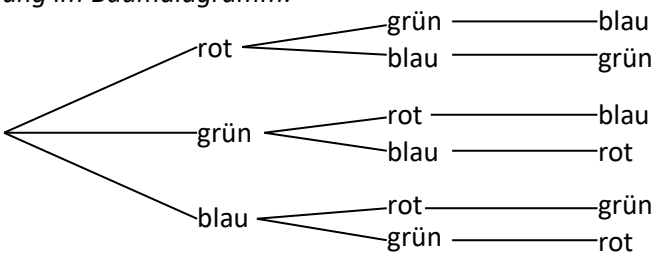
Berechnung der Fakultät einer Zahl

1. In einer Tüte befinden sich eine rote, eine grüne und eine blaue Kugel. Wie viele Möglichkeiten gibt es für die Reihenfolge, in der die Kugeln gezogen werden können?
   Zeichne ein Baumdiagramm, beschreibe die Lösung in Worten und mathematisch.

   Darstellung im Baumdiagramm:



   Für die erste Kugel gibt es 3 Möglichkeiten. Ist die erste Kugel gezogen, bleiben noch

   2 Kugeln übrig. Ist auch die zweite Kugel gezogen, bleibt nur noch eine Kugel übrig.

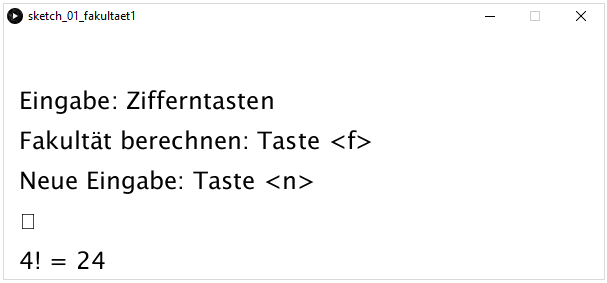
   Es gibt also 3·2·1 = 6 verschiedene Möglichkeiten.

➢ Die **Fakultät !** einer Zahl n ist als Produkt der natürlichen Zahlen von 1 bis n definiert: n! = n(n-1)! Außerdem gilt 0! = 1 (analog zum leeren Produkt)

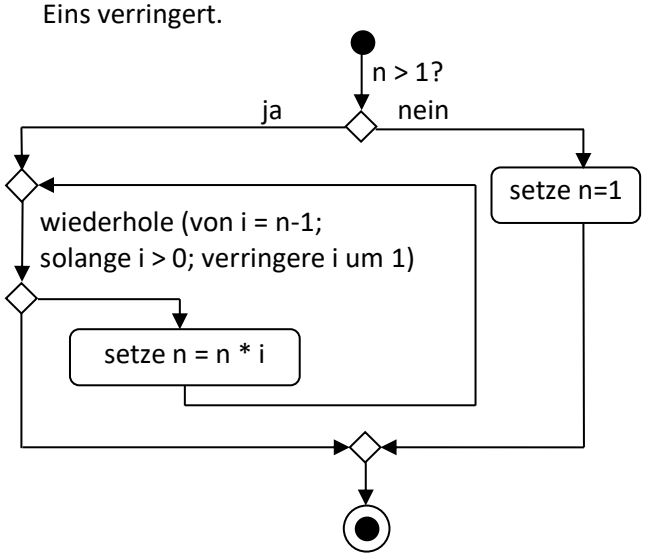
Schreibweise: 4! = 4·3·2·1 = 24

2. Berechne: 5! = *5·4·3·2·1 = 120*

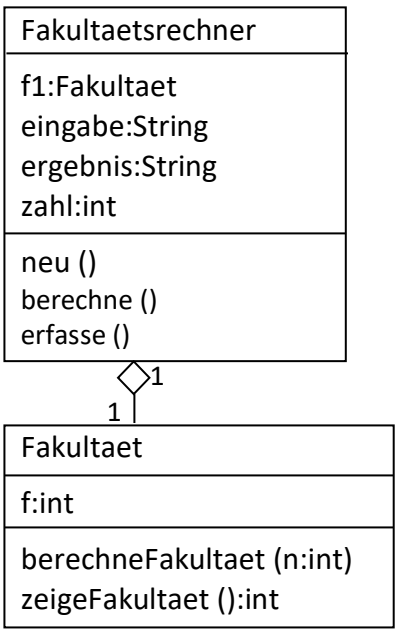
3. Codiere die *Berechnung der Fakultät einer Zahl* in Processing. Speichere die Datei unter `fakultaet1`.
   (*vgl. .\262-materialien\fakultaet\01-fakultaet1*)



   Hinweise:
   - Der Wert für `f` zur Berechnung der Fakultät wird mit Hilfe der Funktion `keyPressed()` in der Methode `erfasse ()` in der Textvariablen `eingabe` erfasst und in einem Textfeld angezeigt.
   - In der Methode `berechneFakultaet ()` der Klasse `Fakultaet` wird die Fakultät einer Zahl `n` in einer Wiederholungsstruktur berechnet.
   - Der Wert der Zählvariablen `i` wird in einer Zählschleife mit der Anweisung `i--` jeweils um Eins verringert.



