

Cvičenie na iteráciu a zoznamy 2

Na zahriatie

Napište funkciu, ktorá vypočíta:

- faktoriál kladného celého čísla
- súčet všetkých nezáporných čísiel v zozname
- súčet 3 najvyšších hodnôt v zozname

Jeleňovi pivo nelej!

Napište funkciu `is_palindrome`, ktorá vezme slovo (reťazec) a vráti `True` pokiaľ je slovo **palindróm**, inak `False`.

Prvočísla

Stránka projecteuler.net obsahuje veľké množstvo matematicko-programátorských úloh rôznej zložitosti.

Vašou úlohou bude vyriešiť [úlohu 7](#).

Pre vyriešenie úlohy môžete postupovať nasledovne:

- Pokiaľ si nie ste istý čo je prvočíslo (*ang.* prime), nájdite si definíciu.
- Rozmyslite si čo musíte overiť, aby ste určili či dané číslo je alebo nie je prvočíslo.
- Napíšte funkciu `is_prime`, ktorá vezme prirodzené číslo a vráti `True` ak číslo je prvočíslo. V opačnom prípade vráti `False`.
- Použite vašu funkciu k tomu, aby ste našli prvých 10 prvočísel a overte, že funguje správne.
- Pokračujte k riešeniu úlohy na stránke.

Samozrejme, existujú aj iné, možno priamejšie, postupy, ktoré vám môžu vyhovovať viac, napr. [Eratostenovo sito](#).

Poradie zoradenia

Napište funkciu `sort_indices`, ktorá vezme zoznam a vráti indexy, ktoré zoznam zoradia. Napríklad:

```
lst = [9, 6, 8]
inds = sort_indices(lst)
inds == [1, 2, 0]
```

Prvý prvok v zoradenom zozname je `lst[1]`, druhý `lst[2]`, tretí `lst[0]`

Ďalšie príklady:

- `sort_indices([1, 2, 3, 4])` vráti `[0, 1, 2, 3]`
- `sort_indices([2, 1, 3, 4])` vráti `[1, 0, 2, 3]`
- `sort_indices([4, 3, 1, 2])` vráti `[2, 3, 1, 0]`

Ďalej, napíšte funkciu `apply_inds`, ktorá vezme zoznam a zoznam indexov, a aplikuje tieto indexy na zoznam (vráti nový zoznam):

```
def apply_inds(lst, inds):
    # TODO

apply_inds([9, 6, 8], [1, 2, 0])    # -> [6, 8, 9]
apply_inds([1, 2, 3], [0, 2, 1])   # -> [1, 3, 2]
```

AoC - 2017, Day 2

Vyriešte úlohu zo stránky [Advent of Code](#). Ak sa nechcete na stránku prihlasovať, môj input nájdete v [st. materiáloch na ISe](#).

Pri riešení môžete postupovať nasledovne:

- Riešenie si najprv vyskúšajte na ukázanom príklade.
- Skopírujte si tabuľku čísiel zo zadania do reťazca a dajte reťazec do premennej:

```
table = ""5 9 2 8
9 4 7 3
3 8 6 5""
```

- Napíšte funkciu, ktorá z reťazca napr. "512 92 21 8"vytvorí zoznam `int`-ov. Budete chcieť využiť metódu `str.split`, `for` cyklus a funkciu `int`.
- Pomocou metódy `str.splitlines` prevedte tabuľku na jednotlivé riadky.
- Pomocou `for`-cyklu a vašej funkcie iterujte cez tieto riadky, a vytvorte zoznam zoznamov, kde vnútorné zoznamy sú riadky tabuľky (už ako `int-y`).
- Jednoduchým iterovaním cez zoznam riadkov teraz získate riešenie. Najmenšiu a najväčšiu hodnotu nájdete pomocou funkcií `min` a `max`.

Pekné vypisovanie stromu

Hierarchie, alebo tzv. stromy v informatike, je možné reprezentovať pomocou zanorených zoznamov. Napr. klasifikácia zvierat je strom.

Vašou úlohou bude konvertovať zoznam na podobný výstup ako môžete vidieť na [wikipedia stránke](#), t.j. vypísať to tak, aby bolo jasné čo je podkategória čoho.

Príklad fungovania funkcie:

```
tree = [1, [2, 3, [4], 5]]
```

```
pretty_print(tree)
```

Output:

```
1
  2
  3
    4
  5
```

V tejto úlohe budete pracovať s klasifikáciou slabozubcov. Zoznam nájdete v [štúdijských materiáloch](#), skopírujte si ho do vášho kódu a dajte do premennej.

ROT13 šifra

Príklad 8.5 z knihy Think Python.

Príklad ROT13 sme už inak videli na prvej hodine: [this.py](#).

Palacinkový problém

Túto úlohu som našiel náhodou [na tejto stránke](#). Zadanie je nasledujúce:

Máte naskladaných n palacínok na tanieri. Palacinkám je priradená veľkosť. Každá palacinka má inú veľkosť. Musíte zoradiť palacinky, aby najväčšia bola dole, najmenšia hore, a jediná operácia, ktorú môžete urobiť je vložiť niekam špachtľu a otočiť všetky palacinky naopak.

Toto je nevyriešený problém pre n palacínok - optimálny algoritmus je neznámy, ale jednoduchý postup na riešenie je takýto:

1. Palacinky reprezentujeme zoznamom, napr. [4, 17, 2, 5, 1, 3]
2. Prehľadáme zoznam a nájdeme najväčšiu palacinku – táto palacinka musí byť úplne dole. V našom prípade to bude koniec zoznamu. Zapamätáme si index, kde sa palacinka v zozname nachádza.
3. Vložíme špachtľu za najväčšiu palacinku a otočíme. Najväčšia palacinka bude úplne hore
4. Vložíme špachtľu úplne dole a otočíme, tak aby bola najväčšia palacinka úplne dole.

Opakujeme s druhou najväčšou palacinkou, ale tentoraz otáčame v poslednom kroku, tak aby najväčšia ostala dole.

Vizualizujte sekvenciu krokov k riešeniu pre obecnú sekvenciu.

Príklad:

```
palacinky = [3, 2, 8, 1]
```

```
solve(palacinky)
```

```
[3, 2, 8, 1]
[8, 2, 3, 1]
[1, 3, 2, 8]
[3, 1, 2, 8]
[2, 1, 3, 8]
[2, 1, 3, 8] # no-op
[1, 2, 3, 8]
```

- Používanie funkcií a ich postupné testovanie vám pomôže - nesnažte sa vyriešiť problém “naraz.”

Poznámka: Je možné, že palacinky budú zoradené skorej než dokončíte postup. Toto môžete / nemusíte zohľadniť vo vašej implementácii. Tiež, ako je vidieť z príkladu, ak je prvé číslo už maximum, tak jedno otočenie je možné vynechať.