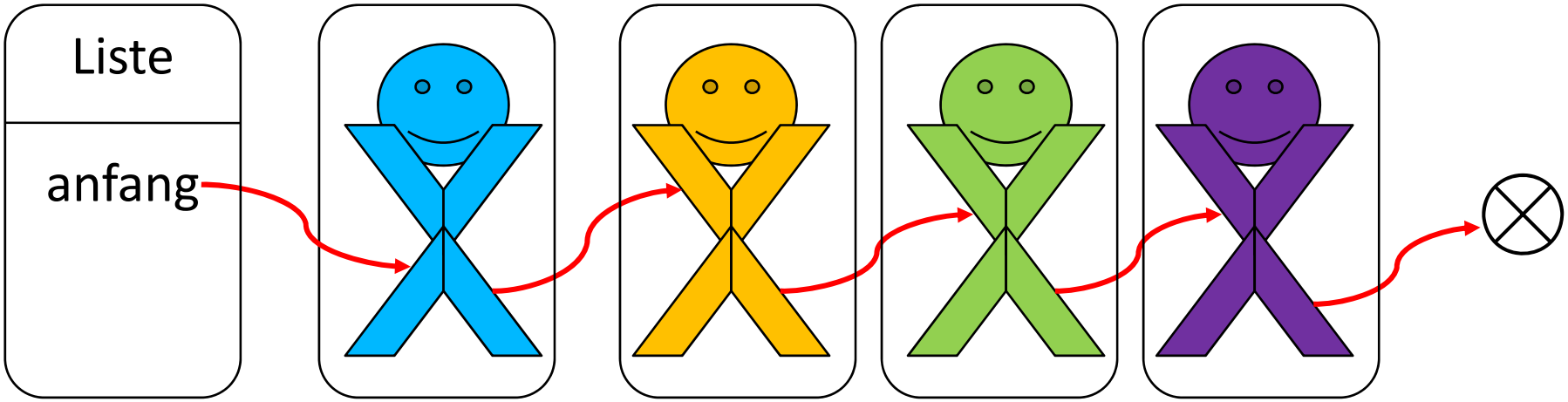
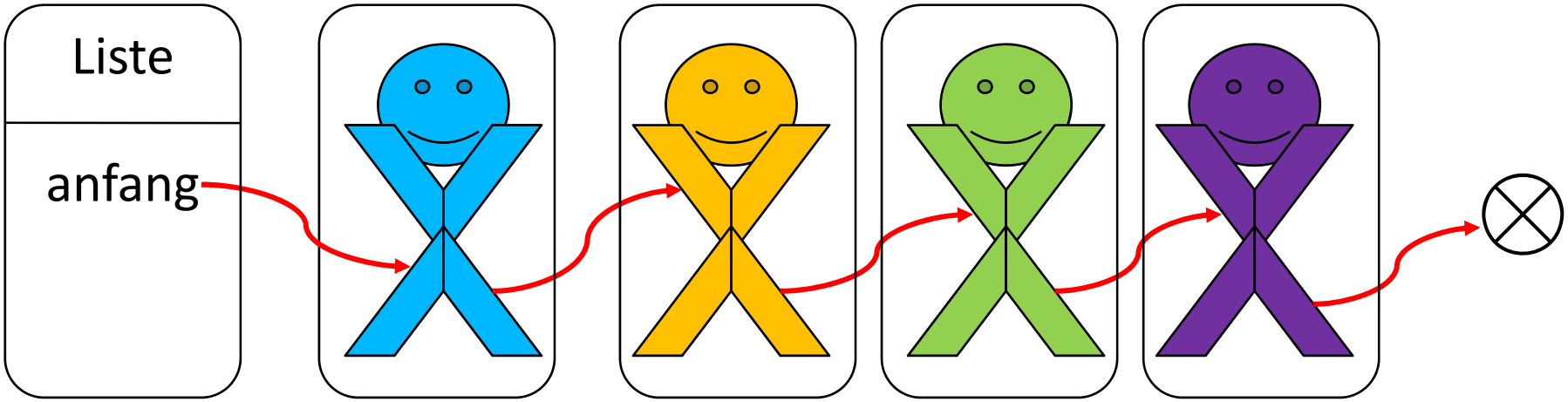
Listen

Rekursive Methoden



Listen

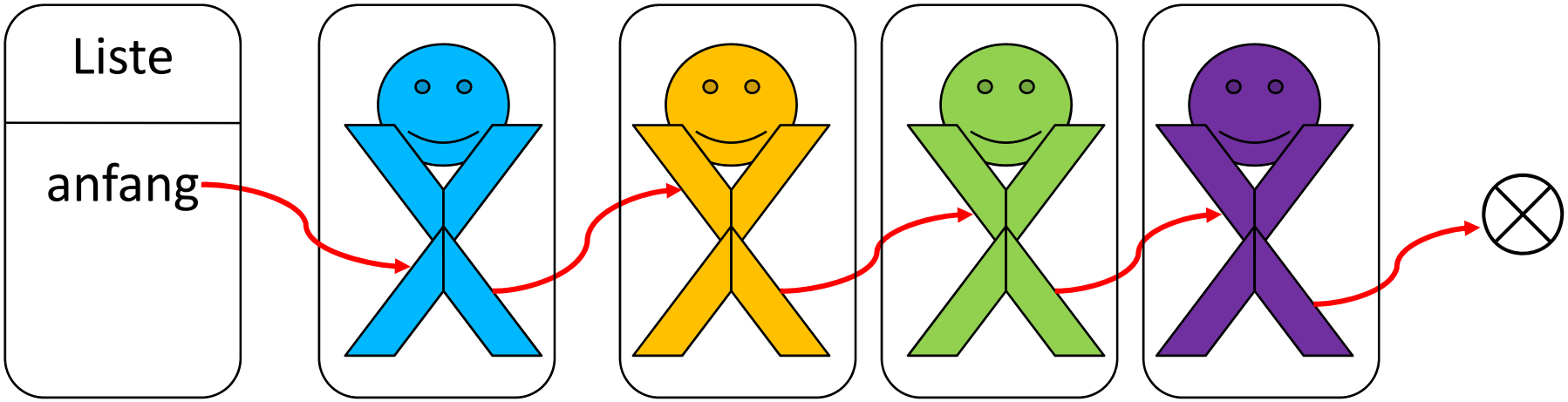
Rekursive Methoden



**Es befindet sich keine Referenz mehr auf das letzte Listenelement.
Trotzdem soll die Länge der Liste bestimmt werden.
Wie lautet die Signatur der Methode?**

