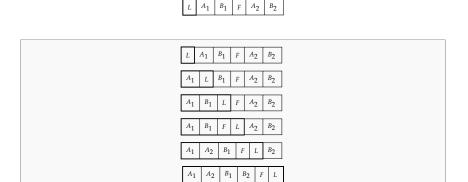
## Aufgabe 7

(a) Führen Sie "Sortieren durch Einfügen" lexikographisch aufsteigend und insitu (in-place) so in einem Schreibtischlauf auf folgendem Feld (Array) aus, dass gleiche Elemente ihre relative Abfolge jederzeit beibehalten (also dass z. B.  $A_1$  stets vor  $A_2$  im Feld steht). Jede Zeile stellt den Zustand des Feldes dar, nachdem das jeweils nächste Element in die Endposition verschoben wurde. Der bereits sortierte Teilbereich steht vor |||. Gleiche Elemente tragen zwecks Unterscheidung ihre "Objektidentität" als Index (z. B. "A1". equals ("A2") aber "A1". != "A2")



(b) Ergänzen Sie die folgende Methode so, dass sie die Zeichenketten im Feld a lexikographisch aufsteigend durch Einfügen sortiert. Sie muss zum vorangehenden Ablauf passen, d. h. sie muss *iterativ* sowie *in-situ* (*in-place*) arbeiten und die relative Reihenfolge gleicher Elemente jederzeit beibehalten. Sie dürfen davon ausgehen, dass kein Eintrag im Feld null ist.

```
void sortierenDurchEinfuegen(String[] a) {
// Hilfsvariable:
String tmp;
```

```
static void sortierenDurchEinfuegen(String[] a) {
          // Hilfsvariable:
          String tmp;
          for (int i = 1; i < a.length; i++) {
            tmp = a[i];
            int j = i;
10
            while (j > 0 \&\& a[j - 1].compareTo(tmp) >= 1) {
11
               a[j] = a[j - 1];
12
               j = j - 1;
13
14
15
            a[j] = tmp;
                           Code-Beispiel auf Github ansehen: src/main/java/org/bschlangaul/examen/examen_46115/jahr_2017/herbst/InsertionSort.java
```