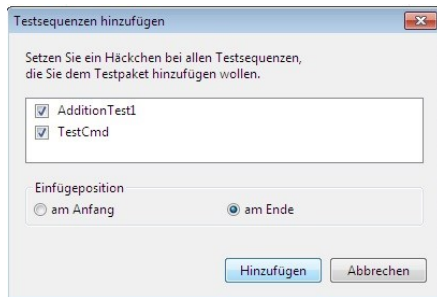
Hat man mehrere Testsequenzen und möchte diese in einem Paket zusammenfassen, so genügt ein Rechtsklick auf „Testpakete“ im linken Reiter, gefolgt von der Auswahl „neues Testpaket“. Im Listenfeld des eben erstellten Paketes muss nach einem Rechtsklick der Menüpunkt „Testsequenzen hinzufügen“ gewählt werden. Im sich öffnenden Fenster kann man anhand von Checkboxes die zu bündelnden Testsequenzen anwählen. Mit einem Testpaket ist es nun möglich, gleich mehrere Testfälle sequenziell abzuarbeiten. Die folgenden Abbildungen verdeutlichen den Ablauf.



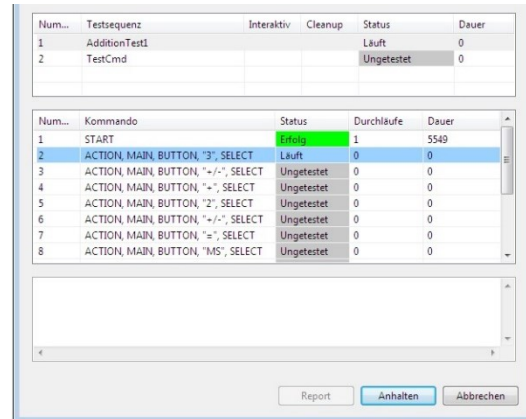
(Abbildung 9: Neues Testpaket erstellen)



(Abbildung 10: Hinzufügen der Testsequenzen)



(Abbildung 11: Hinzufügen der Testsequenzen)



(Abbildung 12: Sequenzielles Testen mehrerer Sequenzen)

#### Aufgabe 4 – Testfälle ATOSj/CalcJ

-> Überlegen Sie sich für das Beispiel Addition mit dem Taschenrechner CalcJ 1.0 sinnvolle Testfälle und tragen Sie diese in eine Tabelle ein

Testfall	Testdaten	Erwartetes Ergebnis
$x < y < 0$	$x = -3, y = -2$	-5
$x > y > 0$	$x = 2, y = 1$	3
$y < x < 0$	$x = -2, y = -3$	-5
$x < 0 < y$	$x = -5, y = 8$	3
$ x  <  y , x < 0 < y$	$x = -5, y = 8$	3 → ( $x + y > 0$ )
$x > y = 0$	$x = 84, y = 0$	84
$x = y < 0$	$x = -10, y = -10$	-20
$x = y > 0$	$x = 11, y = 11$	22