Listen

Rekursive Methoden



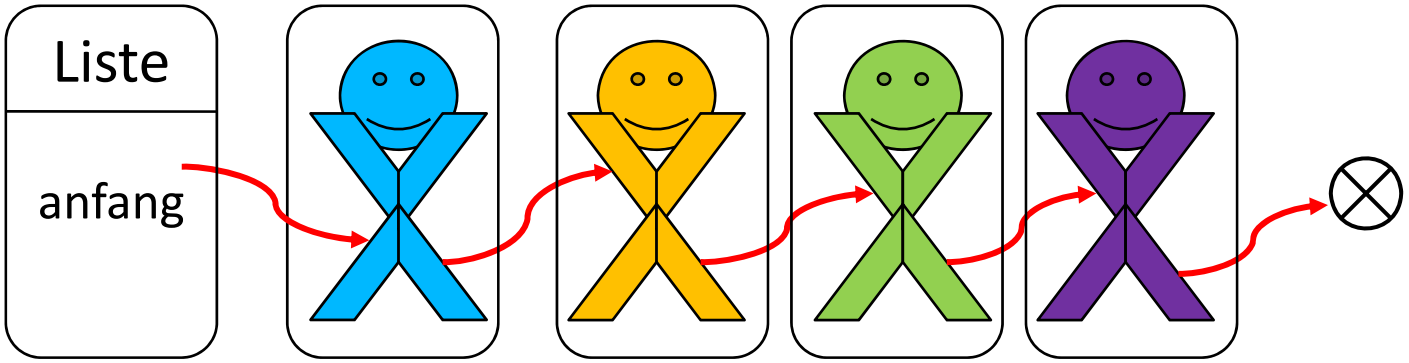

```
public int size(){

}
```

Listen

Rekursive Methoden

Zunächst befinden wir uns in der Klasse Liste. Hier ist zunächst nur Auszuschließen, dass es sich um eine leere Liste handelt, dann ist Der Fall klar. Programmiere!



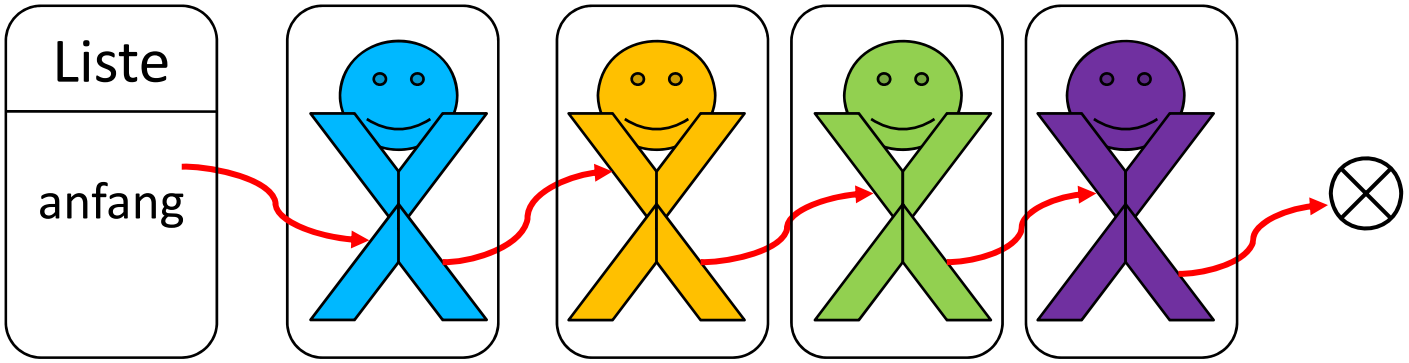

```
public int size(){

}
```

Listen

Rekursive Methoden

Zunächst befinden wir uns in der Klasse Liste. Hier ist zunächst nur auszuschließen, dass es sich um eine leere Liste handelt, dann ist Der Fall klar. Programmiere!

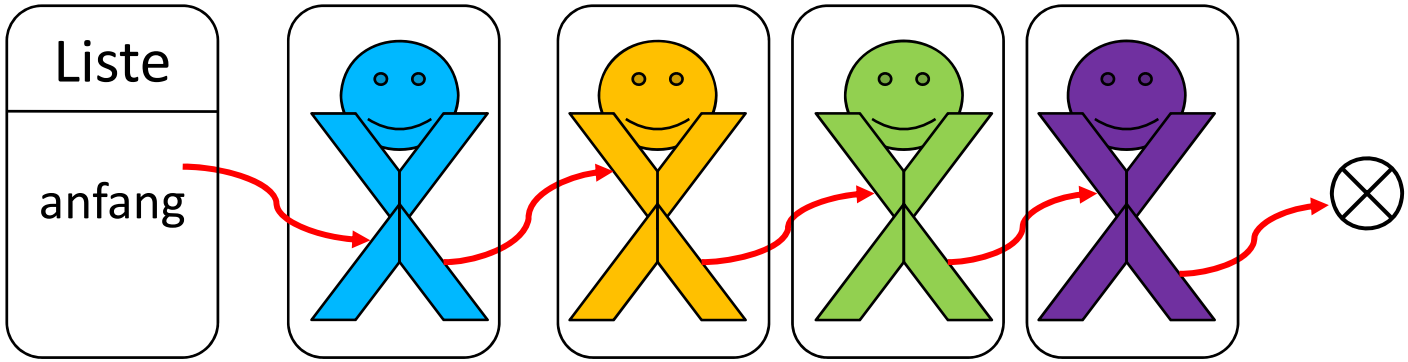



```
public int size(){
   if (anfang==null) return 0;
   else return anfang.size();
}
```

Listen

Rekursive Methoden

Die Methode size(), die in der Klasse Knoten vorhanden ist, ist eine völlig andere als die in der Klasse Liste! Programmiere rekursiv.

