Aufgabe 2

a)

Zu zeigen ist, dass die Laufzeit von `Build-Heap` für eine Liste mit n Einträgen in $\mathcal{O}(n)$ liegt.

Hierzu soll zunächst gezeigt werden, dass die Laufzeit von `Max-Heapify(i)` in $\mathcal{O}(l)$ liegt, wobei l die Anzahl der Ebenen im Heap beschreibt, welche unterhalb des Knotens i liegen.

Zu beobachten ist, dass jeder Aufruf von `Max-Heapify(i)` höchstens einen weiteren rekursiven Aufruf tätigt, welcher mit genau einem Kindknoten von i parametrisiert ist. Dieser Kindknoten liegt im Binärbaum per Definition eine Ebene unter i. Da der ursprüngliche Aufruf von `Max-Heapify` auf Ebene e_i passiert, können alle Ebenen oberhalb von e_i ignoriert werden - daraus folgt, dass höchstens $l = h - e_i$ rekursive Aufrufe getätigt werden. Da die restlichen Operationen von `Max-Heapify` konstante Laufzeit benötigen, liegt die Gesamtlaufzeit in $\mathcal{O}(l)$.

Sei o.B.d.A. die Höhe des Baums $h = 2^n$.

Wie in der letzten Übung gezeigt wurde, liegen in einem vollständigen Binärbaum 2^j Knoten in jeder Ebene j.
Von unten nach oben betrachtet, halbiert sich damit die Anzahl der Knoten auf jeder Ebene, und pro Knoten (welcher kein Blatt ist, also ab der zweiten Ebene von unten) wird einmal `Max-Heapify` aufgerufen, welcher im schlimmsten Fall pro Ebene unterhalb des Knotens einen rekursiven Aufruf tätigt.

Damit ergibt sich eine Laufzeit L von

$$
\begin{aligned}
L &\leq \sum_{i=1}^{h-1} 2^h \cdot i \cdot 0,5^i \\
&= 2^h \sum_{i=1}^{h-1} i \cdot 0,5^i \\
&< 2^h \cdot 0,5/(1-0,5)^2 \\
&= 2^h \cdot 2 \\
&= 2^{\log_2 n} \cdot 2 \\
&= 2n \in \mathcal{O}(n)
\end{aligned}
$$

b)

Lemma:

> Um ein Element in eine Priority-Queue der Länge n einzufügen, benötigt man im schlimmsten Fall $\lceil \log_2 n \rceil$ Schritte.

Beweis:
Eine Priority-Queue ist ein Binärbaum, und ein Binärbaum hat Höhe $h = \lceil \log_2 n \rceil$. Die Operation `Insert` fügt das neue Element an das Ende der Queue, d.h. in die unterste Ebene des Baums, und ruft dann `Increase-Key` auf. Bei `Increase-Key` wird der Knoten in jedem Schritt eine Ebene nach oben geschoben, welches höchstens h-mal passieren kann. Dadurch werden im Worst-Case mindestens $\lceil \log_2 n \rceil$ benötigt.

Zu zeigen:

> Einen Heap mit der `Insert`-Operation einer Priority-Queue aufzubauen, benötig im schlimmsten Fall eine Laufzeit von $\Omega(n \log n)$.

Aus o.g. Lemma ergibt sich für eine Folge von n `Insert`-Operationen auf einen leeren Heap im Worst Case folgende Laufzeit L:

$$\begin{aligned} L &\geq \sum_{i=1}^{n} \lceil \log_2 i \rceil \\ &> \log_2 \left(\prod_{i=1}^{n} i \right) \\ &= \log_2(n!) = \Theta(n \log n) \\ &\Rightarrow L \in \Omega(n \log n) \end{aligned}$$

□