Lösungsidee

Damit die Bauten der Baulwürfe gefunden werden können, muss die gesamte Karte nach Planquadraten mit Hügeln in folgender Form durchsucht werden:

XXX

X X

X X

XXX

Die Lösungsidee ist, dass zuerst die komplette Karte in einen zweidimensionalen Array geladen wird. Somit kann man jedes Planquadrat einzeln betrachtet und auf einen „X“ bzw. einen Hügel überprüfen. Dann kann jeder Punkt mit den Koordinaten X und Y überprüft werden. Um auf ein Bau zu überprüfen, werden auch alle anderen Punkte überprüft. Das lässt sich einfach mit einer Versetzung realisieren. So kann man bspw. Den Punkt Rechts von dem Ursprünglichen Planquadrat mit den Koordinaten X+1 und Y überprüfen. Dasselbe funktioniert auch in Y-Richtung. So kann man das Planquadrat unterhalb mit den Koordinaten X und Y+1 überprüfen. So kann man jetzt alle Planquadrate mit einer Versetzung bis zu X+2 und Y+3 prüfen. Sollten dann alle Hügel passend positioniert sein, so liegt ein Bau vor.

Nun kann jeder einzelne Punkt der Karte auf ein Bau überprüft werden und für jeden gefunden Bau ein Zähler erhöht werden.

Umsetzung

Die Lösungsidee wird in Python umgesetzt. Das sys Modul wird geladen, dass man das Skript mit Argumenten starten kann. So lässt sich schon beim Start die zu durchsuchende Karte auswählen.

Vorerst soll die Kartendatei in ein Array geladen werden, wobei ein Element des Arrays eine Zeile ist. Das erste und Element werden jeweils einer seperaten Breiten-und Höhenvariable zugeordnet und danach aus dem Array entfernt. Nun befinden sich nur noch rohe Kartendaten in diesem Array.

Nun wird ein zweidimensionaler Array erstellt. Das geschieht in einer for-Schleife, welche für jedes Element der rohen Kartendaten einmal durchläuft. Nun wird eine einzelne Y-Achse aus den rohen Kartendaten in eine Variable kopiert. Aus dieser Variable wird der Formatierungsbedingte Zeilenumbruch „\n“ entfernt und schließlich werden alle einzelnen Buchstaben zu einem Array. Der neu entstandene Array wird nun der Karte hinzugefügt. Somit wurde nun ein zweidimensionaler Array mit X- und Y-Koordinaten erstellt:

```
for i in range(len(raw_input)):
y_raw = raw_input[i]
y_raw = list(y_raw[:-1])
map.append(y_raw)
```

Um einzelne Punkte auf einen Hügel zu überprüfen, wird eine eigenständige Funktion erstellt, welche zuerst die komplette Y-Achse aus der Karte lädt und anschließend den Punkt mit der X-Koordinate auf ein „X“ überprüft und entsprechend „True“ oder „False“ zurückgibt.

Damit nun ein Planquadrat und anliegende Planquadrate auf ein Bau überprüft werden können, gibt es eine if-Bedingung in einer eigenständigen Funktion, welche anhand der gegeben X- und Y-Koordinaten den Punkt und alle anliegenden Punkte, welche für einen Bau notwendig sind, überprüft. Hierbei wird mit einer Verschiebung gearbeitet und alle Punkte bis X+2 und Y+3 betrachtet:

if(check(dx,dy)==check(dx+1,dy)==check(dx+2,dy)==check(dx,dy+1)==check(dx,dy+2)==check(dx,dy+3)==check(dx+2,dy+1)==check(dx+2,dy+2)==check(dx+2,dy+3)==check(dx+1,dy+3)==True) and (check(dx+1,dy+1)==check(dx+1,dy+2)==False):

Dass nun die gesamte Karte überprüft werden kann, gibt es zwei for-Schleifen, wobei eine in der anderen positioniert ist. Es wird für jeden Punkt der X- Achse (außer den letzten zwei), für jeden Punkt der Y-Achse des X-Punkts (außer die letzten drei), eine Überprüfung auf einen Bau gestartet. Da ein Bau 3*4 Planquadrate groß ist, findet die Überprüfung in der X-Achse nur bis gesamtbreite-2 und auf der Y-Achse nur bis gesamthöhe-3 statt. Das liegt daran, dass das die letzen möglichen Punkte sind, wo ein Bau existieren könnte. Ohne die Reduzierung würde das Programm wegen der Verschiebung bei der Bauüberprüfung Punkte außerhalb der Karte prüfen und einen Error hervorrufen.

Am Ende des Programmes wird die Anzahl an gefunden Bauten ausgegeben.

Beispiele

Um nun eine Karte nach Bauten zu durchsuchen, starten wir es mit dem Namen der Karte:

python Junioraufgabe2.py karte0.txt

Führen wir das Programm nun so aus, erhalten wir folgendes:

Anzahl der Bauten: 3

Wir erhalten also das korrekte Ergebnis an Bauten innerhalb der Karte.

Führen wir das Programm erneut aus und wählen wir dabei die Beispielkarte 6, erhalten wir folgendes Ergebnis:

Anzahl der Bauten: 559

Das kann sich einfach überprüfen lassen. Die Karte ist nämlich wie folgt angeordnert:

XXXXXXXXX

X XX XX X

X XX XX X

XXXXXXXXX

Und das verhält sich auch in y-Richtung so. Mit der gegebene Länge- und Breite der Karte lässt sich die Anzahl der theoretisch vorhanden Bauten berechnen: (129/3)*(52/4) = 559. Somit lässt sich das vom Programm gezeigte Ergebnis bestätigen.