1. Napíš funkciu posun(zoznam), ktorá posunie prvky v danom zozname tak, že prvý sa presťahuje na koniec. Funkcia nič nevracia, len zmení obsah pôvodného zoznamu. Napríklad:

**>>>** a = [2, 3, '5', 7, 11]

**>>>** posun(a)

**>>>** a

[3, '5', 7, 11, 2]

**>>>** zoz = 'kto druhemu jamu kope'.split()

**>>> for** i **in** range(5):

print(zoz)

posun(zoz)

['kto', 'druhemu', 'jamu', 'kope']

['druhemu', 'jamu', 'kope', 'kto']

['jamu', 'kope', 'kto', 'druhemu']

['kope', 'kto', 'druhemu', 'jamu']

['kto', 'druhemu', 'jamu', 'kope']

1. Napíš funkciu vyhod\_none(ntica), ktorá z danej n-tice vyhodí všetky výskyty None. Funkcia vráti (return) túto novú n-ticu (nič nevypisuje). Napríklad:

**>>>** vyhod\_none((**None**, 1, **None**, **None**))

(1,)

1. Napíš funkciu dve\_kocky(pocet), ktorá vráti 13-prvkový zoznam celých čísel. Tento sa skonštruuje takto: zadaný pocet-krát hodí dvoma kockami (dve nádodné čísla od 1 do 6), pre každý takýto hod urobí ich súčet a v 13-prvkovom zozname si eviduje počet výskytov každého súčtu, napríklad zoznam[5] bude označovať, koľko krát pri našej simulácii padol súčet 5; zrejme na začiatku budú v zozname samé 0 a potom pri každom hode sa číslo na príslušnom indexe zvýši o 1. Funkcia nič nevypisuje, len vráti vytvorený 13-prvkový zoznam celých čísel. Mohol by si dostať napríklad, takýto zoznam:

**>>>** dve\_kocky(1000)

[0, 0, 29, 36, 90, 105, 137, 181, 125, 126, 93, 50, 28]

1. Napíš funkciu desiatkova(cislo), ktorá vráti (return) zoznam cifier daného čísla v desiatkovej sústave. Využi funkcie str a int. Napríklad:

**>>>** desiatkova(11213)

[1, 1, 2, 1, 3]

Všimni si, že takto sa dá vypočítať ciferný súčet čísla:

**>>>** sum(desiatkova(11213))

8

1. Napíš funkciu dvojkova(cislo), ktorá vráti (return) zoznam cifier daného čísla v dvojkovej sústave. Využi f'{cislo:b}'. Napríklad:

**>>>** dvojkova(11213)

[1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1]

1. Napíš funkciu z\_dvojkovej(zoznam), ktorá dostane zoznam cifier dvojkového zápisu čísla v tvare z predchádzajúcej úlohy. Funkcia vráti celé číslo (return), ktorého cifry v dvojkovej sústave zodpovedajú zadanému zoznamu. Napríklad:

**>>>** z\_dvojkovej([1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1])

11213

**>>>** z = [1] + [0] \* 20

[1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

**>>>** z\_dvojkovej(z)

1048576

**>>>** 2 \*\* 20

1048576

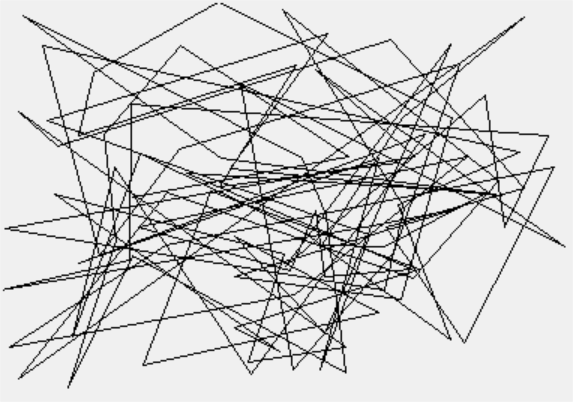
Využi funkciu int(reťazec, 2), pomocou ktorej sa zo znakového reťazca s dvojkovým zápisom čísla vyrobí celé číslo.