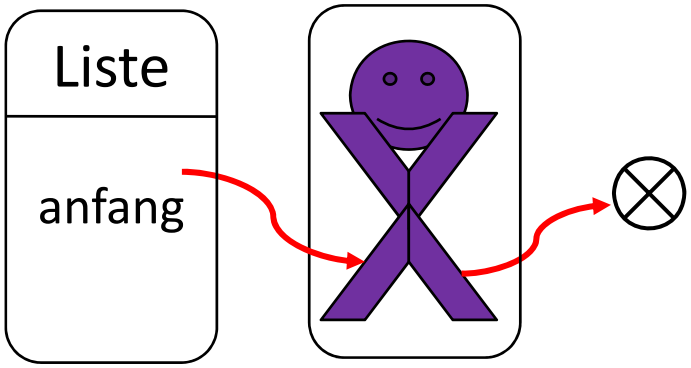


Was ist der einfachste Fall? (Basisfall)?
Wie kann ich jeden Fall auf diesen zugehen? (Rekursionsschritt)?

Listen

Rekursive Methoden

Der einfachste Fall ist, dass es nur ein Element gibt. Null haben wir ja ausgeschlossen. Das kann ich prüfen, indem ich nachsehe, ob das Objekt einen Nachfolger hat.

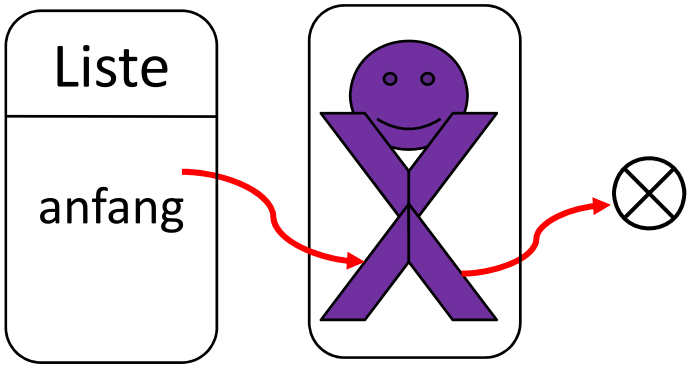


Programmiere den Basisfall!

Listen

Rekursive Methoden

Zunächst befinden wir uns in der Klasse Liste. Hier ist zunächst nur Auszuschließen, dass es sich um eine leere Liste handelt, dann ist Der Fall klar. Programmiere!

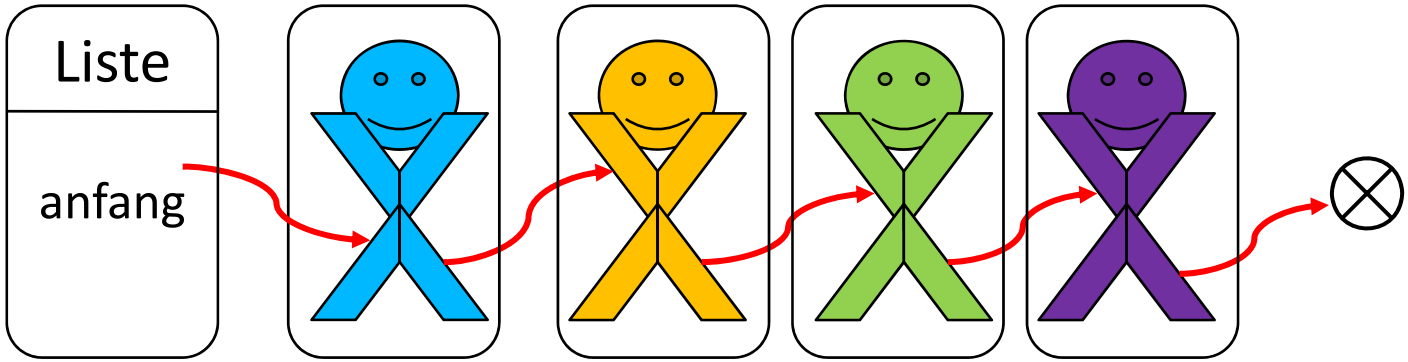



```
public int size(){
   if (nachfolger==null) return 1;
   else …???
}
```

Listen

Rekursive Methoden

Programmiere den Rekursionsschritt!

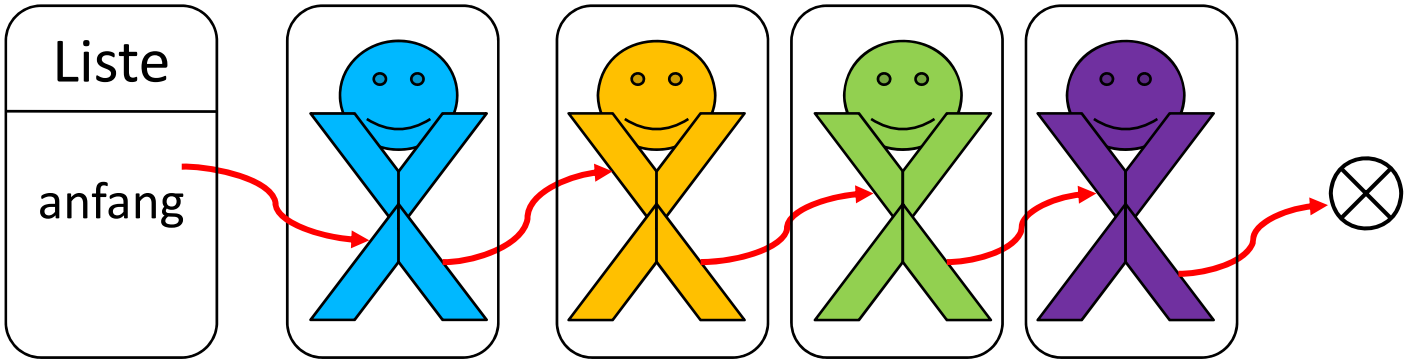



```
public int size(){
    if (nachfolger==null) return 1;
    else …???
}
```

