resitev

January 28, 2024

0.1 Rešitev

Najprej preberimo podatke; ločimo jih na koordinate in navodila za prepogibanje.

[1]:
```
import numpy as np

xy, instructions = open("example.txt").read().split("\n\n")
```

Spisek koordinat razdelimo na vrstice (`xy.splitlines()`). Za vsako vrstico (`for v in xy.splitlines()`) sestavimo seznam, ki vsebuje števila iz vrstice, ki jo razbijemo z vejico kot ločilom (`[int(x) for x in v.split(",")]`). Ta seznam seznamov spremenimo v tabelo

[2]:
```
xy = np.array([[int(x) for x in v.split(",")]
               for v in xy.splitlines()])

xy
```

[2]:
```
array([[ 6, 10],
       [ 0, 14],
       [ 9, 10],
       [ 0,  3],
       [10,  4],
       [ 4, 11],
       [ 6,  0],
       [ 6, 12],
       [ 4,  1],
       [ 0, 13],
       [10, 12],
       [ 3,  4],
       [ 3,  0],
       [ 8,  4],
       [ 1, 10],
       [ 2, 14],
       [ 8, 10],
       [ 9,  0]])
```

V prvem stolpcu so koordinate `x`, v drugem `y`. Poiskati moramo največjo in prišteti 1 (ker začnemo z 0); bo bodo dimenzije naše tabele.

[3]:
```
np.max(xy, axis=0) + 1
```

[3]:
```
array([11, 15])
```

Imeli bomo 15 vrstic in 11 stolpcev, torej moramo to še obrniti. Tako dobimo obliko, ki jo podamo `np.zeros`. Recimo, da bomo uporabljali tabelo `bool`-ov.

[4]:
```
dots = np.zeros(np.max(xy, axis=0)[::-1] + 1, dtype=bool)
```

Zdaj pa na mesta, ki ju določajo koordinate `y` in `x` (dobimo jih z `xy[:, 1]` in `xy[:, 0]`) zapišemo enice.

[5]:
```
dots[xy[:, 1], xy[:, 0]] = 1
```

Na hitro izpišimo, da vidimo, kaj imamo: gremo čez vse vrstice (`for v in dots`), za vsako gremo čez vse vrednosti `for i in v`, in to vrednost, spremenjeno v `int`, uporabimo kot indeks v niz `".#"`; kadar je `i` enak `0`, bomo izpisali `.`, kadar je `1`, pa `#`. Znakce združimo z `"".join`, vrstice pa z `"\n".join`.

[6]:
```
print("\n".join("".join(".#"[int(i)] for i in v) for v in dots))
```

```
…#..#..#.
…#…
…
#…
…#…#.#
…
…
…
…
…
.#…#.##.
…#…
…#…#
#…
#.#…
```

Tole je enako kot v opisu v nalogi, torej pravilno. Celotno branje je torej, še enkrat

[7]:
```
xy, instructions = open("example.txt").read().split("\n\n")

xy = np.array([[int(x) for x in v.split(",")] for v in xy.splitlines()])
dots = np.zeros(np.max(xy, axis=0)[::-1] + 1, dtype=bool)
dots[xy[:, 1], xy[:, 0]] = 1
```

Še nekoliko preprosteje je, če koordinate vrtimo kar brez `numpy`-ja, za kar pa je potrebno znati transponirati podatke z `zip`.

[8]:
```
xy, instructions = open("example.txt").read().split("\n\n")

x, y = zip(*(map(int, v.split(",")) for v in xy.splitlines()))
dots = np.zeros((max(y) + 1, max(x) + 1), dtype=bool)
dots[y, x] = 1
```

Tako ali drugače torej dobimo začetni razpored pik.

Zdaj gremo čez navodila. Če vrstica navodil vsebuje `y`, prepogibamo tako, da seštevamo spodnje in zgornje vrstice, sicer seštevamo levi in desni del. Če sta `h` in `w` trenutni dimenziji (`h, w = dots.shape`), je `dots[:h // 2]` je zgornja polovica vrstic, `dots[:h // 2:-1]` pa prezrcaljena spodnja polovica. Podobno je `dots = dots[:, :w // 2]` leva polovica, `dots[:, :w // 2:-1]` pa prezrcaljena desna. Tidve ali onidve polovici seštejemo in tako dobimo novo matriko pik.

[9]:
```
for instruction in instructions.splitlines():
    h, w = dots.shape
    if "y" in instruction:
        dots = dots[:h // 2] + dots[:h // 2:-1]
    else:
        dots = dots[:, :w // 2] + dots[:, :w // 2:-1]

print("\n".join("".join(".#"[int(i)] for i in v) for v in dots))
```

```
#####
#…#
#…#
#…#
#####
…
…
```

In to je to. Če reč poženemo na pravih podatkih (recimo mojih), dobimo nekaj črk, ki jih je bilo potrebno vnesti. Mimogrede poskrbimo le še za prvi del naloge, ki je hotel, da izpišemo število pik po prvem koraku prepogibanja. Za to bomo šteli korake in če je števec enak `0`, izpisali vsoto tabele.

[10]:
```
xy, instructions = open("input.txt").read().split("\n\n")

xy = np.array([[int(x) for x in v.split(",")] for v in xy.splitlines()])
dots = np.zeros(np.max(xy, axis=0)[::-1] + 1, dtype=bool)
dots[xy[:, 1], xy[:, 0]] = 1

for i, instruction in enumerate(instructions.splitlines()):
    h, w = dots.shape
    if "x" in instruction:
        dots = dots[:, :w // 2] + dots[:, :w // 2:-1]
    else:
        dots = dots[:h // 2] + dots[:h // 2:-1]
    if not i:
```

```
        print(np.sum(dots))

print("\n".join("".join(".#"[int(i)] for i in v) for v in dots))
```

```
712
###..#…#..#.####…##.###…##.####.
#..#.#…#..#.#…#.#..#…#.#…
###..#…####.###…#.#..#…#.###..
#..#.#…#..#.#…#.###…#.#…
#..#.#…#..#.#…#..#.#…#..#.#…
###..####.#..#.#…##..#…##..#…
```