Zufallsgenerator-Klasse `Random`

- in `Basic.java` enthalten
- Konstruktor:
  - `public Random(int min, int max)`
  - erzeugt Zufallsgenerator zur Erzeugung gleichverteilter Zufallszahlen im Bereich `[min, max]`
  - mehrere Methoden um Zufallszahlen zu erzeugen, u.a.
    `public int readInt()`

    hier gilt: `min <= readInt() <= max`
- Code-Beispiel:

```
Random randomGenerator = new Random(0, 1);
randomGenerator.readInt();    // liefert 0 oder 1
```

Rekursion 1

- Rekursion beinhaltet immer:
  - Abbruchbedingung(en) = Rekursionsanker zum Anhalten
  - rekursive(n) Aufruf(e)

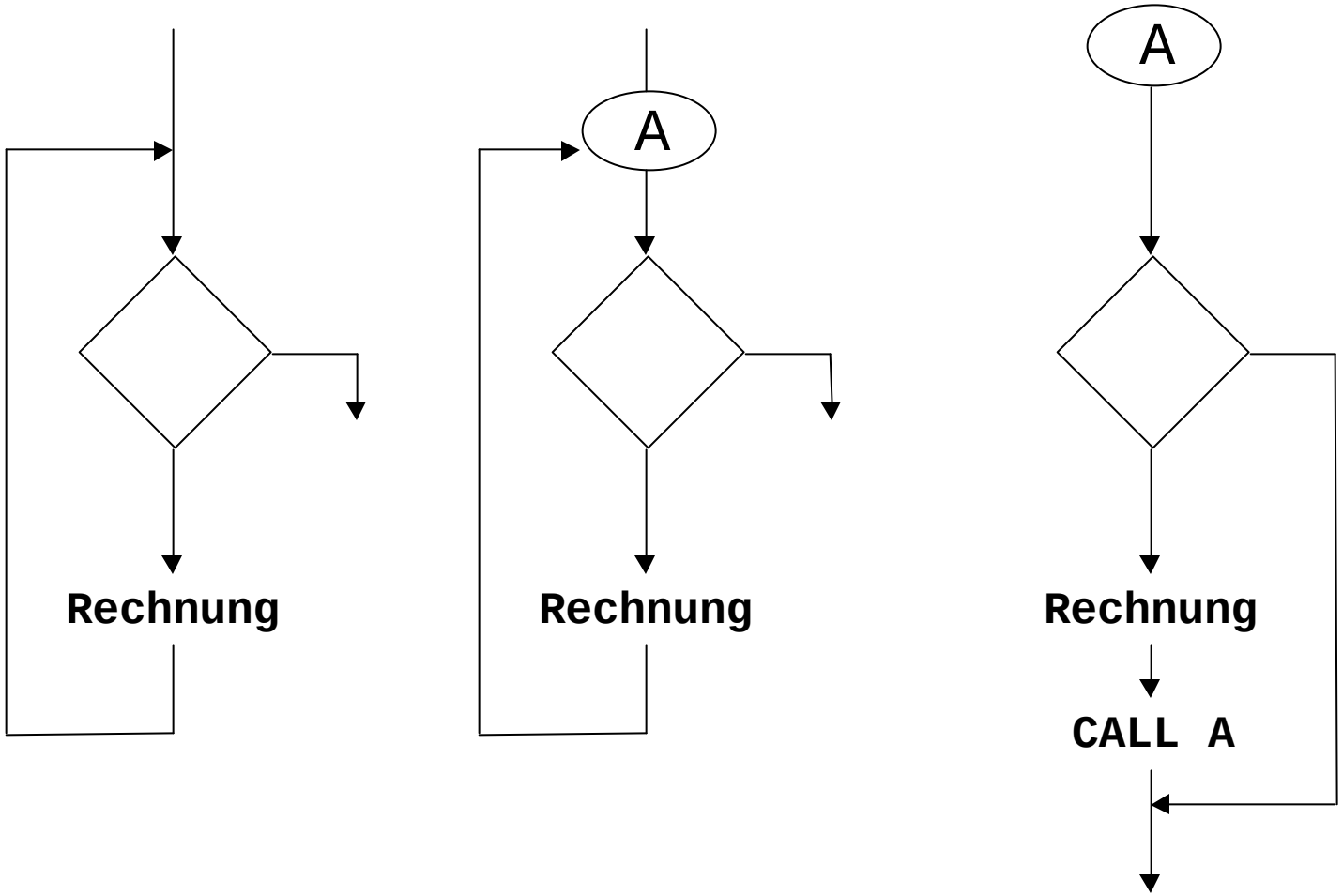
- Jede Rekursion läßt sich in eine Iteration umwandeln und umgekehrt

