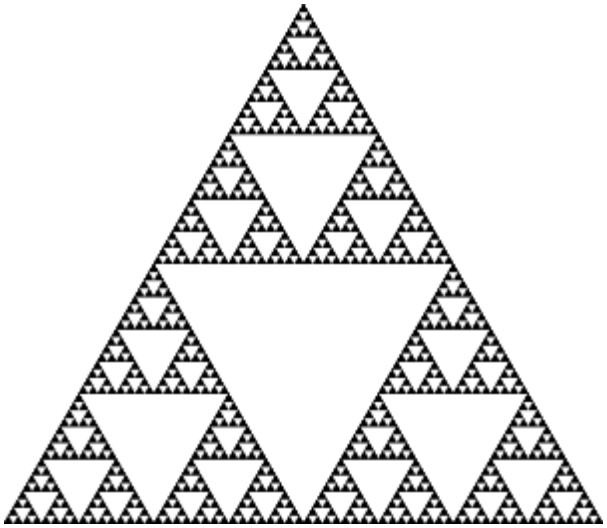
Aufgabe 1

1. Was fällt Ihnen bei diesem Bild auf? Notieren Sie sich mindestens zwei Beobachtungen.



Das Bild besteht komplett aus Dreiecken.
Wiederholungen → Rekursiv
da nicht endlich → kein Algorithmus

Aufgabe 2

2. Bestimmen Sie in der Methode zur rekursiven Berechnung der Fakultät die Rekursionsbasis und die Rekursionsvorschrift.

```
/**
* Rekursive Methode zur Berechnung der Fakultät.
* @param n Wert, für den die Fakultät berechnet wird (n>=0)
* @return n!
*/
public long fak(int n)
{
  if((n == 1)||(n == 0)) {                    // (n == 1) || (n == 0)
    return 1;
  }
  else {
    return (n * fak(n-1));                    // fak(n-1)
  }
}
```

Rekursionsbasis: *(n == 1) || (n == 0)*
Rekursionsvorschrift: *fak(n-1)*

Aufgabe 3

3. Bestimmen Sie die Rekursionsbasis und die Rekursions-vorschrift für den rekursiven Algorithmus des Euklid (ALG1).

```
public int ggtRekursiv(int ersteZahl, int zweiteZahl) {
  if(ersteZahl > zweiteZahl) {
    return ggtRekursiv(ersteZahl-zweiteZahl, zweiteZahl);
  }
  else if(ersteZahl < zweiteZahl) {
    return ggtRekursiv(ersteZahl, zweiteZahl-ersteZahl);
  }
  else { // ersteZahl == zweiteZahl
    return ersteZahl;
  }
}
```

Rekursionsbasis: *Wenn die erste Zahl gleich der zweiten Zahl ist, gib eine der beiden Zahlen zurück (z.B. die erste)*

Rekursionsvorschrift: *Wenn die erste Zahl grösser als die zweite Zahl ist, die zweite Zahl von der ersten Zahl abziehen. Andernfalls die erste Zahl von der Zweiten Zahl abziehen. ggtRekursiv() mit entsprechenden Parameter aufrufen.*

Aufgabe 4

4. Finden Sie eine Aufgabe oder ein Problem aus dem Alltag, welches sich rekursiv lösen lässt. Identifizieren Sie dabei die Rekursionsbasis und die Rekursionsvorschrift.
   Beispiel: Steige Treppe mit n = 1'000 Stufen hoch.
   - Basis: Falls (n == 0), dann mache gar nichts.
   - Vorschrift: Sonst, steige 1 Stufe hoch und noch Treppe mit (n-1) Stufen.

```
public void steigeTreppe(int stufen)
{
  if(stufen > 0) {
    steigeEineStufe();
    steigeTreppe(stufen-1);
  }
}
```

Rekursionsbasis: *Falls (n == 0), dann mache gar nichts.*

Rekursionsvorschrift: *Sonst, steige 1 Stufe hoch und noch Treppe mit (n-1) Stufen.*

Aufgabe 5

5. Beschreiben Sie, was folgende rekursive Methode x() (beschrieben mit Pseudo-Code) beim Lesen von "Hallo." tut:

```
Methode x() : Rückgabewert keiner
  z = liesZeichen()
  falls (z != '.')
  dann
    x()
  sonst
    tue nichts
  gibAus(z)
```

Die Methode gibt den String rückwärts aus (inkl. Punkt).

Hallo. ➔ *.ollaH*

Aufgabe 6

6. Was tut die folgende Methode x(...)?
   Hinweis: Der erste Aufruf von x(...) erfolgt mit Index 0. Indizes sind wie in Java nummeriert: 0 ... (länge-1).

```
Methode x(array[], länge, index): Rückgabewert keiner
  gibAus( array[index] )
  falls (index < (länge -1))
  dann
    x(array, länge, index+1)
  sonst
    tue nichts
```

Die Methode gibt jedes Element vom Array der Reihe nach aus.

Aufgabe 7

Schreiben Sie auf Papier eine rekursive Methode (in Pseudocode oder in Java), die ein Array von Zahlen rückwärts ausgibt.

```
Methode x(array[], index): Rückgabewert keiner
  gibAus( array[index] )
  falls (index > 0)
  dann
    x(array, index-1)
  sonst
    tue nichts
```

Die Methode gibt jedes Element vom Array der Reihe nach aus.