Aktivitätsdiagramm der Methode berechneFakultaet ()



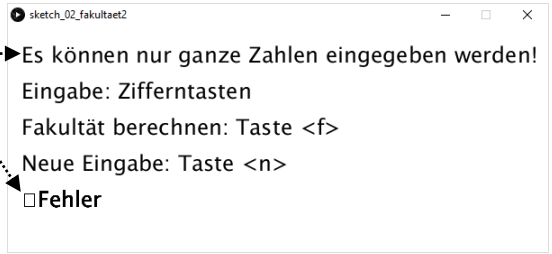
Testen und optimieren des Programms

4. Zur Softwareentwicklung gehört auch, fehlerhafte Benutzereingaben zu verhindern.
   - Teste das Programm `fakultaet1`, indem du gezielt unzulässige Eingaben vornimmst.

     Die Fakultät wird auch ohne Eingabe berechnet.

     Man kann auch Text eintragen.

     Bei fehlerhaften Eingaben wird trotzdem ein Ergebnis angezeigt, was nicht korrekt ist.
   - Ergänze deshalb dein Programm, damit nach Betätigen der Taste <f> eine Fehlermeldung angezeigt wird, wenn eine fehlerhafte Eingabe gemacht wurde.
     Speichere die Datei unter `fakultaet2`.
     (*vgl. .\262-materialien\fakultaet\02-fakultaet2*)

     Hinweise:
     - Die Fehlermeldung **NaN** bedeutet engl. „**N**ot **a** **N**umber“ - keine Zahl.
       Mit der Schreibweise `if (Float.isNaN(float(eingabe)))` kann man überprüfen, ob keine Zahl vorliegt. Damit kann ausgeschlossen werden, dass Buchstaben eingegeben werden.

5. Wenn du das Programm weiter testest, wirst du feststellen, dass 17! eine negative Zahl ergibt. Das ist keineswegs der Fall. Vielmehr können mit dem Datentyp `int` nur Zahlen von –2.147.483.647 bis 2.147.483.647 dargestellt werden. Bei größeren Werten sind die Ergebnisse fehlerhaft.

   Mit dem Datentyp `long` geht der Wertebereich bis zu 9.223.372.036.854.775.807 (`fakultaet3`).
   (*vgl. .\262-materialien\fakultaet\03-fakultaet3*)

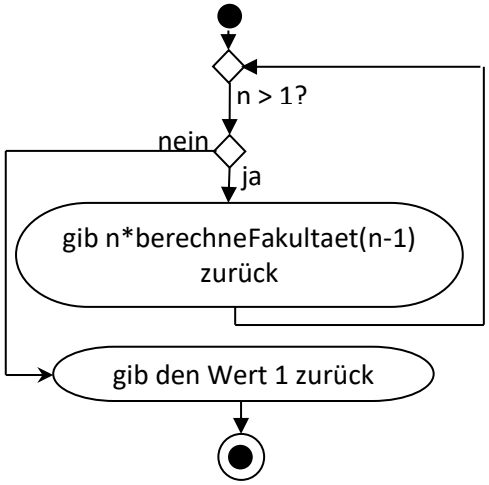
Rekursion

Für die Berechnung der Fakultät einer Zahl gibt es eine elegantere Lösungsmöglichkeit ohne Wiederholungsstruktur:

- Als **Rekursion** bezeichnet man eine Programmiertechnik, in der eine Funktion sich selbst aufruft. Jeder Aufruf der rekursiven Funktion muss sich in endlich vielen Schritten auflösen lassen, sie darf also nicht in eine Endlosschleife geraten.

6. Ändere die Methode `berechneFakultaet ()` in dem Programm `fakultaet3` (Vorlagedatei `v30_fakultaet3`) so ab, dass die Fakultät einer Zahl mit Hilfe einer rekursiven Struktur berechnet wird (siehe Aktivitätsdiagramm rechts).
   Speichere die Datei unter `fakultaet4`.
   (*vgl. .\262-materialien\fakultaet\04-fakultaet4*)

   Solltest du mit dem Aktivitätsdiagramm nicht klarkommen:



Aktivitätsdiagramm: Methode `berechneFakultaet ()` *mit rekursiver Struktur*

```
class Fakultaet {
  Fakultaet () {
  }
  long berechneFakultaet (long n) {
    if (n>1) {
      return (n*berechneFakultaet (n-1));
    }
    else {
      return 1;
} } }
```