Aufgabe 20

a) Prinzipiell funktioniert der Algorithmus so, dass bei jedem Durchlauf der äußeren Schleife das größte Element des noch-nicht-sortierten Teil-Arrays* in die rechteste Position getauscht wird.

Das Tauschen passiert so, dass man das erste (linke) Element des Teil-Arrays* mit dem rechten Nachbarn vergleicht. Dann geht es so weiter:

1) Wenn $a_i > a_{i+1}$: Tausche a_i und a_{i+1}. Vergleiche a_{i+1} nun mit a_{i+2} und führe dasselbe durch.
2) Wenn $a_i \leq a_{i+1}$: Tausche nichts. Vergleiche a_{i+1} und a_{i+2} und führe dasselbe durch.

Also kurz gesagt: "Ziehe" a_i solange nach rechts, bis ein a_j größer ist, danach "ziehe" a_j nach rechts usus.

Dies wird solange wiederholt, bis man keine Elemente mehr vertauscht, d.h. wenn das Array sortiert ist.

Ganz umgangssprachlich gesagt, bubblet das größte Element des Durchlaufs nach rechts bzw. oben.

b) Nach jedem Schleifendurchlauf ist das größte Element des noch nicht sortierten Arrays nach oben gebubblet. Dementsprechend ist das Array nach einer gewissen Zeit sortiert (siehe a). D.h. es können keine Elemente mehr vertauscht werden, wodurch der boolean 'vertauscht', welcher am Beginn des Schleifenrumpfs auf false gesetzt wird, auf false bleibt. Dadurch bricht die while-Schleife ab und der Algorithmus terminiert.

c) Der worst-case ist eine streng monoton fallende Folge. Beim ersten Durchlauf werden n-1 Vertauschungen durchgeführt, beim zweiten n-2 ...

Es folgt:

$$T(n) = \sum_{i=1}^{n-1} (n-i) = (n-1) + \ldots + (n-n+1) + (n-n) = \sum_{i=1}^{n-1} i = \frac{n^2-n}{2} \in \mathcal{O}(n^2)$$