Selection Sort

Schritt	*Indizes* 0	1	2	3	4	5	6	*Index aktuell*	*Index Min.*
1	**4**	**39**	**4**	**0**	**21**	**8**	**11**	**0**	**3**
2	**0**	**39**	**4**	**4**	**21**	**8**	**11**	**1**	**2**
3	**0**	**4**	**39**	**4**	**21**	**8**	**11**	**2**	**3**
4	**0**	**4**	**4**	**39**	**21**	**8**	**11**	**3**	**5**
5	**0**	**4**	**4**	**8**	**21**	**39**	**11**	**4**	**6**
6	**0**	**4**	**4**	**8**	**11**	**39**	**21**	**5**	**6**
7	**0**	**4**	**4**	**8**	**11**	**21**	**39**		

Äußere Schleife: Wiederhole bis zum vorletzten Element des Arrays:

Suche das Minimum im verbleibenden Array:
Setze Variable min auf das aktuelle Element der äußeren Schleife
Setze Variable minIndex auf den aktuellen Index

Innere Schleife: Wiederhole vom akt. Element der äußeren Schleife an:

Falls das akt. Element der inneren Schleife kleiner als min ist,
setze min auf dieses Element, und
setze minIndex auf dessen Position

Tausche: Setze Element an der Stelle minIndex auf das akt. Element der äußeren Schl.
Setze akt. Element der äußeren Schl. auf min.

```
int i, j;            // Variablen für äußere und innere Schleife
int min, minIndex;

for (i = 0; i < liste.length - 1; i++)
{
   // Suche Minimum und dessen Position (vom aktuellen Index an)
   min = liste[i];
   minIndex = i;
   for (j = i + 1; j < liste.length; j++)
   {
      if (liste[j] < min)
      {
         min = liste[j];
         minIndex = j;
      }
   }
   // Vertausche Minimum mit dem aktuellen Element
   liste[minIndex] = liste[i];
   liste[i]        = min;
}
```

Bubble Sort

Schritt	*Indizes*							*Index aktuell*
	0	*1*	*2*	*3*	*4*	*5*	*6*	
1	4	39	4	0	21	8	2	0
2	4	39	4	0	21	8	2	1
3	4	4	39	0	21	8	2	2
4	4	4	0	39	21	8	2	3
5	4	4	0	21	39	8	2	4
6	4	4	0	21	8	39	2	5
7	4	4	0	21	8	2	39	0
8	4	4	0	21	8	2	39	1
9	4	0	4	21	8	2	39	2
10	4	0	4	21	8	2	39	3
11	4	0	4	8	21	2	39	4
12	4	0	4	8	2	21	39	0
13	0	4	4	8	2	21	39	1
14	0	4	4	8	2	21	39	2
15	0	4	4	8	2	21	39	3
16	0	4	4	2	8	21	39	0
17	0	4	4	2	8	21	39	1
18	0	4	4	2	8	21	39	2
19	0	4	2	4	8	21	39	0
20	0	4	2	4	8	21	39	1
21	0	2	4	4	8	21	39	0

Äußere Schleife: Wiederholt das Vertauschen der Nachbarn
Wiederhole bis zum vorletzten Element (mit Index i):

Die letzten „i“ Elemente sind schon korrekt sortiert.

Innere Schleife:
Wiederhole bis eine Position vor dem „i-t letzten“

Falls aktuelles Element der inneren Schleife größer als sein Nachfolger,
Tausche die beiden (mithilfe einer Variablen)

```
int i, j, temp;

for (i = 0; i < liste.length - 1; i++)
{
   // Die letzten i Elemente sind schon richtig sortiert.
   // Wiederhole bis eine Pos. vor dem i-t letzten
   for (j = 0; j < liste.length - i - 1; j++)
   {
      // Verleiche mit Nachfolger und tausche ggf.
      if (liste[j] > liste[j+1])
      {
         temp       = liste[j];
         liste[j]   = liste[j+1];
         liste[j+1] = temp;
      }
   }
}
```

Insertion Sort

Schritt	*Indizes*							*Index aktuell*	*Index korrekt*
	0	*1*	*2*	*3*	*4*	*5*	*6*		
1	4	39	4	0	21	8	11	0	0
2	4	39	4	0	21	8	11	1	1
3	4	39	4	0	21	8	11	2	1
4	4	4	39	0	21	8	11	3	0
5	0	4	4	39	21	8	11	4	3
6	0	4	4	21	39	8	11	5	3
7	0	4	4	8	21	39	11	6	4
8	0	4	4	8	11	21	39		

Äußere for-Schleife: Wiederholt das Einfügen der aktuellen Zahl nach links.
Wiederhole vom zweiten bis zum letzten Element:

> Setze Variable aktuell auf das aktuelle Element der äußeren Schleife
>
> Innere Schleife:
> Vom aktuellen Element der äußeren Schleife gehe schrittweise solange nach links,
> bis entweder eine Zahl gefunden ist, die kleiner als „aktuell“ ist
> oder der Anfang des Array erreicht ist.
>
> > Falls das aktuelle Element der inneren Schleife größer als „aktuell“ ist,
> > Verschiebe dieses Element eins nach rechts.
>
> An die Stelle, wo die innere Schleife aufgehört hat:
> Setze dort „aktuell“ ein.

```
int i, j, aktuell;

for (i = 1; i < liste.length; i++)
{
   aktuell = liste[i];
   j = i-1;

   // Gehe solange nach links, bis ein Element gefunden ist,
   // das kleiner oder gleich dem aktuellen Element ist
   // ODER bis das Ende der Liste erreicht ist
   while (j >= 0 && aktuell < liste[j])
   {
      // Schiebe jedes Element, das größer ist, nach rechts
      liste[j+1] = liste[j];
      j--;
   }
   // An dieser Stelle füge das aktuelle Element ein
   liste[j+1] = aktuell;
}
```