Aufgabe 6: Rekursion

Für Binomialkoeffizienten $\binom{n}{k}$ gelten neben den grundlegenden Beziehungen $\binom{n}{0} = 1$ und $\binom{n}{n} = 1$ auch folgende Formeln:

A $\binom{n+1}{k} = \binom{n}{k-1} + \binom{n}{k}$

B $\binom{n}{k} = \binom{n-1}{k-1} \cdot \frac{n}{k}$

(a) Implementieren Sie unter Verwendung von Beziehung (A) eine rekursive Methode `binRek(n, k)` zur Berechnung des Binomialkoeffizienten in einer objektorientierten Programmiersprache oder entsprechendem Pseudocode!

Zuerst verwandeln wir die Beziehung (A) geringfügig um, indem wir n durch $n-1$ ersetzen:
$\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$

```java
public static int binRek(int n, int k) {
  if (k == 0 || k == n) {
    return 1;
  } else {
    return binRek(n - 1, k - 1) + binRek(n - 1, k);
  }
}
```

Code-Beispiel auf Github ansehen:
src/main/java/org/bschlangaul/examen/examen_46115/jahr_2014/fruehjahr/rekursion/Rekursion.java

(b) Implementieren Sie unter Verwendung von Beziehung (B) eine iterative Methode `binIt(n, k)` zur Berechnung des Binomialkoeffizienten in einer objektorientierten Programmiersprache oder entsprechendem Pseudocode!

```java
public static int binIt(int n, int k) {
  // Das Ergebnis wird als Kommazahl deklariert, da nicht alle
  // Zwischenergebnisse ganze Zahlen sind.
  double ergebnis = 1;
  while (k > 0) {
    ergebnis = ergebnis * n / k;
    n--;
    k--;
  }
  // Vor dem Zurückgeben kann das Ergebnis nun in eine ganze Zahl
  // umgewandelt werden.
  return (int) ergebnis;
}
```

Code-Beispiel auf Github ansehen:
src/main/java/org/bschlangaul/examen/examen_46115/jahr_2014/fruehjahr/rekursion/Rekursion.java

(c) Geben Sie die Laufzeitkomplexität der Methoden `binRek(n, k)` und `binIt(n, k)` aus den vorhergehenden beiden Teilaufgaben in O-Notation an!

```java
/**
 *
```

```
 */
public class Rekursion {

  /**
   * Berechnet rekursiv den Binominalkoeffizienten „n über k". Dabei muss gelten:
   * n &#x3E;= 0, k &#x3E;= 0 und n &#x3E;= k.
   *
   * @param n Ganzzahl n
   * @param k Ganzzahl k
   *
   * @return Eine Ganzzahl.
   */
  public static int binRek(int n, int k) {
    if (k == 0 || k == n) {
      return 1;
    } else {
      return binRek(n - 1, k - 1) + binRek(n - 1, k);
    }
  }

  /**
   * Berechnet iterativ den Binominalkoeffizienten „n über k". Dabei muss gelten:
   * n &#x3E;= 0, k &#x3E;= 0 und n &#x3E;= k.
   *
   * @param n Ganzzahl n
   * @param k Ganzzahl k
   *
   * @return Eine Ganzzahl.
   */
  public static int binIt(int n, int k) {
    // Das Ergebnis wird als Kommazahl deklariert, da nicht alle
    // Zwischenergebnisse ganze Zahlen sind.
    double ergebnis = 1;
    while (k > 0) {
      ergebnis = ergebnis * n / k;
      n--;
      k--;
    }
    // Vor dem Zurückgeben kann das Ergebnis nun in eine ganze Zahl
    // umgewandelt werden.
    return (int) ergebnis;
  }
}
```

Code-Beispiel auf Github ansehen: `src/main/java/org/bschlangaul/examen/examen_46115/jahr_2014/fruehjahr/rekursion/Rekursion.java`