Übungsblatt 2

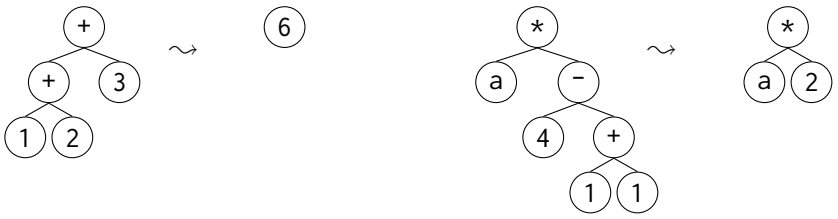
Projektübung 2 (Abgabe bis 26.11.2017)

Im Rahmen dieser Projektübung implementieren Sie eine Konstantenfaltung auf ASTs, die (ganzzahlige) Teilausdrücke erkennt und vereinfacht, deren Ergebnis statisch im Übersetzer bestimmt werden können.

a) Machen Sie sich zunächst damit vertraut, wie Konstanten im AST repräsentiert werden und wie Sie bestimmen, ob es sich bei diesen Konstanten um ganzzahlige Werte handelt.

Hinweis: Zur Repräsentation des *Wertes* einer ganzzahligen Konstante verwenden wir die Klasse `IntValue`. Diese stellt sicher, dass der Wert in der korrekten Größe abgespeichert wird, und bietet Methoden für Berechnungen an, die das in der Sprachspezifikation geforderte Überlaufverhalten berücksichtigen.

b) Die e2-Sprachspezifikation beschreibt in Abschnitt 5.1, welche arithmetischen Ausdrücke die Konstantenfaltung erkennen und vereinfachen soll. Implementieren Sie einen Besucher `ConstantFolding`, der den AST abläuft und die entsprechenden Transformationen durchführt. Beispiele für das Ergebnis der Faltung:



c) Rufen Sie den AST-Besucher in der Methode `e2c.E2Compiler.performConstantFolding()` geeignet auf (diese Methode wird von den Testfällen verwendet, siehe unten) und sorgen Sie dafür, dass die Konstantenfaltung auch tatsächlich in Ihrem Übersetzer angewendet wird.

Wichtig: Die Methode `performConstantFolding()` bekommt u.U. nur einen „Teil-AST“ als Argument übergeben, beispielsweise einen einzelnen arithmetischen Ausdruck. Durch die Konstantenfaltung kann es passieren, dass die Wurzel des Baums ersetzt wird (wie in dem linken Beispiel oben). Stellen Sie deshalb sicher, dass die Methode immer den richtigen Wurzelknoten als Ergebnis zurückgibt!

d) **Optional**: Die in der Sprachspezifikation beschriebenen Transformationen decken bei weitem nicht alle möglichen Konstantenfaltungen ab, die während der Übersetzung durchgeführt werden könnten. Bessere Verfahren können beispielsweise auch Ausdrücke wie `((a + 2) + 3)` zu `(a + 5)` vereinfachen, indem nicht nur die direkten Kinder binärer Operationen betrachtet werden. Schaffen Sie es, auch solche Ausdrücke zu falten? Tipp: Bringen Sie die Ausdrücke dazu zunächst auf eine Art „Normalform“, indem Sie konstante Operanden nach Möglichkeit durch kommutatives Vertauschen immer zum ersten (oder zweiten) Kind einer binären Operation machen – dies reduziert die Anzahl an zu betrachtenden Fällen.

Wichtig: Implementieren Sie diese fortgeschrittenen Faltungen in einem zweiten Besucher bzw. führen Sie diese nur dann aus, wenn ein entsprechendes Flag gesetzt ist. Stellen Sie insbesondere sicher, dass beim Aufruf in der Methode `performConstantFolding()` nur die in der Sprachspezifikation geforderten Faltungen durchgeführt werden (andernfalls kann es passieren, dass manche der Testfälle fehlschlagen).

e) Falls die Kommandozeilenoption „`--dot`“ gesetzt ist, soll der AST sowohl *vor* als auch *nach* der Konstantenfaltung im Dot-Format (siehe auch Projektübung 1) ausgegeben werden, um das Ergebnis der Faltung visuell überprüfen zu können. Verwenden Sie für die erste Ausgabe den über die Kommandozeile spezifizierten Dateinamen. Ersetzen Sie für die Ausgabe nach der Faltung die Dateiendung durch „`.folded.dot`“.[1]

[1] **Tipp**: `e2c.util.FileUtil.changeFileExtension()`

f) Testen Sie Ihre Implementierung mit Hilfe des *Gradle*-Tasks „`testFolding`“. Sofern Ihre Implementierung alle Testfälle besteht, erscheint am Ende die Ausgabe „`BUILD SUCCESSFUL`“.

Die Testfälle wenden die Konstantenfaltung sowohl auf einzelne Ausdrücke an (um die Funktionalität der Faltung selbst zu testen), als auch auf komplette Programme (um zu überprüfen, ob der Besucher alle Stellen berücksichtigt, an denen ein potentiell faltbarer Ausdruck auftreten kann).

Versehen Sie den finalen Stand Ihrer Implementierung mittels „`git tag`“ mit der Markierung „`folding`“ und pushen Sie Ihre Änderungen auf den `git`-Server des Lehrstuhls.

Hinweis: Diese Projektübung ist nicht Teil eines Meilensteins. Da eine funktionierende Konstantenfaltung aber später noch relevant werden wird (Bestimmung der Größe eines Arrays), sollten Sie sicherstellen, dass Ihre Implementierung alle von uns zur Verfügung gestellten Testfälle besteht.