## A.2:

Argumentieren Sie, weshalb der Algorithmus korrekt ist. Weisen Sie dazu nach, dass für i ≥ 1 die Schleifeninvariante

: Vor dem i–ten Durchlauf durch den Rumpf der for-Schleife besteht aus den ursprünglichen Elementen , aber in sortierter Reihenfolge.

gilt. Zerlegen Sie Ihre Argumentation in folgende drei Schritte:

**1. Initialisierung:** Es gilt vor der ersten Ausführung des Rumpfes der for-Schleife.

* Aussage ist trivial, da nur ein Element enthält

**2. Invarianz:** Falls für ein i ≥ 1 die Invariante gilt, dann gilt auch .

* Wenn Invariante gilt, ist der Teil der Liste vor dem i-ten Element sortiert.
* Bei ist das i-te Element bereits sortiert, da gilt

Nach dem i-ten Durchlauf:

* ist nach Voraussetzung sortiert
* ist sortiert, da Elemente durch Einfügen von nur nach rechts verschoben wurden 🡪 Reihenfolge ist gleichgeblieben und Elemente waren vorher sortiert
* Bei Element ist Schleife abgebrochen, also muss negierte Schleifenbedingung gelten 🡪 muss also größer/gleich
* Schleife ist bis zu diesem Punkt gelaufen 🡪 also muss While-Bedingung gelten und somit muss

**3. Terminierung:** Wenn die for-Schleife terminiert , können wir aus auf die Korrektheit des Algorithmus schließen.

* Korrektheit wurde durch vollständige Induktion bewiesen
* Schritt 1 stellt Induktionsanfang dar
* Schritt 2 stellt Induktionsschritt da
* Somit gilt für alle
* Wenn for-Schleife terminiert, dann gilt i=m
* Nach Schritt 2 gilt und somit 🡪 ist sortiert
* Nach Terminierung der for-Schleife hat i den Wert 🡪 wegen Syntax aus ThI (for-Schleife als While-Schleife realisiert)