Izpit iz predmeta Programiranje 1, 22. januar 2011 ob 12.00.

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, ne lastnih prenosnikih.

Študenti s predolgimi vratovi in podobnimi hibami bodo morali zapustili izpit, katerega opravljanje se bo štelo kot neuspešno. Hujše kršitve bomo prijavili disciplinski komisiji.

Čas pisanja: 90 minut.

1. Sodost čebel

Pri procesu izdelave medu iz nektarja je pomembno, da ločimo čebele, ki so obrale liho število cvetov od čebel, ki so obrale sodo število cvetov.

Podan je seznam, ki vsebuje terke `(ime_cebele, stevilo_obranih_cvetov)`. Napišite funkcijo `loci(cebele)`, ki prejme takšen seznam in vrne dva seznama. V prvem naj bodo imena čebel, ki so obrale sodo število cvetov, v drugem pa imena čebel, ki so obrale liho število cvetov. Za seznam `[('Ana', 5), ('Berta', 7), ('Cilka', 2)]` naj funkcija vrne seznama `['Cilka']` in `['Ana', 'Berta']`. Čebele v obeh seznamih naj bodo urejene enako kot v prvotnem seznamu.

2. Besedna igra

Neki otroci se takole igrajo z besedami:

- Uredijo črke po abecedi. Tako iz HANA nastane AANH.
- Zamenjajo večkratne ponovitve črke s črko in številko, ki pove, kolikokrat se črka ponovi. Iz AANH tako dobijo A2N1H1.

Njihov postopek torej predela besedo HANA v A2N1H1.

Napišite funkcijo `skrij(beseda)`, ki takole predeluje besede: kot argument dobi besedo (npr. “HANA”) in kot rezultat vrne predelano besedo (“A2N1H1”).

Nasvet: ni potrebno, da funkcija izvaja natančno gornji postopek. Uberete lahko bližnjico, pomembno je le, da je rezultat enak. Potrebovali boste nekaj (ne vse) od naslednjega: `defaultdict`, `Counter`, `sorted`.

3. Čebele z leve in desne

V nekem vrtu so rože posejane ob potki, zato je količina nektarja v njih opisana z običajnim (enodimenzionalnim) seznamom. Obiranja se lotita dve čebeli: prva gre z leve proti desni, druga z desne proti levi. Čebela potrebuje eno sekundo, da se premakne iz cveta na cvet in eno sekundo, da obere eno enoto nektarja. Napišite funkcijo `srecanje(vrt)`, ki pove na katerem cvetu se bosta čebeli srečali. Za vrt `[1, 4, 1, 2, 8, 3]` mora vrniti 4.

Funkcija mora upoštevati, da se lahko čebeli srečata tudi med letom in ne samo na cvetu. Za seznam `[0, 0, 0, 0]`, naj vaš program vrne `1.5`.

Lahko se zgodi, da ob trenutku srečanja ostane kakšen cvet neobran. Čebeli se na vrtu `[6, 0]` srečata na prvem cvetu, na katerem je v tem trenutku še 5 enot nektarja.

Namig: najprej razmislite, koliko časa bosta potrebovali obe čebeli skupaj, nato za eno od čebel izračunajte, kje bo takrat.

4. Največ vnukov

Imejmo rodovnike, ki so podani tako, kot so bili na predavanjih iz rekurzije. Napišite funkcijo `najvec_vnukov(ime, rodovnik)`, ki pove, koliko vnukov ima oseba z največ vnuki. Funkcija dobi kot argument slovar z rodovnikom in ime osebe, po katere rodbini gledamo (*vševši z osebo samo*).

Pazi: rodovnik tule ni globalna spremenljivka, kot je bila na predavanjih, temveč argument funkcije.

Nasvet: mogoče vam bo lažje, če najprej napišete funkcijo `vnukov(ime, rodovnik)`, ki pove, koliko vnukov ima oseba `ime`.

5. Liki

Definirajte razred `Liki`, ki bo shranjeval zbirko likov. Razredu `Liki` dodajte metode:

- `krog(self, x, y, r)`, ki doda krog s podanimi koordinatami središča in polmerom;
- `pravokotnik(self, x1, y1, x2, y2)`, ki doda pravokotnik s podanimi koordinatami nasprotnih oglišč;
- `ploscina(self)`, ki vrne skupno ploščino vseh shranjenih likov
- `krogi(self, max_r)`, ki vrne seznam `[(x1, y1), (x2, y2), ..., (xn, yn)]` s središči vseh shranjenih krogov, ki imajo polmer manjši od `max_r`.