Izpit iz predmeta Programiranje 1, 7. februar 2018 ob 11.00. Čas pisanja: 90 minut.

Rešitev oddajte prek Učilnice. Za rešitev naloge lahko dobite določeno število točk, **tudi če ne prestane testov**. Funkcija, ki prestane vse teste, **še ni nujno pravilna**. Upošteva se tudi kvaliteta rešitve.

Dovoljena je uporaba vseh materialov na Učilnici in druge literature na poljubnih medijih. Prepovedan je dostop do vseh drugih spletnih strani in vsaka oblika komunikacije, razen s profesorjem oz. asistentom.

1. Nogavice brez para

Ker smo moški, vemo, barvno slepi, obenem pa v družini z ženo in *n* otroki ni mogoče, da bi imel vsak same enake nogavice, se lahko znajdemo tako, da na vsako nogavico prišijemo številko. Enake nogavice imajo enako številko; ker imamo več enakih nogavic, ima lahko več nogavic enako številko.

Recimo, da iz pralnega stroja potegnemo nogavice s številkami `[1, 2, 3, 2, 3, 1, 3, 1, 1, 1, 1]`. Imamo torej tri pare nogavic `1`, en par nogavic `2` in en par nogavic `3` in (eh, spet!) še eno `3` brez para.

Napiši funkcijo `nogavice(s)`, ki prejme seznam številk nogavic in vrne seznam vseh številk nogavic brez para. V gornjem primeru torej vrne `[3]`. Vrstni red elementov v seznamu naj bo enak vrstnemu redu zadnjih pojavitev teh številk; za `[1, 1, 4, 1, 3, 1]` vrne `[4, 3]` in ne `[3, 4]`.

2. Bingo

Igra Bingo poteka tako, da ima vsak igralec listek z nekaj številkami. Voditelj žreba številke. Če se izžrebana številka nahaja na igralčevem listku, jo ta prečrta. Zmaga igralec, ki prvi prečrta vse številke.

Vedeževalka nam je napovedala vrstni red, v katerem bodo žrebane številke. Imamo seznam listkov za Bingo; izbrali bi radi tistega, na katerem bodo prej prečrtane se številke.

Napiši funkcijo `bingo(listki, vrstni_red)`, ki prejme listke (seznam seznamov številk) in vrne listek, na katerem bodo najprej prečrtane vse številke.

```
bingo([[4, 1, 2, 3, 5], [6, 1, 2, 3, 4], [7, 6, 4, 3, 2]],
      [4, 2, 8, 3, 1, 6, 5, 7])
```

vrne `[6, 1, 2, 3, 4]` (saj bo ta očitno prehitel prvega in zadnjega, zaradi 5 in 7).

Funkcija naj ne spreminja podanega seznama!

3. Selitve

Napiši funkcijo `selitve(zacetek, datoteka_selitev, kraj)`, ki prejme začetno razdelitev oseb, ime datoteke s selitvami in ime nekega kraja. Vrniti mora množico imen oseb, ki po podanih selitvah živijo v podanem kraju.

Začetna razdelitev oseb po krajih je lahko, recimo, `[("Ljubljana", "Jana"), ("Šentvid", "Vid"), ("Mala Polana", "Ana"), ("Maribor", "Bor"), ("Kamnik", "Nik"), ("Ozeljan", "Jan"), ("Županje Njive", "Ive"), ("Koroška Bela", "Ela"))]`. Vedno le po ena oseba v vsakem mestu.

Argument `datoteka_selitev` pove ime datoteke z vsebino, ki je lahko, recimo, taka

```
Najprej grejo iz "Mala Polana" v "Šentvid".
Iz "Kamnik" pa tudi v "Šentvid".
Pa še iz "Koroška Bela" odidejo v "Ozeljan".
Potem pa iz "Šentvid" v "Maribor" (kwa?!).
```

Vsaka vrstica torej vsebuje dve imeni krajev, zapisani v dvojnih narekovajih. V vrstici ni drugih dvojnih narekovajev. Vedno se preselijo vsi prebivalci podanega kraja.

Če je gornje začetno stanje shranjeno v seznamu `zacetek`, one štiri vrstice pa v datoteki `selitve.txt`, mora klic `selitve(zacetek, "selitve.txt", "Maribor")` vrniti `{"Ana", "Vid", Nik", "Bor"}`, klic `selitve(zacetek, "selitve.txt", "Šentvid")` pa vrne prazno množico (saj so šli Šentvidčani v Maribor).

Pri reševanju si lahko – ni pa nujno – pomagaš s tem, kar boš sprogramiral(a) v peti nalogi.

OBRNI LIST

4. Rekurzivni štumfi, zokni, kalcete, fuzetlne in kucjte

Napiši **rekurzivno** funkcijo `brez_para(nogavica, nogavice)`, ki prejme številko nogavice in podoben seznam kot prva naloga. Vrniti mora `True`, če je nogavica brez para, in `False`, če ni.

Klic `brez_para(39, [41, 39, 39, 41, 41, 39, 39])` vrne `False` in klic `brez_para(41, [41, 39, 39, 41, 41, 39, 39])` vrne `True`, saj imamo eno 41 brez para.

5. Sledilnik

Napiši razred `Sledilnik`, katerega konstruktor prejme začetno razdelitev po krajih v takšni obliki, kot smo jo imeli tudi v tretji nalogi. Te podatke si shrani v poljubni obliki, ki sem vam zdi primerna. Če vam pride prav (ni pa nujno, da vam bo), lahko predpostavite, da so imena unikatna, torej, da ni dveh oseb z enakim imenom.

Razred ima naslednje metode:

- `kje_zivi(self, oseba)` vrne kraj, v katerem trenutno živi podana oseba;
- `prebivalci(self, kraj)` vrne množico prebivalcev podanega kraja;
- `preseli(self, oseba, kraj)` preseli podano osebo v podani kraj;
- `preseli_vse(self, odkod, kam)` preseli vse prebivalce kraja `odkod` v kraj `kam`;
- `selitev(self)` vrne število vseh selitev. Če se oseba seli v kraj, v katerem že živi, se to šteje za selitev. Če se iz nekega kraja preselijo tri osebe v drug kraj, so to tri selitve.