Listen

Rekursive Methoden

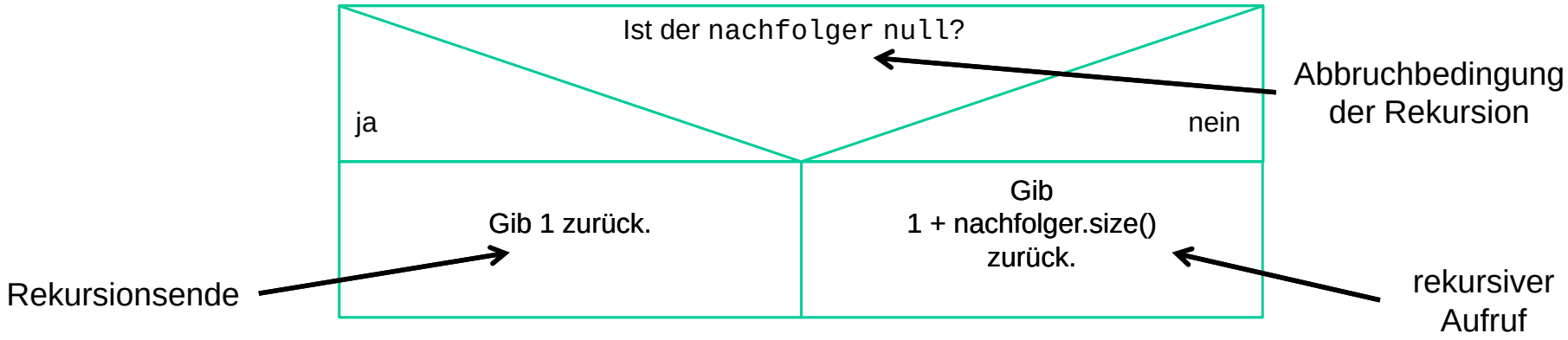
Programmiere den Rekursionsschritt!




```
public int size(){
    if (nachfolger==null) return 1;
    else return 1+nachfolger.size();
}
```

