Jugendwettbewerb Aufgabe 2: Kacheln

Team-ID: 00xxx

Vor- und Nachname: xxx xxx

November 2019

Inhaltsverzeichnis

Lösungsidee

Die Karte wird in einem 2-Dimensionalen Array gespeichert. Die Sternchen werden von oben links nach unten rechts ersetzt. Bei diesem Vorgehen kann man davon ausgehen, dass links und oberhalb des Sternchens immer kein Sternchen ist. Daraus ergibt sich folgender Algorithmus zum Ersetzen der Sternchen:

- `FALLS oben linke Spalte auf Kachel UND NICHT ganz linke Spalte der Karte, DANN nimm Symbol links vom Sternchen`
- `SONST FALLS oberen Zeile auf Kachel UND NICHT oberste Zeile der Karte, DANN nimm Symbol über  Sternchen`
- `SONST FALLS rechte Spalte auf Kachel UND NICHT rechte Spalte der Karte und rechts vom Sternchen ist kein Sternchen, DANN nimm Symbol rechts vom Sternchen`
- `SONST FALLS untere Zeile der Kachel und nicht unterste Zeile der Karte und unter dem Sternchen ist kein Sternchen, DANN nimm Symbol unter dem Sternchen`
- `SONST Symbol egal`

Umsetzung

Programmiersprache

Als Programmiersprache wurde Python 3.7.4 verwendet, da es sehr gute Funktionen zum Operieren mit Dateien, als auch auch gute Möglichkeiten zum Arbeiten mit 2D-Arrays hat.

Einlesen

Der Inhalt der Datei wird in die globalen Variablen `rows`, `cols` und `map` eingelesen. Es wird das kompakte Format der Karten eingelesen.

Ausgeben

Die Ausgabe erfolgt auf das Terminal, es gibt kompakte, aber auch spacy Ausgaben.

Beispiele

Ausgabe für Eingabedatei `map_compact.txt`

```
2
2

1 0  0 0
0 1  1 0

0 1  1 0
0 1  1 1
```

Ausgabe für Eingabedatei `map_compact2.txt`

```
6
7

1 0  0 1  1 0  0 1  1 1  1 0  0 1
0 1  1 0  0 0  0 0  0 1  1 0  0 1

0 1  1 0  0 0  0 0  0 1  1 0  0 1
1 1  1 0  0 0  0 0  0 1  1 1  1 1

1 1  1 0  0 0  0 0  0 1  1 1  1 1
0 0  0 1  1 0  0 0  0 1  1 1  1 0

0 0  0 1  1 0  0 0  0 1  1 1  1 0
0 1  1 1  1 1  1 1  1 1  1 0  0 0

0 1  1 1  1 1  1 1  1 1  1 0  0 0
1 1  1 0  0 0  0 0  0 1  1 0  0 1

1 1  1 0  0 0  0 0  0 1  1 0  0 1
0 1  1 1  1 0  0 0  0 0  0 0  0 1
```

Ausgabe für Eingabedatei `map_compact3.txt`

```
4
7

1 0  0 0  0 1  1 1  1 0  0 0  0 1
1 1  1 1  1 0  0 1  1 1  1 0  0 0
```

```
11 11 10 01 11 10 00
00 01 10 00 00 01 11

00 01 10 00 00 01 11
11 10 01 10 00 00 01

11 10 01 10 00 00 01
11 11 11 10 00 01 11
```

Ausgabe für Eingabedatei `map_compact4.txt`

```
9
16

11 10 00 00 01 10 01 11 10 01 11 10 00 01 11 11
00 01 10 01 11 10 00 01 11 11 10 00 00 01 10 01

00 01 10 01 11 10 00 01 11 11 10 00 00 01 10 01
01 11 11 11 11 11 10 00 00 00 01 10 01 11 10 00

01 11 11 11 11 11 10 00 00 00 01 10 01 11 10 00
11 10 00 00 00 00 00 01 10 01 11 10 01 11 10 00

11 10 00 00 00 00 00 01 10 01 11 10 01 11 10 00
00 00 00 00 00 01 11 11 11 11 11 11 10 00 00 00

00 00 00 00 00 01 11 11 11 11 11 11 10 00 00 00
00 00 00 00 01 11 11 11 11 10 00 00 01 11 10 01

00 00 00 00 01 11 11 11 11 10 00 00 01 11 10 01
00 01 10 01 11 11 10 00 01 10 00 01 11 10 00 00

00 01 10 01 11 11 10 00 01 10 00 01 11 10 00 00
00 01 10 01 10 00 00 00 00 00 01 11 11 11 11 11

00 01 10 01 10 00 00 00 00 00 01 11 11 11 11 11
01 11 10 00 00 00 00 00 00 01 11 11 10 00 00 00

01 11 10 00 00 00 00 00 00 01 11 11 10 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Ausgabe für Eingabedatei `map_compact5.txt`

```
5
5
```

```
11 11 11 11 11
10 00 00 01 11

10 00 00 01 11
00 01 10 00 00

00 01 10 10 00
00 00 00 00 00

00 00 01 10 00
01 11 11 11 11

01 11 11 11 11
11 10 00 01 11
```

Quellcode

```
def print_map():
 print(rows)
 print(cols)
 for row in range(len(map)):
   line = ""
   for col in range(len(map[row]) - 1):
     line = line + map[row][col]
   print(line)

def print_map_spacy():
 print(rows)
 print(cols)
 print("")
 for row in range(len(map)):
   line = ""
   for col in range(len(map[row]) - 1):
     line = line + map[row][col] + " "
     if not is_left_col_on_tile(col):
       line = line + "  "
   print(line)
   if not is_upper_row_on_tile(row):
     print("")

def is_left_col_on_tile(col):
 return col % 2 == 0
```

```
def is_upper_row_on_tile(row):
 return row % 2 == 0

def is_left_col_on_map(col):
 return col == 0

def is_upper_row_on_map(row):
 return row == 0

def is_right_col_on_map(col):
 return col == 2 * cols - 1

def is_lowest_row_on_map(row):
 return row == 2 * rows - 1

def set_symbol_in_map(row, col, symbol):
 map[row] = map[row][:col] + symbol + map[row][col + 1:]

#file = open("map_compact.txt")
#file = open("map_compact2.txt")
#file = open("map_compact3.txt")
#file = open("map_compact4.txt")
file = open("map_compact5.txt")

rows = int(file.readline())
cols = int(file.readline())
map = file.readlines()

print("Eingabe:")
print_map()

for row in range(2*rows):
 for col in range(2*cols):
   symbol = map[row][col]
   if symbol == "*":
     if is_left_col_on_tile(col) and not is_left_col_on_map(col):
       set_symbol_in_map(row, col, map[row][col - 1])
     elif is_upper_row_on_tile(row) and not is_upper_row_on_map(row):
       set_symbol_in_map(row, col, map[row - 1][col])
```

```
    elif not is_left_col_on_tile(col) and not is_right_col_on_map(col)
and "*" != map[row][col + 1]:
      set_symbol_in_map(row, col, map[row][col + 1])
    elif not is_upper_row_on_tile(row) and not
is_lowest_row_on_map(row) and "*" != map[row + 1][col]:
      set_symbol_in_map(row, col, map[row + 1][col])
    else:
      set_symbol_in_map(row, col, "0")

print("Ausgabe:")
print_map()

print("Ausgabe Spacy:")
print_map_spacy()
```

Fehler und Anmerkungen

Die Karte 5, funktioniert nicht, da die Eingabe nicht validiert wird. Das Programm müsste um eine Validierung der Eingabe ergänzt werden.