Wegberechnung im Gitter

Betrachten Sie das folgende Gitter mit $m+1$ Zeilen und $n+1$ Spalten ($m \geq 1$ und $n \geq 1$):[1] geeksforgeeks[2]

Angenommen, Sie befinden sich zu Beginn am Punkt $(0,0)$ und wollen zum Punkt (m,n).

Für die Anzahl $A(i,j)$ aller verschiedenen Wege vom Punkt $(0,0)$ zum Punkt (i,j) lassen sich folgende drei Fälle unterscheiden (es geht jeweils um die kürzesten Wege ohne Umweg!):

- $1 \leq i \leq m$ und $j = 0$:

  Es gibt genau einen Weg von $(0,0)$ nach $(i,0)$ für $1 \leq i \leq m$.

- $i = 0$ und $1 \leq j \leq n$:

  Es gibt genau einen Weg von $(0,0)$ nach $(0,j)$ für $1 \leq j \leq n$.

- $1 \leq i \leq m$ und $1 \leq j \leq n$:

  auf dem Weg zu (i,j) muss als vorletzter Punkt entweder $(i-1,j)$ oder $(i,j-1)$ besucht worden sein.

Daraus ergibt sich folgende Rekursionsgleichung:

$$A(i,j) = \begin{cases} 1 & \text{falls } (1 \leq i \leq m \text{ und } j = 0) \text{ oder } (i = 0 \text{ und } 1 \leq j \leq n) \\ A(i-1,j) + A(i,j-1) & \text{falls } 1 \leq i \leq m \text{ und } 1 \leq j \leq n \end{cases}$$

Implementieren Sie die Java-Klasse `Gitter` mit der Methode

```
public int berechneAnzahlWege(),
```

die ausgehend von der Rekursionsgleichung durch dynamische Programmierung die Anzahl aller Wege vom Punkt $(0,0)$ zum Punkt (m,n) berechnet. Die Überprüfung, ob $m \leq 1$ und $n \leq 1$ gilt, können Sie der Einfachheit halber weglassen.

```java

  public int berechneAnzahlWege() {
    int i, j;
    for (i = 1; i <= m; i++) {
      anzahlWege[i][0] = 1;
    }
    for (j = 1; j <= n; j++) {
      anzahlWege[0][j] = 1;
    }
    for (i = 1; i <= m; i++) {
      for (j = 1; j <= n; j++) {
        anzahlWege[i][j] = anzahlWege[i - 1][j] + anzahlWege[i][j - 1];
      }
    }
```

[1] Quelle möglicherweise von `https://www.yumpu.com/de/document/read/17936760/ubungen-zum-prasenzmodul-algorithmen-und-datenstrukturen`

[2] `https://www.geeksforgeeks.org/count-possible-paths-top-left-bottom-right-nxm-matrix/`

```
    return anzahlWege[m][n];
```