11.3

Idee: Wir benutzen dynamische Programmierung, um vom 33. Stock alle möglichen „rauf“ und „runter“ Optionen auszuprobieren. Wir benutzen ein Array A, um die bereits besuchten Felder zu speichern Dabei speichern wir einerseits, wie viele Schritte wir bereits gegangen sind (also wie weit weg wir von der 33 sind) und, ob wir um zur 33 zu gelangen als nächstes rauf, oder runter müssen.
Dabei wollen wir, um nicht einfach alle Optionen überprüfen zu müssen (und Gebrauch von der DP zu machen) folgendes tun :

1. Ist das Feld an der Stelle $A[i]$ leer, so schreiben wir unsere Distanz und Richtung einfach rein.
2. Andernfalls überprüfen wir, ob unsere aktuelle Distanz $d_i + 1 <$ als die Distanz d im bereits beschriebenen Array. Falls ja, überschreiben wir das Array. Ansonsten hören wir an der Stelle auf.

Wir terminieren, falls wir das im Array eine Distanz für $A[31]$ haben *(31, da i ab 0 zählt)*. Zudem betrachten wir, ob wir out of bounds landen (Wir haben nur 50 Stockwerke).

Durch diese Vorgehensweise überprüfen wir nicht alle möglichen Teilbäume und nutzen bereits errechnete Ergebnisse.

Für $i = 3$ sähe A also wie folgt aus („rauf“ = u, „runter“ = d):

$A[10]$	$A[16]$	$A[21]$	$A[27]$	$A[32]$	$A[33]$	$A[38]$	$A[44]$
2,u	3,u	1,u	2,d	0,-	3,u	1,d	2,d
	3,d		2,u		3,d		

Die Doppelungen sollen hier nur die Arbeitsweise des Algorithmus verdeutlichen. Bei einem echten Durchlauf würde dieser die erste gewählte Option stehen lassen.

Mit diesem Algorithmus kommen wir auf die Lösung, dass wir 14 Operationen brauchen, um vom 32. Stock in den 33. zu gelangen.