```c
/*
 ================================================================================
 Aufgabe     : Sortieren - Woche 7
 Autor       : Erik Kaufmann
 Matrikel    : 1390365
 Version     : 1.1
 ================================================================================
 */
#include <stdbool.h>
#include <stdio.h>
#include <stdlib.h>
#include "dhbwsortsimple.h"

 //Ab hier Aufgaben


 //Auf true setzen, damit BubbleSort getestet wird
bool BubbleSortImplemented() {
    return true;
}

//BubbleSort
void BubbleSortArray(Student_p* array, int count)
{
    Student_p* temp;

    //for (int i = 0; i < count; i++)
    //{
    //  printf("%s %d\n", array[i]->lastname, array[i]->matrnr);
    //}

    // count -1 --> array[n] --> [0]->[n-1] --> [n] knallt
    // Äußere Schleife zählt runter,
    for (int k = count - 1; k >= 0; k--)
    {
        for (int i = 1; i <= k; i++)
        {
            if (array[i - 1]->matrnr > array[i]->matrnr) // Prüfe, welche Mat-Nummer größer ist.
            {
                //swap
                temp = array[i];
                array[i] = array[i - 1];
                array[i - 1] = temp;
            }
        }
    }

    //printf("\n-------------------------------\n");
```

```
49
50      //for (int f = 0; f < count; f++)
51      //{
52      //  printf("%s %d\n", array[f]->lastname, array[f]->matrnr);
53      //}
54
55      return;
56  }
57
58
59  //Auf true setzen, damit SelectionSort getestet wird
60  bool SelectionSortImplemented() {
61      return true;
62  }
63
64
65  void Swap(Student_p* studA, Student_p* studB, int indexA, int
      indexB)
66  {
67      //printf("\nTausche [%d] %s %d mit [%d]%s %d\n", indexB,
          (*studB)->lastname, (*studB)->matrnr, indexA,(*studA)-
          >lastname, (*studA)->matrnr);
68      Student_p* temp = *studA;
69      *studA = *studB;
70      *studB = temp;
71  }
72
73  //SelectionSort
74  void SelectionSortArray(Student_p* array, int count)
75  {
76      //for (int i = 0; i < count; i++)
77      //{
78      //  printf("%s %d\n", array[i]->lastname, array[i]->matrnr);
79      //}
80
81      if (array != NULL)
82      {
83          int smallestElementIndex = 0;
84
85          bool swap = false;
86
87          for (int i = 0; i < count; i++) // Jedes Element wird
              geprüft
88          {
89              for (int k = i + 1; k < count; k++) // Und bis zum Ende
                  durchlaufen
90              {
91                  if (array[i]->matrnr > array[k]->matrnr)
92                  {
93                      // new smallest elemment and index
94                      array[i]->matrnr = array[k]->matrnr;
95                      smallestElementIndex = k;
96                      swap = true;
```

```
 97                  }
 98              }
 99
100              if (swap)
101              {
102                  //swap pointer
103                  Swap(&array[smallestElementIndex], &array[i],
                      smallestElementIndex, i);
104                  swap = false;
105
106                  //for (int f = 0; f < count; f++)
107                  //{
108                  //  printf("[%d] %s %d\n", f, array[f]->lastname,
                      array[f]->matrnr);
109                  //}
110              }
111          }
112      }
113
114      //printf("\n--------------------------------\n");
115
116      //for (int f = 0; f < count; f++)
117      //{
118      //  printf("%s %d\n", array[f]->lastname, array[f]->matrnr);
119      //}
120
121      return;
122  }
123
```