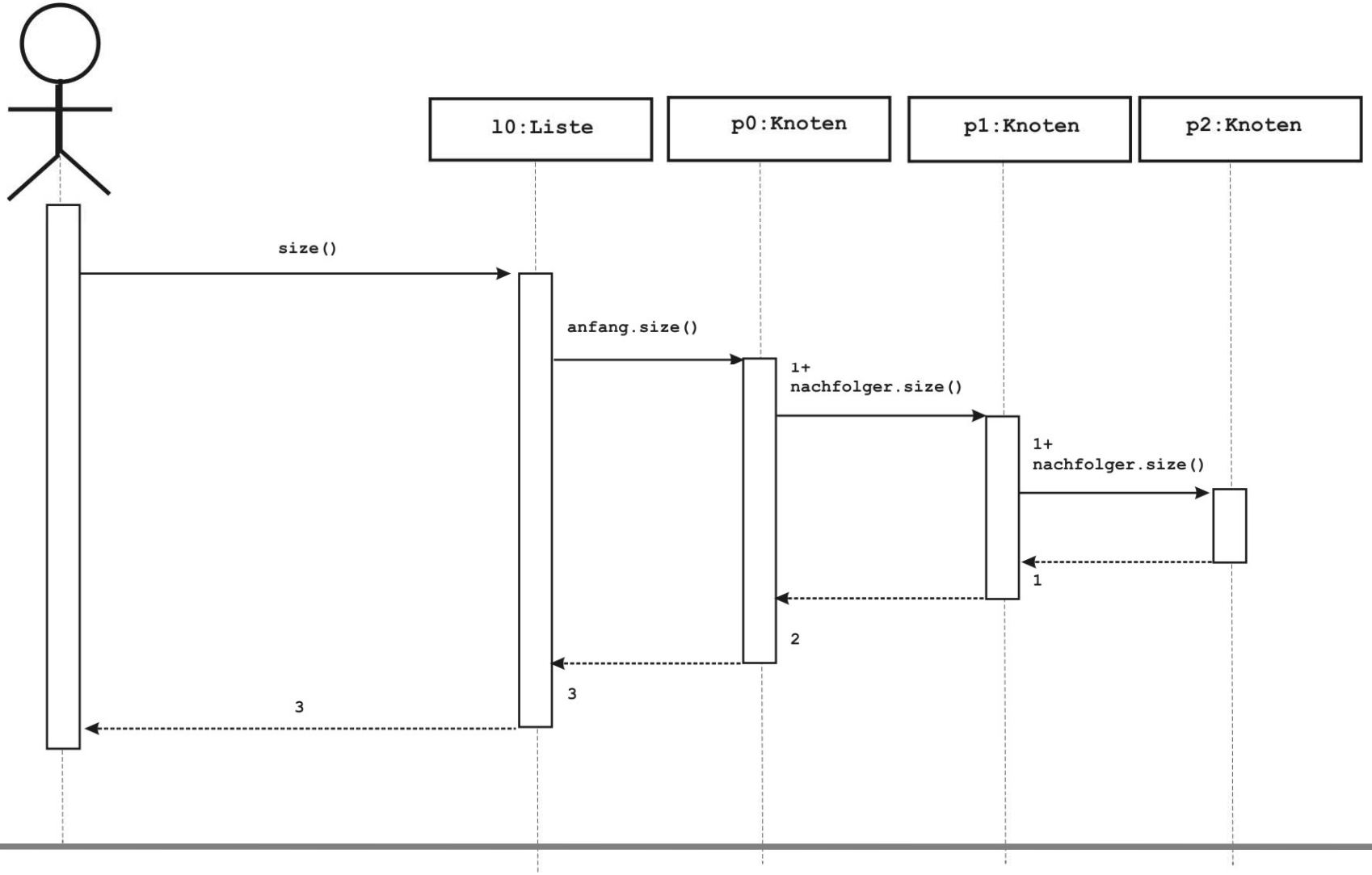
Listen

Rekursive Methoden



Listen

Rekursive Methoden



Listen

Rekursive Methoden

Lade Dir die Vorlage für BlueJ

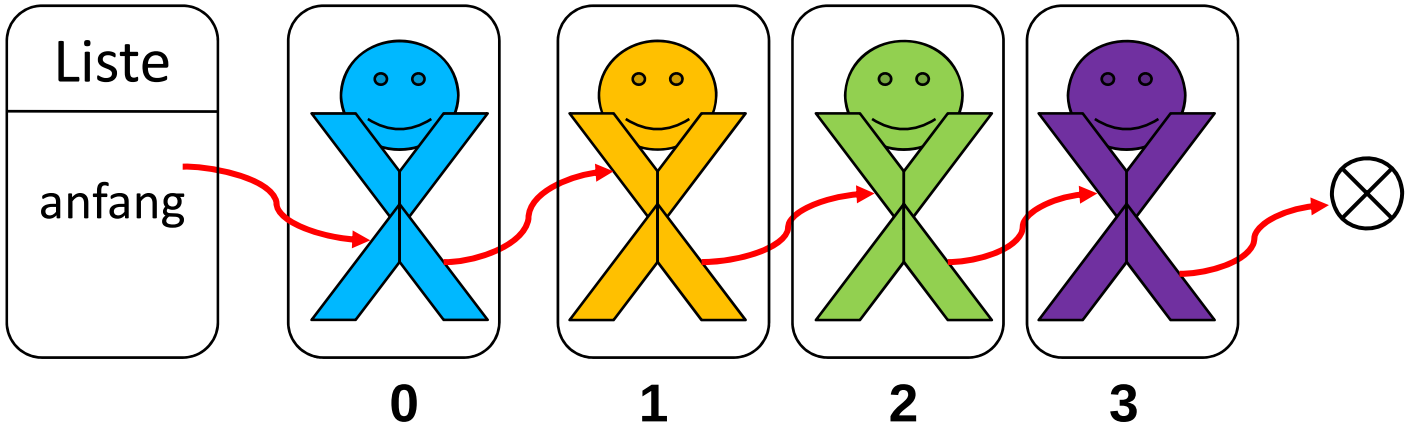
Liste_04_Aufgabe herunter.

(www.manuel-friedrich.net, Informatik, 11. Jgst.) und programmiere die Methode **`size()`.**

Listen

Die Methode `getKnoten(int i) : Knoten`

i stellt den Index dar, beginndend mit 0, als Ergebnis liefert die Methode eine Referenz auf den Knoten zurück.

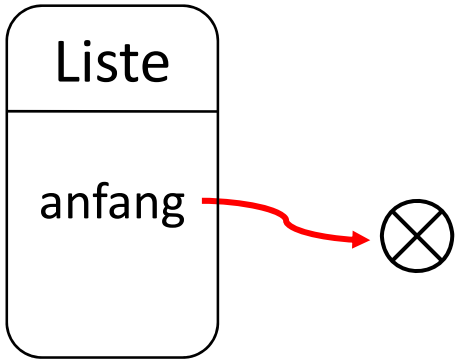


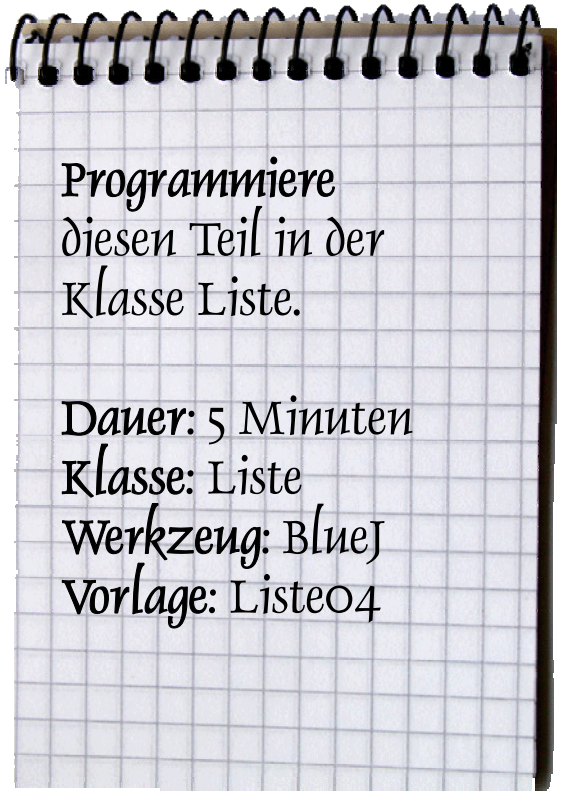

```
public int getKnoten(int i){
    …
}
```

