

Cvícenie 5

Instrukcie:

- Vypracujte všetky ulohy. Na cvícení sa pokuste vypracovať čo najviac uloh a ulohy, ktoré nestihnete na cvícení, potom vypracujte doma.
- **Pozor! Nevytvárajte rekurzívne funkcie pomocou globálnych premenných!** Je to zlý zvyk a na skúske bude za takéto riešenia 0 bodov!
- **Specialne dávam do pozornosti ulohu číslo 7, v ktorej sa oboznámite so slavnou a podľa mňa aj veľmi peknou algoritmicou ulohou „Towers of Hanoi“.**
- **V prípade, že sa na niektorej ulohy zaseknete, pýtajte sa cvičiaceho.**

Časť prvá: Debugger

1. Vyvojové prostredia pre tvorbu počítačových programov spravidla obsahujú nástroj zvaný Debugger. Pomocou tohto nástroja môžete prechádzať vaším programom krok po kroku a môžete sledovať, ako sa menia hodnoty premenných vo vašom programe. Toto je veľmi užitočné pri hľadaní chýb vo vašom programe. Prejdite si tutorial o používaní Debuggera vo vyvojovom prostredí IDLE dostupný na:

<https://www.cs.uky.edu/~keen/help/debug-tutorial/debug.html>

Osvojte si prácu s Debuggerom v IDLE. **Bude to pre vás užitočné!** Ak radšej používate iné vyvojové prostredie ako IDLE, osvojte si prácu s Debuggerom vo vašom preferovanom vyvojovom prostredí!

Časť druhá: Rekúzia

1. Definujte funkciu s parametrom n , ktorá vráti súčet $1+2+3+\dots+n$. Vo funkcii použite **rekúziu!**
Príklad činnosti funkcie: funkcia pre hodnotu parametra $n=1$ vráti 1, funkcia pre hodnotu parametra $n=5$ vráti 15, pre hodnotu parametra $n=10$ vráti 55.

2. Definujte funkciu s parametrom n . Funkcia načíta z klavesnice n čísel a vráti súčet načítaných čísel. Vo funkcii použite **rekúziu!**
Príklad činnosti funkcie: funkcia pre hodnotu parametra $n=5$ načíta 5 čísel. Ak budú načítané čísla 7, 4, -2, 4, -7, funkcia vráti číslo 6.

3. Definujte funkciu s parametrom n . Funkcia načíta z klavesnice n čísel a vráti počet, koľko z načítaných čísel bolo párnych. Vo funkcii použite **rekúziu!**
Príklad činnosti funkcie: funkcia pre hodnotu parametra $n=5$ načíta 5 čísel. Ak budú načítané čísla 7, 4, 0, 10, -7, funkcia vráti číslo 3 (lebo 4, 0 a 10 sú párne). Ak budú načítané čísla 7, 3, 1, 11, -7, funkcia vráti číslo 0 (lebo žiadne z 5 načítaných čísel nebolo párne).

4. Definujte funkciu s parametrom n . Funkcia načíta z klavesnice n čísel a vráti najväčšie z načítaných čísel. Vo funkcii použite **rekúziu!**
Príklad činnosti funkcie: funkcia pre hodnotu parametra $n=5$ načíta 5 čísel. Ak budú načítané čísla 7, 4, 0, 10, -7, funkcia vráti číslo 10.

5. Definujte funkciu s parametrom n . Funkcia načíta z klavesnice n čísel a vráti hodnotu True, ak je súčet načítaných čísel párny. V opačnom prípade vráti funkcia hodnotu False. Vo funkcii použite **rekúziu!**

Pomocka: Zamyslite sa, ako sa zmení parita čísel, ak k nemu pridáme párne číslo, a ako sa zmení, ak k nemu pridáme nepárne číslo.

Priklad cinnosti funkcie: funkcia pre hodnotu parametra $n=5$ nacita 5 cisel. Ak budu nacistane cisla 7, 4, 0, 10, -7, funkcia **vrati** True, pretoze $7+4+0+10+(-7) = 14$, co je parne cislo.