1. Napíš funkciu nahodne\_body(pocet), ktorá vráti zadaný počet náhodných bodov v grafickej ploche (dvojíc čísel z 380x260). Funkcia vráti zoznam, ktorý bude obsahovať dvojprvkové n-tice celých čísel. Napríklad:

**>>>** nahodne\_body(5)

[(118, 103), (299, 27), (247, 118), (166, 237), (269, 225)]

1. Ak by si celú postupnosť 100 náhodných bodov z predchádzajúcej úlohy vykreslil pomocou jediného create\_line, dostal by si nejakú čmáranicu (vyskúšaj). Ďalej otestuj, ako sa vykreslí táto postupnosť, keď sa najprv usporiada pomocou funkcie sorted. Napríklad, náhodná postupnosť 100 bodov:

[](http://python.input.sk/_images/09_c01.png)

a po usporiadaní pomocou sorted:

[](http://python.input.sk/_images/09_c02.png)

1. Napíš funkciu sort\_y(zoznam), kde zoznam obsahuje dvojice (typ tuple) celých čísel. Tieto reprezentujú súradnice (x, y) nejakých bodov v rovine. Tento zoznam treba usporiadať od najmenšieho po najväčší, lenže podľa druhých prvkov dvojíc (y-ových súradníc). Uvedom si, že volanie zoznam.sort() by usporiadalo zoznam podľa prvých prvkov dvojíc (x-ových súradníc). Napríklad:

**>>>** xy = [(100, 30), (200, 10), (300, 20)]

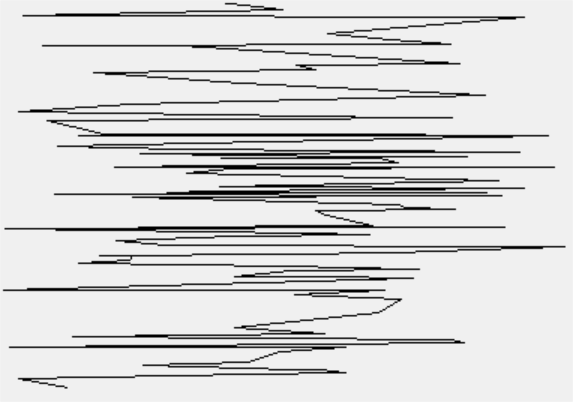
**>>>** sort\_y(xy)

**>>>** xy

[(200, 10), (300, 20), (100, 30)]

Triediť môžeš takto: najprv v zozname v každej dvojici vymeníš x a y, potom normálne utriediš (napríklad pomocou metódy sort) a nakoniec v takto utriedenom zozname vrátiš všetky dvojice do pôvodného stavu (vymeníš x a y).

Podobne ako v predchádzajúcej úlohe vykresli takto usporiadanú postupnosť vrcholov pomocou create\_line:

[](http://python.input.sk/_images/09_c03.png)

1. Napíš funkciu prerob(cislo), ktorá z daného celého čísla vyrobí reťazec, ale tak, že cifry zoskupí do trojíc (sprava) a oddelí podčiarkovníkom. Využi funkciu join, preto z čísla najprv vyrob zoznam trojznakových reťazcov. Funkcia nič nevypisuje, ale vráti (return) výsledný reťazec. Napríklad:

**>>>** prerob(12345678)

'12\_345\_678'

Je zaujímavé, že Python aj takto zadané celé čísla považuje za korektné, napríklad:

**>>>** 12\_345\_678

12345678

Výsledok svojej funkcie prerob môžeš skontrolovať pomocou f'{cislo:\_}'.

1. Napíš funkciu sipka(xy1, xy2), ktorá do grafickej plochy nakreslí šípku z bodu xy1 do bodu xy2. Oba tieto parametre sú dvojice čísel (tuple), t.j. súradnice bodov. Šípku kresli pomocou create\_line, v ktorej využiješ ďalší pomenovaný parameter arrow='last' (podobné šípky sme kreslili v 15. úlohe z 5.cvičenia). Teraz nakresli štyri šípky, ktoré nakreslia obrys takéhoto štvorca:

**import** **tkinter**

**def** sipka(xy1, xy2):

...

canvas = tkinter.Canvas()

canvas.pack()

canvas.create\_rectangle(150, 50, 250, 150, fill='gold')

sipka(...)

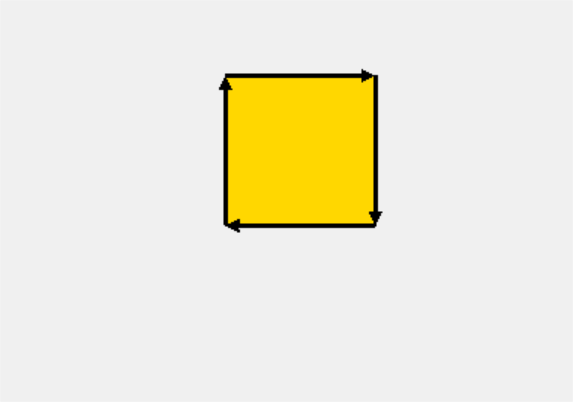
sipka(...)

sipka(...)

sipka(...)

tkinter.mainloop()

Mal by si dostať takýto obrázok:

[](http://python.input.sk/_images/09_c04.png)

1. Napíš funkciu vektor(xy, dlzka, uhol), ktorá nakreslí vektor (teda šípku) z bodu xy (dvojprvkový tuple celých čísel) s danou dĺžkou a s daným uhlom (v stupňoch). Uvedom si, že koncové body takéhoto vektora ležia na kružnici s polomerom dlzka a stredom xy. Na kreslenie šípky využi volanie funkcie sipka z predchádzajúcej úlohy. Funkcia vektor vráti (return) súradnicu koncového bodu vektora ako dvojicu (tuple) celých čísel. Napríklad:

**import** **tkinter**

**from** **math** **import** sin, cos, radians

**def** sipka(xy1, xy2):

...