Listen

Die Methode `getKnoten(int i) : Knoten`

In der Klasse Liste müssen wir unterscheiden, ob die Liste leer ist…





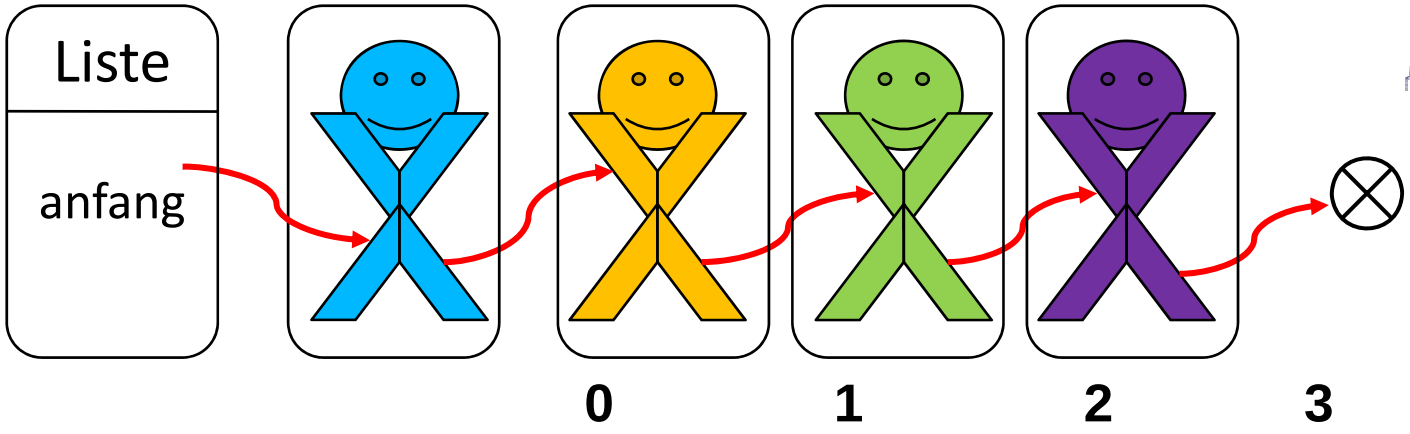

```
public int getKnoten(int i){

}
```

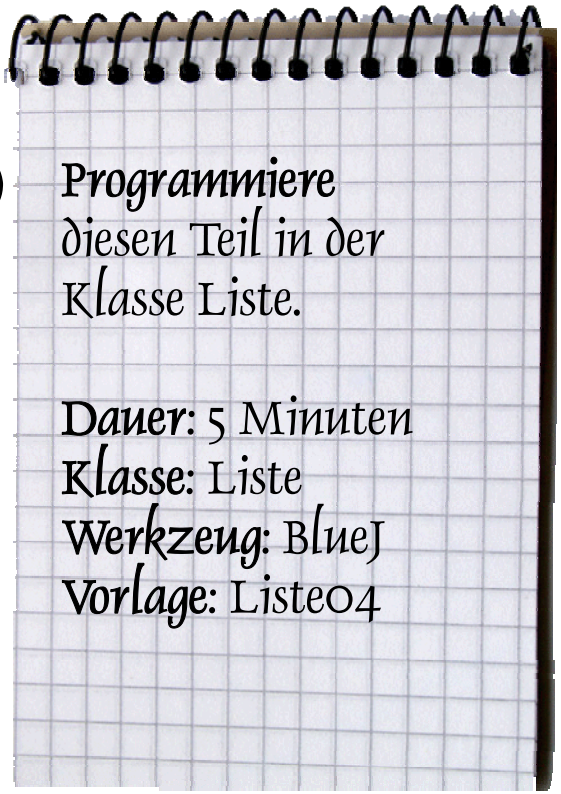
Listen

Die Methode `getKnoten(int i) : Knoten`

… oder ob sich mindestens 1 Knoten in der Liste befindet. Dann wird der Methodenaufruf getKnoten(int i) einfach an das erste Objekt weitergegeben.



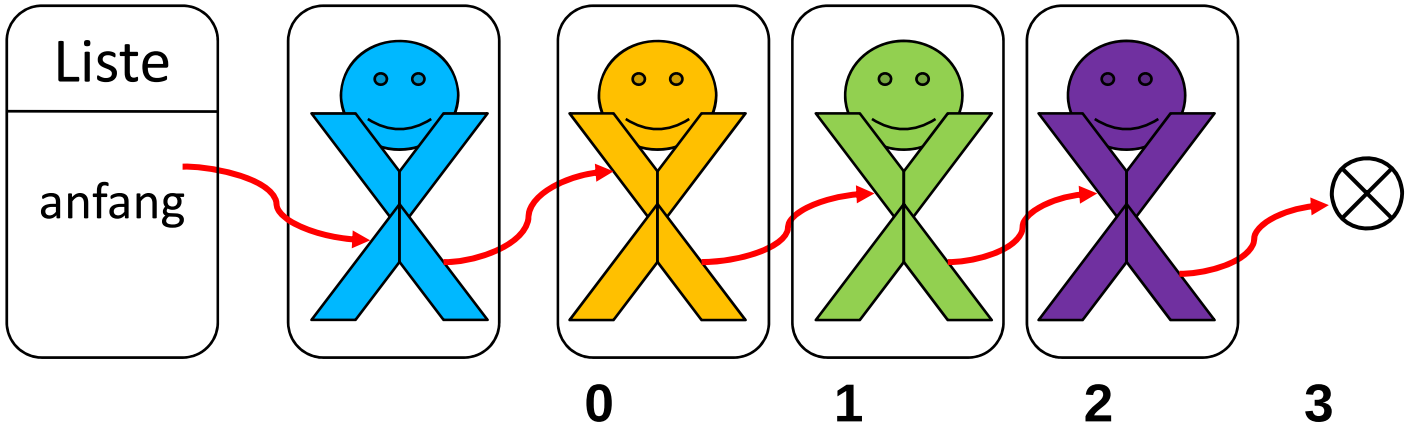

```
public int getKnoten(int i){
    if (anfang==null) return null;
    else …
}
```



Listen

Die Methode `getKnoten(int i) : Knoten`

In der Klasse Knoten wird nun der Rekursive Aufruf durchgeführt. Der Methodenname ist gleich.

