

Izpit iz predmeta Programiranje 1, 26. januar 2010 ob 14.00

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. Dan v letu

Napiši funkcijo `danVLetu(dan, mesec)`, ki prejme datum v letu 2010 in vrne zaporedno številko dneva v letu. Tako je, na primer, 10. februar 41. dan v letu.

Primer

```
>>> danVLetu(26, 1)
26
>>> zapdan = danVLetu(10, 2)
>>> zapdan
41
```

2. Po starosti

Napiši funkcijo `poStarosti(s)`, ki dobi kot argument seznam `s`, ki vsebuje pare (ime osebe, EMŠO) in vrne nov seznam, ki vsebuje imena oseb, urejena po starosti, začenši z najstarejšimi.

EMŠO je sestavljena tako, da sta prvi številki rojstni dan v mesecu, drugi številki 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

```
>>> osebe = [("Ana", "2401983505012"), ("Berta", "1509980505132"), ("Cilka",
"0203001505333"), ("Dani", "1005983505333")]
>>> poStarosti(osebe)
['Berta', 'Ana', 'Dani', 'Cilka']
>>> poStarosti(osebe)[0]
'Berta'
>>> poStarosti(osebe)[-1]
'Cilka'
```

3. Število znakov

Napišite funkcijo `najraznolika(bes)`, ki kot argument prejme seznam nizov in kot rezultat vrne niz, ki ima največ različnih znakov. Male in velike črke upoštevajte kot iste znake - beseda "MamA" ima samo dva različna znaka.

Primer

```
>>> besede = ["RABarbara", "izpit", "zmagA"]
>>> najraznolika(besede)
'izpit'
```

4. Numerologija

Numerologi računajo takole. Naj bo vrednost znaka A enaka 1, vrednost znaka B naj bo 2, vrednost C 3, vrednost D 4 in tako lepo naprej po angleški abecedi. Vzemimo neko besedo, denimo "Benjamin". Vrednosti črk te besede so 2, 5, 14, 10, 1, 13, 9 in 14. Če jih seštejemo, dobimo 68. Nato seštejemo 6 in 8: dobimo 14. Nato seštejemo 1 in 4 ter dobimo 5. Ker smo tako prišli do enomestnega števila, se ustavimo: vrednosti imena BENJAMIN je 5. Za vajo počunajmo še Berto: vrednosti črk so 2, 5, 18, 20, 1. Vsota je 46. $4+6=10$. $1+0=1$. Tu se ustavimo, vrednost imena Berta je 1.

Od te številke je domnevno odvisno prav vse v vašem življenju, vključno s tem, ali boste uspešno opravili današnji izpit ali ne. Zato se zdi, da bi bilo koristno, če bi napisali funkcijo `numerologija(ime)`, ki kot argument dobi ime (same črke, brez presledkov ali drugih znakov) in vrne njegovo vrednost. Male in velike črke imajo iste vrednosti.

Primer

```
>>> numerologija("Berta")
1
>>> numerologija("Benjamin")
5
>>> numerologija("Berta") == numerologija("Benjamin")
False
```

Namig: če funkciji `ord` podamo znak, vrne kodo ASCII tega znaka. Kode ASCII velikih črk so za 64 večje od vrednosti črk, kakršne potrebujemo pri tej nalogi - `ord("A")` je 65, `ord("B")` je 66...

5. Delnice

Imamo seznam z gibanji določene delnice v posameznih mesecih 2009, ki je lahko, na primer, takšen:

```
delnica = [1, -2, -4, 1, 2, -1, 3, 4, -2, 1, -5, -5]
```

Delnica na gornjem seznamu je pridobila 1 točko januarja, februarja je izgubila 2, marca je izgubila štiri, aprila pridobila eno... Napišite funkcijo `posrednik`, ki dobi seznam, kakršen je gornji in vrne par (terko), ki pove, kdaj je bilo delnico pametno kupiti in prodati, da bi z njo dosegli največji dobiček. Delnico bomo kupili in prodali samo enkrat.

Primer

```
>>> delnica = [1, -2, -4, 1, 2, -1, 3, 4, -2, 1, -5, -5]
>>> kupim, prodam = posrednik(delnica)
>>> kupim
3
>>> prodam
8
```

Delnico je smiselno kupiti v tretjem mesecu (če je januar "ničti") in prodati pred osmim. Vmes vrednost delnice sicer enkrat tudi nekoliko pade (-1), vendar nas to ne moti, saj po tem še pošteno zraste. Skupaj smo v tem času zaslužili $1+2-1+3+4=9$ točk.