**def** vektor(xy, dlzka, uhol):

...

**return** ...

canvas = tkinter.Canvas()

canvas.pack()

poz = (200, 120)

**for** uhol **in** range(0, 720, 144):

poz = vektor(poz, 100, uhol)

tkinter.mainloop()

nakreslí:

[](http://python.input.sk/_images/09_c05.png)

1. Napíš funkciu zoznam\_cisel(retazec), ktorá z daného reťazca vyrobí zoznam celých čísel - použi metódu split. Napríklad:

**>>>** a = zoznam\_cisel('12 345 6 -78 9000')

**>>>** a

[12, 345, 6, -78, 9000]

1. Napíš funkciu retazec(zoznam), ktorá zo zoznamu čísel vyrobí reťazec - použi metódu join. Napríklad:

**>>>** retazec([12, 3.45, 6, -78, 9000])

'12 3.45 6 -78 9000'

1. Napíš funkciu zadaj(pocet), ktorá vráti prečítanú n-ticu čísel zo vstupu (input). Funkcia si vypýta od používateľa, aby zadal príslušný počet čísel (napríklad input(f'zadaj {pocet} čísla: ')) a ak ich používateľ nezadá požadovaný počet, bude si tento input pýtať znovu a znovu. Funkcia vráti (return) n-ticu celých čísel a nie n-ticu reťazcov. Napríklad:

**>>>** tri = zadaj(3)

zadaj 3 čísla: 12 3

zadaj 3 čísla: 12 3 456

**>>>** tri

(12, 3, 456)

1. Napíš funkciu zisti(slovo1, slovo2), ktorá zistí (vráti True alebo False), či sú dve zadané slová zložené presne z tých istých písmen - možno v inom poradí. Napríklad:

**>>>** zisti('Skola', 'Lasko')

True

**>>>** zisti('poobede', 'bopeodo')

False

Pomôcka: použi funkciu sorted

1. Napíš funkciu vsetky\_rozne(zoznam), ktorá zistí (vráti True alebo False), či sú všetky prvky zoznamu rôzne. Najprv si vyrob utriedený pomocný zoznam (nepokaz pôvodný) a v ňom zisťuj, či sa nenachádzajú dve rovnaké hodnoty za sebou. Napríklad:

**>>>** vsetky\_rozne([3, 8, 7, 9, 4, 1, 6, 10, 5, 2])

True

**>>>** zoz = [3, 8, 7, 9, 4, 1, 6, 3, 10, 5, 2]

**>>>** vsetky\_rozne(zoz)

False

**>>>** zoz

[3, 8, 7, 9, 4, 1, 6, 3, 10, 5, 2]

1. Napíš funkciu enum(postupnost), ktorá vráti (return) n-ticu dvojíc. V tejto n-tici je prvý prvok poradové číslo (od 0 do počet prvkov postupnosti minus 1) a druhým je prvok z danej postupnosti. Malo by to dať rovnaký výsledok ako tuple(enumerate(postupnost)) ale bez použitia enumerate. Napríklad:

**>>>** enum([12, 'dva', 3.14])

((0, 12), (1, 'dva'), (2, 3.14))

1. Python ponúka ešte jednu štandardnú funkciu zip. Táto funkcia, keď dostane nejaké dve postupnosti (napríklad zoznam, n-ticu, reťazec, range, …), vytvorí z nich jednu novú postupnosť dvojíc (tuple): v každej takejto dvojici je jeden prvok z prvej a jeden z druhej postupnosti. Môžeš vyskúšať, napríklad:

**>>>** list(zip('python', [2, 3, 5, 7]))

[('p', 2), ('y', 3), ('t', 5), ('h', 7)]

Zrejme, ak je jedna z týchto postupností kratšia, výsledok sa nastaví podľa nej. Napíš funkciu moj\_zip(post1, post2), ktorá z dvoch postupností (iterovateľných objektov možno rôznej dĺžky) vytvorí jeden zoznam dvojíc. Nepouži štandardnú funkciu zip().

1. Prepíš funkciu enum z (18) úlohy tak, aby neobsahovala cyklus a využila tvoju funkciu moj\_zip. Teda:

**def** enum(postupnost):

**return** ... moj\_zip ...