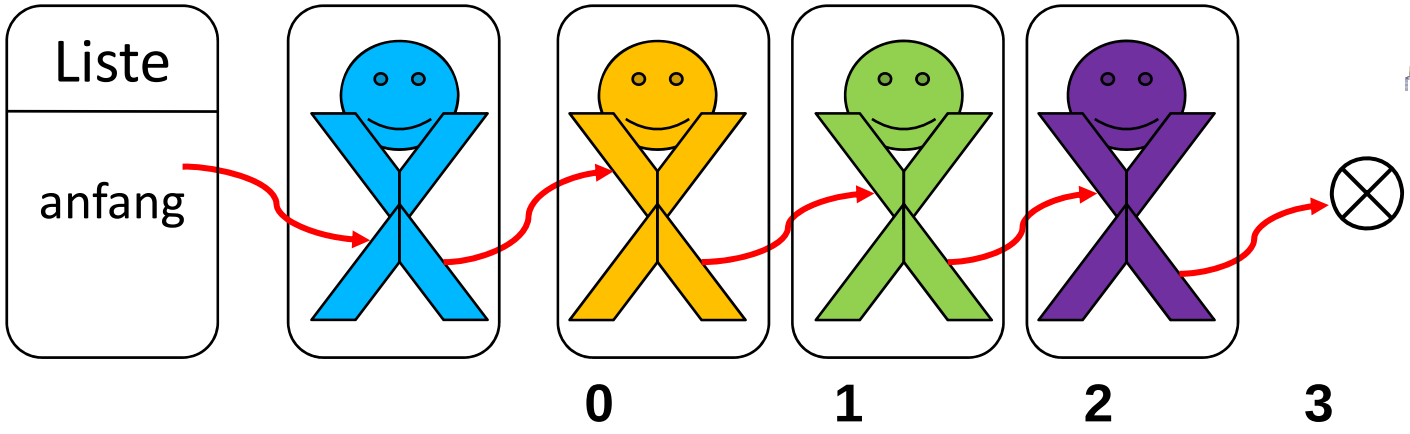



```
public int getKnoten(int i){
   …
}
```

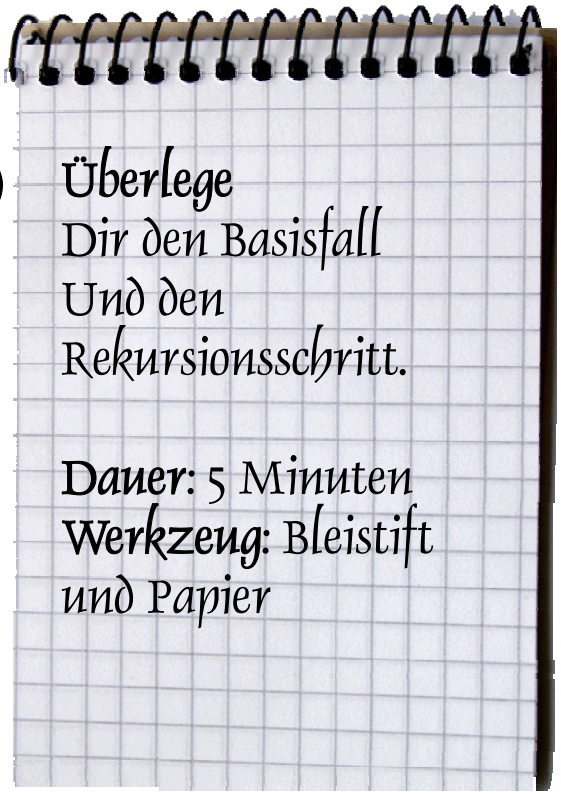
Listen

Die Methode `getKnoten(int i) : Knoten`

Für einen rekursiven Aufruf benötigen wir den Basisfall und den Rekursionsschritt.

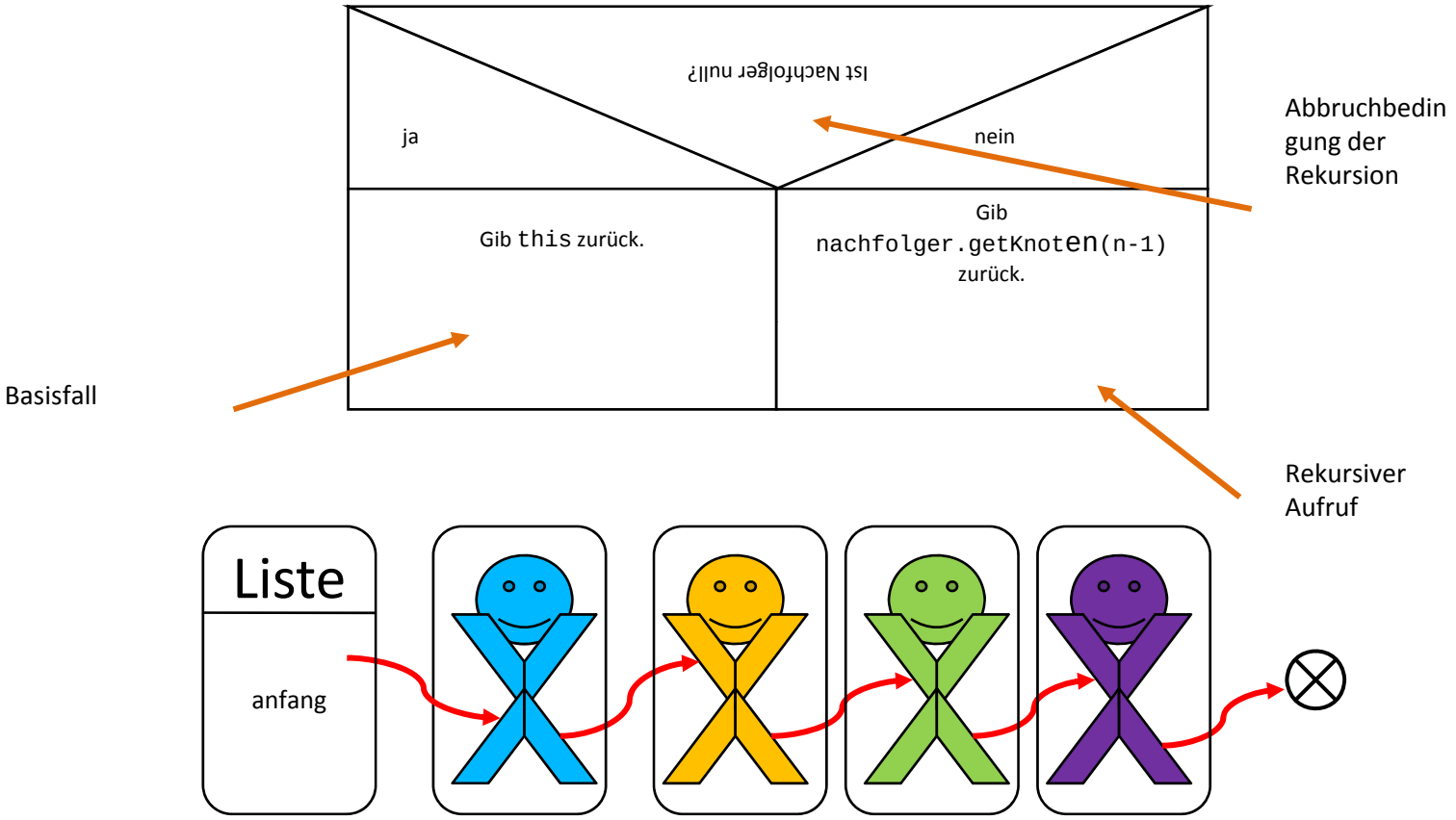



```
public int getKnoten(int i){
   …
}
```



