Izpit iz predmeta Programiranje 1, 26. januar 2010 ob 10.30

Pri reševanju nalog je dovoljena domiselnost. Bližnjice do pravilne rešitve nalog so vredne pohvale. Dovoljena je raba standardnih Pythonovih knjižnic (`os`, `random`, `math`...), prepovedane pa so dodatne knjižnice, kot so PIL, numpy, PyQt... Ali je neka knjižnica standardna knjižnica ali ne, boste najlažje prepoznali po tem, ali je opisana v Pythonovi dokumentaciji.

Funkcije naj imajo takšna imena, kot jih predpisuje naloga. V rešitvah naj bo označeno, kateri del kode predstavlja rešitev katere naloge. Če naloga zahteva, naj funkcija *vrne rezultat*, mora *vrniti rezultat*, ne pa *izpisovati*. Funkcija mora podatke dobiti prek argumentov, ne prek zunanjih (npr. globalnih) spremenljivk, če ni eksplicitno določeno drugače.

Vse naloge so vredne enako število točk.

Rešitve vseh nalog shranite v eno samo datoteko .py in jo oddajte prek tega strežnika na enak način, kot ste oddajali domače naloge. Uporabniki prenosnikov bodo rešitve oddajali na USB ključih.

Pri reševanju nalog je dovoljena vsa literatura na poljubnih medijih. Prepovedana pa je uporaba interneta, z izjemo učilnice (različice, do katere imate dostop s teh računalnikov). Na računalniku mora biti izključena brezžična povezava in bluetooth ter ugasnjen Skype in drugi programi ter priprave, ki jih je mogoče uporabiti za komuniciranje. Kršilci teh pravil bodo zapustili izpit, katerega opravljanje se bo štelo kot neuspešno. Hujše kršitve bomo prijavili disciplinski komisiji za študente.

1. Starost

Napiši funkcijo `starost(emso)`, ki kot argument prejme EMŠO osebe in kot rezultat vrne njeno dopolnjeno starost. Upoštevajte, da smo danes 26. januarja, torej je nekdo, ki je bil rojen 20. januarja 2000 (ali celo 26. januarja 2000) star 10 let, nekdo, ki je rojen 10. februarja 2000 pa samo 9 let, ker letos še ni praznoval rojstnega dne.

EMŠO je sestavljena tako, da sta prvi števki rojstni dan v mesecu, drugi števki predstavljata mesec, naslednje tri pa leto brez tisočice. Prvih sedem števk EMŠO osebe, rojene 10. 5. 1983, bi bil 1005983. Prvih sedem števk EMŠO osebe, rojene 2. 3. 2001, pa bi bil 0203001. Ali je oseba rojena leta 1xxx ali 2xxx, sklepamo po stotici.

Primer

```
>>> starost("2601971500123")
39
>>> starost("2001971500123")
39
>>> starost("2002971500125")
38
```

2. Bomboni

Imamo gručo otrok in v seznamu je zapisano, koliko bombonov ima kateri od njih. Ker ne želimo prepirov, bi radi poskrbeli, da bodo imeli vsi otroci enako bombonov. Ker bombonov seveda ne moremo jemati, je naša naloga napisati funkcijo `bomboni(s)`, ki kot argument dobi seznam s številom bombonov, ki jih imajo otroci, kot rezultat pa vrne, koliko bombonov je potrebno razdeliti, da jih bodo imeli vsi otroci toliko, kolikor jih ima ta, ki jih ima največ.

Primer

```
>>> bomboni([5, 8, 6, 4])
9
```

Največ bombonov ima drugi otrok, 8. Torej bomo dali prvemu 3, tretjemu 2 in četrtemu 4, da jih bodo imeli vsi po 8, tako kot drugi otrok. Skupaj bomo torej razdelili 3+2+4=9 bombonov, zato funkcija vrne 9.

3. Podobna beseda

Globalna spremenljivka `besede` vsebuje seznam besed. Naj bo takšen:

```
besede = ["ana", "berta", "cilka", "dani", "ema", "fanci", "greta", "hilda"]
```

Predpostaviti smete, da so vse besede zapisane s samimi malimi črkami.

Sestavite funkcijo `podobna(beseda)`, ki kot argument sprejme besedo in kot rezultat vrne tisto besedo iz gornjega seznama, v kateri se pojavi čim več črk, ki se pojavijo tudi v dani besedi. Vsako ujemajočo se črko štejemo le enkrat, tudi če se v obeh besedah pojavi večkrat. Če obstaja več besed, ki imajo največ skupnih črk z dano besedo, naj funkcija vrne tisto med njimi, ki je prej po abecedi.

Funkcija naj deluje tako, da ne razlikuje med malimi in velikimi črkami.

Primer

```
>>> podobna("merjasec")
'berta'
>>> podobna("zmaj")
'ema'
>>> podobna("Krava")
'berta'
```

Ana in krava ujemata v eni črki (a) in ne v dveh (dva a-ja).

4. Pari števil

Napišite funkcijo `pari()`, ki vrne seznam parov števil med 1 in 1000, za katera velja, da imata različno število mest, vendar je vsota njunih števk enaka. Vsak par naj se pojavi le enkrat, ne glede na vrstni red. V seznamu naj se torej pojavi, recimo, (72, 526) ali (526, 72), ne pa oboje! Vrstni red parov znotraj seznama je poljuben.

Primer: prvih deset elementov seznama je lahko, recimo, takšnih: (76, 526), (88, 754), (8, 134), (27, 153), (95, 617), (67, 922), (52, 7), (73, 541), (69, 465), (96, 726). Par (76, 256) je na seznamu, ker 7+6 = 5+2+6. Pač pa na seznamu ni para (760, 526), saj imata števili enako število mest.

5. Trgovinski računi

V tej nalogi se bomo igrali s trgovinskimi računi takšne oblike:

```
Trgovina z mešanim blagom Stane
Pot k Stanetu 18
4321 Lem


Sepuljke          15.12
Sepuljke (bio)    22.00
posebni popust
Knjiga            13.50
------------------------
skupaj            50.62


Hvala za nakup!
Se priporočamo.
```

Med imenom artikla (oz. besedo "skupaj") in ceno (oz. skupno vsoto) je tabulator (\t). Tabulatorjev na drugih mestih v dokumentu ni. Pred vsoto vedno piše "skupaj". Vsota sledi seznamu kupljenih artiklov in cen. Besedilo pred seznamom artiklov in cen ter za njim je lahko poljubno in poljubno dolgo. Prav tako se lahko znotraj seznama artiklov pojavljajo dodatne vrstice s poljubnim besedilom, kot je na primer gornji "posebni popust".

Sestavite funkcijo `pravilen(imeDat)`, ki kot argument prejme ime datoteke s trgovinskim računom. Funkcija prebere račun in vrne vrednost `True`, če je vsota na računu pravilna in `False`, če ni pravilna ali pa račun ne vsebuje besede "skupaj" in vsote.

Za gornji račun funkcija vrne `True`, saj je 15.12+22.00+13.50 res enako 50.62.