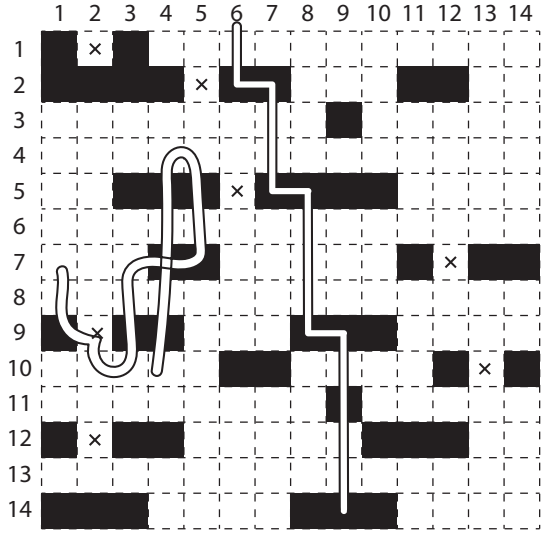
Izpit iz predmeta Programiranje 1, 7. februar 2023 **Čas pisanja: ~~60 minut~~ → 90 minut**

Dovoljena je uporaba literature (Učilnica in vse, kar prinesete s seboj), prepovedana pa je vsakršna komunikacija.

1. Prehodi

Prehodi so prosta polja, ki imajo na sosednjem levem in desnem polju oviro. Na sliki so označena s križci. Napiši funkcijo `prehodi(ovire)`, ki vrne množico koordinat prehodov v obliki parov `(y, x)`.



Slika kaže razpored ovir v testih. Funkcije morajo delovati za poljuben razpored ovir (predpostaviti pa smeš, da se ovire ne prekrivajo) in velikost steze.

Ovire so podane kot seznam terk (y, x0, x1), kjer je y številka vrstice, x0 in x1 pa skrajni levi in desni stolpec ovire. Seznam ovir ni urejen ne po vrsticah ne kakorkoli drugače.

2. Nadloga

Nek kolesar obvlada veščino voženj po ovirah. Napiši funkcijo `nadlezna_ovira(ovire, y, x, pot)`, ki prejme ovire, začetne koordinate kolesarja in pot v obliki niza znakov <, >, ^ in v. Vrne naj oviro, na kateri je kolesar preživel največ "potez".

Klic `nadlezna_ovira(ovire, 10, 4, "^^^^^^>vvv<<vvv<^<^^")`, ki opisuje (levo) pot na sliki, vrne `(7, 4, 5)`, saj je bil kolesar na njej trikrat (najprej na poti gor, potem pa še dvakrat zapored, ko je šel levo prek ovire).

Pot se nikoli ne začne na oviri, lahko pa se na njej konča.

Če kolesar ne naleti na nobeno oviro, naj funkcija vrne `None`. Če je enako nadležnih več ovir, lahko vrne poljubno izmed njih.

3. Odstranjevanje

Po (pre)dolgih letih so zdaj končno zamenjali vodjo MOL-ovega Oddelka za gospodarske dejavnosti in motorni promet. (To ni šala, kar preverite!)

Napiši funkcijo `odstrani(ovire, stolpci)`, ki prejme seznam ovir in številk stolpcev. Iz seznama naj odstrani vse ovire, ki zapirajo kateregakoli od podanih stolpcev. Funkcija ne sme vrniti ničesar: spreminjati mora podani seznam.

4. Žaba

Tako kot veverice si tudi žabe ne upajo stopiti na kolesarsko pot, temveč se premikajo zgolj po ovirah. Žaba lahko hodi levo in desno po oviri, skače pa vedno samo naravnost navgor. (Zaradi močnega vetra v to smer.)

Žaba želi priti na vrh poti in pri tem delati čim krajše skoke. Za pot z ovire (14, 8, 10) bo morala narediti vsaj en skok dolžine 3; primer možne takšne poti je na sliki.

Napiši funkcijo `najdaljsi_skok(ovire, ovira)`, ki prejme seznam ovir in začetno oviro (kot terko `(y, x0, x1)`), ter vrne najdaljši skok, ki ga bo morala opraviti žaba, če izbere pot, pri kateri je najdaljši skok čim krajši.

V pomoč: v datoteki s testi je funkcija `naprej(ovire, ovira)`, ki vrne vse ovire, na katere je možno skočiti z `ovira`.

Pazi (1): Med deveto in peto vrstico so tri prazne vrste, zato se šteje, da je skok med njima dolg 3 (ne 4!).

Pazi (2): žaba naj ne skače kar brezglavo na najbližjo oviro! Če iz (14, 8, 10) skoči na (12, 10, 12) in potem na (10, 12, 12), se zafrkne, saj mora potem z (10, 12, 12) narediti skok dolžine 7!

5. Anarhist

Napiši razred `Kolesar`.

- Konstruktor prejme argumente `(ovire, y, x)`, to je seznam ovir in začetno točko. Začetna točka gotovo ni na oviri. Razred sme v kasnejših metodah spreminjati podani seznam.
- Metoda `premik(smer)` ga premakne s trenutne točke za eno polje v podani smeri (<, >, ^ ali v). Če je na tistem polju slučajno ovira, jo kolesar zdrobi v sončni prah. Potem te ovire ni več.
- Metoda `lokacija()` vrne trenutno lokacijo kolesarja v obliki terke `(y, x)`.
- Metoda `uspesnost()` vrne število ovir, ki jih je kolesar uničil doslej.

Kolesar, ki bi opravil (levo) pot na sliki, bi pri tem uničil štiri ovire.