

Aufgabe 1 (Kat. B):

- a) Finden Sie alle Fehler im Code mit Nennung der Zeilennummer und Fehlerart und machen Sie einen Korrekturvorschlag
- b) Zeichnen Sie ein UML-Anwendungsfalldiagramm
- c) Setzen Sie den Code in ein Qt-Projekt um

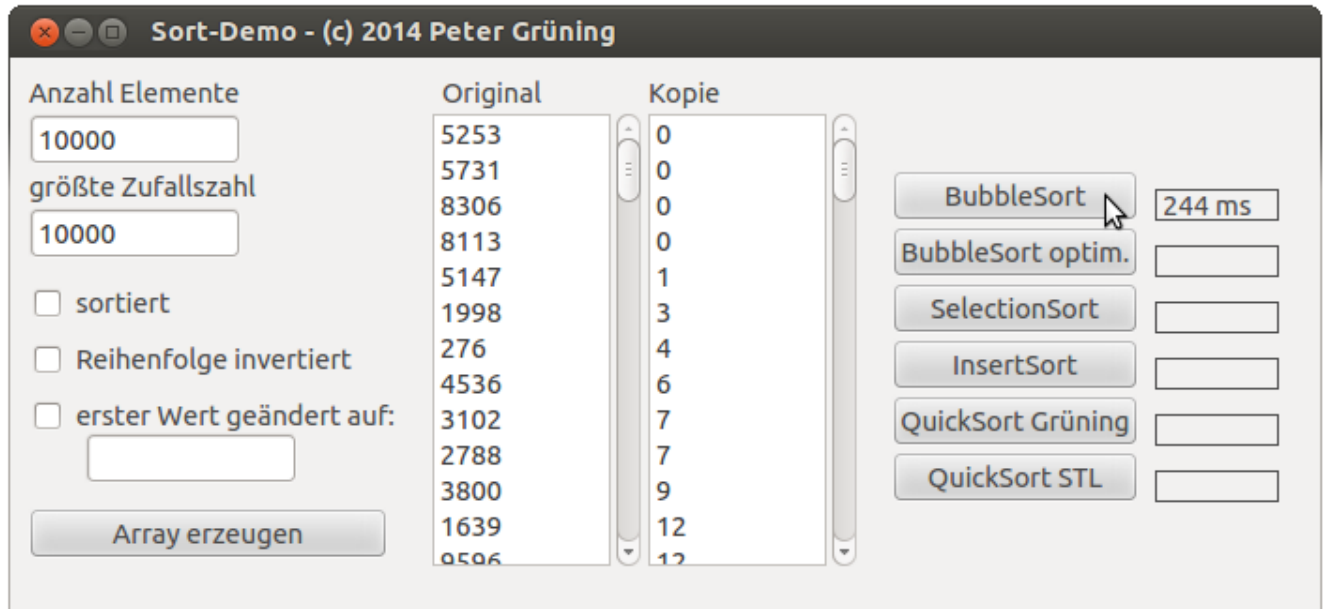
```
1 typedef int datentyp; //Datentyp des Arrays
2
3 datentyp* origArray;
4
5 void arrayAnzeigen(datentyp *array,int anz, QListWidget* lwAnzeige) {
6     lwAnzeige->clear();
7     lwAnzeige->setVisible(false); //sonst kostet die Anzeige zu viel Rechenzeit!
8     for (int i=0; i<=anz; i++) {
9         lwAnzeige->addItem(QString::number(array[i]));
10    }
11    lwAnzeige->setVisible(true);
12 }
13
14 void bubbleSort(datentyp *feld, int anz) //aufsteigendes Sortieren
15 {
16     datentyp *tmp;
17     for (int x=0; x<anz-1; x++) {
18         for (int i=0; i<=anz-1; i++) {
19             if (feld[i]<feld[i+1]) {
20                 tmp = feld[i];
21                 feld[i+1] = feld[i];
22                 feld[i+1] = tmp;
23             } //if
24         } //for i
25     } // for x
26 }
27
28 void FrmMain::on_btnBubbleSort_clicked()
29 {
30     this->setCursor(Qt::WaitCursor);
31     QTime time;
32     datentyp* tmpArray = new datentyp[anzElemente];
33     tmpArray = origArray;
34     time.start();
35     bubbleSort(tmpArray, anzElemente);
36     ui->lblZeitBubbleSort->setText(QString::number(time.elapsed())+ " ms");
37     arrayAnzeigen(tmpArray,anzElemente,ui->lwKopie);
38     delete tmpArray;
39     this->setCursor(Qt::ArrowCursor);
40 }
```

Aufgabe 2 (Kat. B):

Entwerfen (Struktogramm) und codieren Sie eine Funktion zum aufsteigenden Sortieren eines Arrays. Verwenden Sie folgende Sortieridee: Suche den kleinsten Wert - setze diesen auf die 0-te Stelle im Array - suche im Restarray wieder den kleinsten Wert - setze diesen an die 1-te Stelle im Array - usw. bis das Array sortiert ist.

Aufgabe 3 (Kat. C):

Entwickeln Sie ein Programm, um Sortieralgorithmen miteinander bzgl. ihrer Rechenzeiten zu vergleichen. Sortiert wird stets nur eine Kopie (!) des vorher erzeugten Arrays. Daher sind die Schaltflächen erst anklickbar, wenn das Originalarray erzeugt wurde. Die Rechenzeit ist in ms anzuzeigen (siehe Beispiel).



Anzahl Elemente	Original	Kopie	Algorithmus	Rechenzeit (ms)
10000	5253	0	BubbleSort	244 ms
größte Zufallszahl	5731	0	BubbleSort optim.	
<input type="checkbox"/> sortiert	8306	0	SelectionSort	
<input type="checkbox"/> Reihenfolge invertiert	8113	0	InsertSort	
<input type="checkbox"/> erster Wert geändert auf:	5147	1	QuickSort Grüning	
	1998	3	QuickSort STL	
	276	4		
	4536	6		
	3102	7		
	2788	7		
	3800	9		
	1639	12		
	9596	12		