- Rekursion meist ineffizienter, aber manchmal klarer

  - abhängig von Problemstellung

Rekursion 2

- Umwandlung Schleife → rekursive Prozedur:



Rekursion 3

- Grundstruktur eines rekursiven Algorithmus

```
Rekursion()
   if Abbruchbedingung  -- Rekursionsanker
      löse einfaches Problem;
      gib Ergebnis zurück;
   else -- rekursiver Aufruf
      zerlege Problem in einfachere(s) Problem(e)
      und rufe Rekursion() für diese(s)
      kleinere(n) Problem(e) auf;
      vereinige einfachere Lösung(en);
   end
end
```

Faktorielle

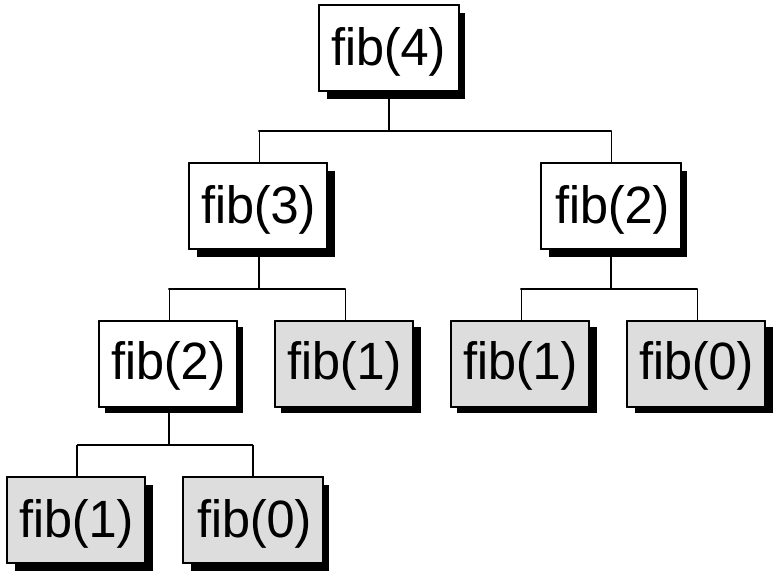
```
public static long fact(long n) {
   if (n == 1) // Rekursionsanker
        return 1;
   else // rekursiver Aufruf
        return fact(n-1) * n;
}
```

$$\text{fact}(n) = \begin{cases} 1 & \text{für } n = 1 \\ \text{fact}(n-1) * n & \text{für } n > 1 \end{cases}$$

Aufruf	**n**	**Zwischenergebnis**
fact(4) = ?		24
fact(3) * 4	4	6 * 4
fact(2) * 3	3	2 * 3
fact(1) * 2	2	1 * 2
1	1	1

Fibonacci-Folge

$$\text{fib}(n) = \begin{cases} 0 & \text{für } n = 0 \\ 1 & \text{für } n = 1 \\ \text{fib}(n-1) + \text{fib}(n-2) & \text{für } n > 1 \end{cases}$$




```
public static long fib(long n) {
  if (n <= 1)  // Rekursionsanker
    return n;
  else  // rekursiver Aufruf
    return fib(n-1) + fib(n-2);
}
```

Binärbaum

- Ist eine rekursive Datenstruktur
- Ein Binärbaum ist
  - ein Einzelelement gefolgt von einem linken und einem rechten (Teil-)baum oder
  - ein leerer Baum

- Rekursive Traversierung
  - Preorder (Wurzel, links, rechts)
  - Inorder (links, Wurzel, rechts)
  - Postorder (links, rechts, Wurzel)