Podan je slovar `postaje`, katerega ključi so imena postaj Biciklja, vrednosti pa slovarji z zemljepisnimi koordinatami postajališč, kapaciteto (število stojal) in trenutnim številom koles na postaji. (Glej slovar v testih.)

Podana je tudi funkcija `razdalja_geo(lat1, lon1, lat2, lon2)`, ki za podani par zemljepisnih koordinat vrne razdaljo med točkama v kilometrih. (Funkcija deluje pravilno samo za kratke razdalje.)

0. Razdalje

Napišite funkcijo `razdalja(postaja1, postaja2)`, ki prejme imeni postaj in vrne razdaljo med njima. (Funkcija je vredna 0 točk. Pišete jo za ogrevanje in rabo v naslednjih nalogah.)

1. Najbližje 3

Avtor izpita ni uporabnik Biciklja (enkrat je bilo dovolj), ve pa, da na določeni postaji včasih ne moreš dobiti ali oddati kolesa, zato se je potrebno ozreti po bližnjih postajah. Napišite funkcijo `najblizje3(postaja)`, ki za podano postajo poišče najbližje tri postaje in vrne seznam (treh) parov (razdalja, ime). Seznam naj bo urejen po razdaljah.

Klic `najblizje3("ŽIVALSKI VRT")` vrne `[(0.73687236710574, 'VIŠKO POLJE'), (0.9639016462683163, 'TEHNOLOŠKI PARK'), (1.0760703913614853, 'SOSESKA NOVO BRDO')]`.

2. Izolirana postaja

Napišite funkcijo `izolirana()`, ki poišče in vrne ime postaje, ki je najbolj oddaljena od postaje, ki ji je najbližja. Za te konkretne postaje je to Zalog, vendar mora biti funkcija napisana splošno.

3. Tabela postaj

Napišite funkcijo `zapisi(ime_datoteke)`, ki v podano datoteko zapiše imena postaj (na 40 mest), ki ji sledi število razpoložljivih koles in število mest. Zapis mora biti urejen po abecedi imen postaj (šumniki pa bodo na koncu; uporabite običajno urejanje v Pythonu) in v obliki, ki jo kaže okvirček.

```
KONGRESNI TRG-ŠUBIČEVA ULICA              9/20
KOPALIŠČE ILIRIJA                         0/20
KOPALIŠČE KOLEZIJA                       19/20
KOPRSKA ULICA                             3/8
KOSEŠKI BAJER                            14/20
```

4. Samo dol

```
[[1, 2, 1, 0, 3, 0],
 [0, 3, 7, 0, 5, 0],
 [0, 1, 0, 2, 3, 0],
 [0, 3, 0, 0, 4, 2],
 [0, 1, 5, 0, 0, 3],
 [0, 0, 5, 1, 3, 0]]
```

Biciklji so s svojo neudobnostjo primerni bolj kot ne za vožnjo navzdol. Tabela (seznam seznamov) na sliki predstavlja nadmorske višine nekih točk na neki pravokotni mreži. Ker kolesar vozi le navzdol, lahko gre s polja [1][1] (trojka) le v celico v eni vrstici višje (na 2), levo (na 0) ali vrstico nižje (na 1), ne pa desno, ker je tam 7. Po diagonalah ne vozi.

Napišite funkcijo `spust_na_0(matrika, v, s)`, ki za podani v (vrstica) in s (stoplec) pove, koliko ničel je dosegljivih s tega polja. Če je možno do neke ničle priti na več načinov, jo štejte večkrat, ker je kolesar neumen in ne opazi, kadar pride do istega cilja, vendar po drugi poti. (Naloga je takšna zato, da je lažja, ne težja. Pri "normalni" rešitvi boste večkratne poti že "ponesreči" šteli večkrat.)

Z omenjene trojke lahko pride do petih ničel: na **eno** neposredno, do **dveh** prek spodnje enke, če gre gor na dvojko, pa lahko gre od ondod levo ali desno na enki v prvi vrstici in potem na ničli zraven njiju. Ničlo v prvi vrstici štejemo dvakrat, ker jo dosežemo na dva načina.

5. Kontroler

Napišite razred `Kontroler`.

- Konstruktor ne prejme argumentov. Shrani lahko, kar želite. Recimo podatke iz `postaje`. Ali pa tudi ne.
- `stanje(postaja)` vrne število razpoložljivih koles na podani postaji
- `odpelji(postaja)` vrne `True`, če je na postaji kako kolo in `False`, če ga ni. Če je, si zapomni, da je na postaji eno kolo manj.
- `vrni(postaja)` vrne `True`, če je na postaji kako prosto mesto za kolo in `False`, če ga ni. Če je, si zapomni, da je na postaji eno kolo več.

Pomembno: razred ne sme spreminjati vrednosti v (globalnem) slovarju `postaje`!