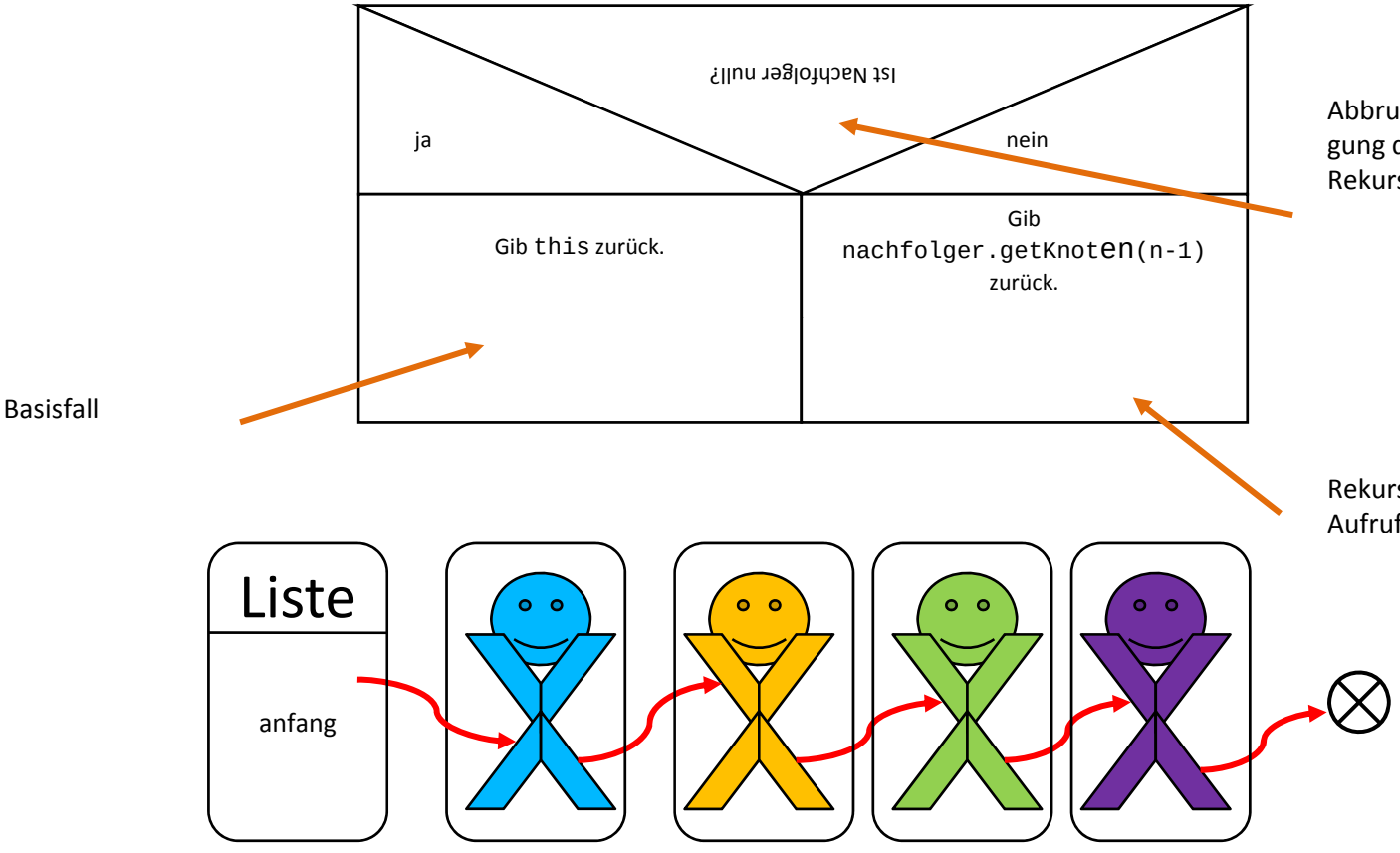
Listen

Die Methode `getKnoten(int i) : Knoten`



Listen

Die Methode `getKnoten(int i) : Knoten`



Programmiere diesen Teil in der Klasse Knoten.

Dauer: 5 Minuten
Klasse: Knoten
Werkzeug: BlueJ
Vorlage: Liste04

Teste danach die Methode

Listen

Die Methode `getKnoten(int i) : Knoten`

Für einen rekursiven Aufruf benötigen wir den Basisfall und den Rekursionsschritt.

```
public int getKnoten(int i){
  if (i==0) return this;
  else
  return getKnoten(i-1);
}
```