

Vse rešitve shranite v eno samo datoteko s končnico `.py` in jo oddajte prek Učilnice. Vse funkcije naj imajo takšna imena, kot jih predpisuje naloga. **Pozorno preberite** naloge in ne rešujte le na podlagi primerov!

Da rešitev ne bi imela trivialnih napak, **jo preverite s testi** v ločeni datoteki na Učilnici. Za rešitev naloge lahko dobite določeno število točk, **tudi če ne prestane testov**. Funkcija, ki prestane vse teste, **še ni nujno pravilna**.

Pri reševanju nalog je dovoljena vsa literatura na poljubnih medijih, ves material, ki je objavljen na Učilnici, vključno z objavljenimi programi; njihova uporaba in predelava se ne šteje za prepisovanje.

Izpit morate pisati na fakultetnih računalnikih. Študenti s predolgimi vratovi in podobnimi hibami bodo morali zapustiti izpit. Hujše kršitve, na primer uporabo telefonov in podobno, bomo prijavili disciplinski komisiji.

Študentom **Kognitivne znanosti** je potrebno za 100 % rešiti štiri naloge. Naloga iz snovi, ki se vas ne tiče, je zadnja, vendar jo lahko, če želite, rešite; točke se vam bodo upoštevale.

1. Najprej lihi

Napišite funkcijo `najprej_lihi(s)`, ki kot argument prejme seznam števil `s`. Njihov vrstni red naj spremeni tako, da bodo v njem najprej liha, nato soda števila – vsaka zase v enakem vrstnem redu, kot so bila prej. Funkcija ne sme vračati ničesar.

Primer

```
>>> s = [5, 8, 4, 17, 13, 10, 9]
>>> najprej_lihi(s)
>>> s
[5, 17, 13, 9, 8, 4, 10] # 5, 17, 13 in 9 so v enakem vrstnem redu kot prej; soda tudi
```

2. Binarno

Napišite **rekurzivno** funkcijo `binarno(n)`, ki kot argument prejme število `n` in kot rezultat vrne niz s tem številom v dvojiškem zapisu.

Primeri

```
>>> binarno(42)
'101010'
```

Namig: desno številko dobiš tako, da izračunaš ostanek po deljenju z 2. Pred njo pa moraš postaviti niz, ki ga dobiš, če v dvojiški zapis spremeniš število `n`, ki ga celoštevilsko deliš z 2.

3. Dopisovalci

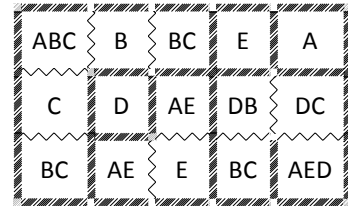
V tej nalogi bomo delali s slovarji, katerih ključi bodo imena ljudi, pripadajoče vrednosti pa množice imen vseh, ki jim je dotični pisal. Če je takšen slovar, recimo, `relacije` in je `relacije['Ana']` enako `{'Berta', 'Cilka', 'Dani'}`, to pomeni, da je Ana doslej pošiljala sporočila Berti, Cilki in Dani. Napisati boste morali več funkcij.

- `dopis(kdo, komu, relacije)` naj v slovar `relacije` pribeleži, da je `kdo` pisal `komu`. (Če je `kdo` v preteklosti že pisal `komu`, `dopis` ne bo spremenil ničesar.)
- `najzgovornejsi(relacije)` naj vrne ime osebe, ki je doslej pisala največ različnim osebam. Če je takšnih več, naj vrne poljubno med njimi.
- `vse_osebe(relacije)` naj vrne množico vseh oseb v slovarju – tako teh, ki so komu pisale, kot tistih, ki jim je kdaj kdo pisal.
- `neznanci(ime, relacije)` naj vrne množico imen vseh tistih oseb, ki so v slovarju, vendar jim oseba `ime` ni nikoli pisala.

Predpostaviti smete, da je `relacije` tipa `defaultdict(set)` in ne navaden `dict`.

4. Zaporniki

Petnajst zapornikov se je znašlo v petnajstih celicah mednarodnega protiterorističnega zapora. Dva zapornika se lahko pogovarjata, če sta v sosednjih celicah in govorita vsaj en skupni jezik (ali, učeno povedano, če je presek jezikov, ki jih govorita, neprazen ;).



Zapornik v prvi celici govori jezike A, B in C, zato se lahko pogovarja s svojim desnim sosedom (ki govori jezik B) in spodnjim sosedom (ki govori jezik C). Spodnji sosed se lahko pogovarja s še spodnejšim, ki govori B in C. Nesrečnik, ki čepi v zgornji desni celici govori jezik A in se ne more pogovarjati ne z levim sosedom (ta govori E) ne s spodnjim (ki govori DC). Pare, ki se lahko pogovarjajo, smo v gornjem primeru označili tako, da smo "stanjšali" zid med celicama.

Napišite funkcijo `sogovorniki(zapor)`, ki prejme razpored zapornikov v obliki `[["ABC", "B", "BC", "E", "A"], ["C", "D", "AE", "DB", "DC"], ["BC", "AE", "E", "BC", "AED"]]` in vrne število parov zapornikov, ki se bodo lahko pogovarjali. V gornjem primeru funkcija vrne 9.

Funkcija naj bo seveda splošna: sprejema naj tudi drugačne konfiguracije zaporov in več jezikov. Seveda pa bo en jezik vedno opisan z eno črko niza.

5. Blok

Definirajte razred `Blok`, ki opisuje stanovanjski blok s podanim številom nadstropij. V vsakem nadstropju bo eno stanovanje. Razred naj ima naslednje metode.

- Konstruktor sprejme kot argument število nadstropij v bloku.
- `vseli(nadstropje, ime)` v nadstropje `nadstropje` vseli osebo z imenom `ime` in vrne `True`. Če je stanovanje že zasedeno, pa pusti v njem prejšnjega lastnika in vrne `False`.
- `stanovalec(nadstropje)` vrne ime stanovalca v podanem nadstropju. Če je stanovanje še prazno, vrne `None`.
- `izseli(nadstropje)` sprazni stanovanje v nadstropju `nadstropje`. Metoda ne vrne ničesar.
- `kamorkoli(ime)` vseli osebo s podanim imenom v najvišje prosto nadstropje in vrne `True`. Če je ves blok že povsem poln, vrne `False`.