rekursives Backtracking

Folgende Methode soll das Feld a (garantiert der Länge $2n$ und beim ersten Aufruf von außen mit 0 initialisiert) mittels rekursivem Backtracking so mit Zahlen $1 \leq x \leq n$ befüllen, dass jedes x genau zweimal in a vorkommt und der Abstand zwischen den Vorkommen genau x ist. Sie soll genau dann `true` zurückgeben, wenn es eine Lösung gibt.

Beispiele:

- `fill(2, [])` → `false`
- `fill(3, [])` → `[3; 1; 2; 1; 3; 2]`
- `fill(4, [])` → `[4; 1; 3; 1; 2; 4; 3; 2]`

```
boolean fill (int n , int[] a) {
  if (n <= 0) {
    return true;
  }
  // TODO
  return false;
}
```

```

  public static boolean fill(int n, int[] a) {
    if (n <= 0) {
      return true;
    }
    for (int i = 0; i < a.length - n - 1; i++) {
      int j = i + n + 1; // Zwischen i und j muessen genau n andere Zahlen sein
      if (a[i] == 0 && a[j] == 0) {
        a[i] = a[j] = n;
        if (fill(n - 1, a)) {
          return true;
        }
        a[i] = a[j] = 0;
      }
    }
    return false;
  }
```

```
fill(0, []):
fill(1, []): false
fill(2, []): false
fill(3, []): 3 1 2 1 3 2
fill(4, []): 4 1 3 1 2 4 3 2
fill(5, []): false
fill(6, []): false
fill(7, []): 7 3 6 2 5 3 2 4 7 6 5 1 4 1
fill(8, []): 8 3 7 2 6 3 2 4 5 8 7 6 4 1 5 1
fill(9, []): false
fill(10, []): false
fill(11, []): 11 6 10 2 9 3 2 8 6 3 7 5 11 10 9 4 8 5 7 1 4 1
```