6. Definujte funkciu, ktora pre argument n vrati sucet prvocisel mensich ako n . Vo funkcii pouzite **rekurziu**! Mozete predpokladat, ze mate k dispozicii funkciu *test_prvociselnosti* z minuleho cvicenia.

Priklad cinnosti funkcie: funkcia pre hodnotu parametra $n=10$ **vrati** cislo 17, pretoze prvocisla mensie ako 10 su 2, 3, 5, 7 a $2+3+5+7 = 17$. Funkcia pre hodnotu parametra $n=17$ **vrati** cislo 41, pretoze prvocisla mensie ako 17 su 2, 3, 5, 7, 11, 13 a $2+3+5+7+11+13 = 41$.

7. Vsetky vyssie uvedene ulohy by sa dali jednoducho vyriesit aj bez rekurzie. Rekurzia nam ale niekedy umoznuje jednoducho riesit aj ulohy, ktore by sa nam inak riesili velmi tazko. Prikladom takejto ulohy je slavna algoritmicka uloha "Towers of Hanoi".

a) Pozrite si toto video o ulohe "Towers of Hanoi":

<https://www.youtube.com/watch?v=8lhxIOAfDss>

Podla mna je velmi pekne a zaujimave :)

Komentar1:

Vo videu pouziva profesor Altenkirch prostredie Jupyter. Rovnake funkcie, ktore vytvara a spusta prof Altenkirch vo videu, si ale mozete vytvorit a spustit aj v prostredi Idle.

Komentar2:

Vo videu vytvori prof Altenkirch funkciu

```
def move(f, t) :  
    print("Move a disc from {} to {}".format(f, t))
```

V tejto funkcii vyuziva prof Altenkirch metodu format() , s ktorou sme sa este na predmete nestretli. Ak by sme sa chceli vyhnut metode format(), mohli by sme funkciu move definovat nasledovne

```
def move(f, t) :  
    print("Move a disc from "+f+" to "+t+"!")
```

Takto definovana funkcia move ma rovnaky efekt ako funkcia move od prof Altenkircha. Namiesto metody format() sa v nej vyuziva iba spajanie retazcov, ktore sme preberali este na prvej prednaske.

b) V skripte vo videu vyssie je funkcia hanoi(n, f, h, t), ktora vypise sadu instrukcii ako vyriesit ulohu "Towers of Hanoi", ak na zaciatku mame n diskov na stlpe s nazvom f a tieto disky chceme premiestnit na stlp s nazvom t a mame este k dispozicii stlp s nazvom h . Vasou ulohou je teraz definovat funkciu hanoi_count(n), ktora vrati, kolko presunov diskov treba vykonat, ak chceme vyriesit ulohu "Towers of Hanoi" s n diskami (inak povedane funkcia hanoi_count(n) vrati pocet riadkov, ktore vypise funkcia hanoi(n, f, h, t)). Funkcia hanoi_count(n) musi byt rekurzivna, nesmie volat funkciu hanoi(n, f, h, t) a nesmie nic vypisovat.

8. Vyrieste cvicenie 6.3 na strane 61 v knihe.

Priklad cinnosti funkcie, ktoru mate v danom cviceni vytvorit: volanie *is_palindrome("anna")* **vrati** True, pretoze retazec "anna" je palindrom. Volanie *is_palindrome("anno")* **vrati** False, pretoze retazec "anno" nie je palindrom.

9. Vyrieste cvicenie 6.4 na strane 61 v knihe.

Priklad cinnosti funkcie, ktoru mate v danom cviceni vytvorit: volanie `is_power(8,2)` **vrati** True, pretoze $8 = 2^3$. Volanie `is_power(1,3)` **vrati** True, pretoze $1 = 3^0$. Volanie `is_power(81,3)` **vrati** True, pretoze $81 = 3^4$. Volanie `is_power(80,3)` **vrati** False, pretoze 80 nie je mocnina 3.

10. Vyrieste cvicenie 6.5 na strane 61 v knihe.

Priklad cinnosti funkcie, ktoru mate v danom cviceni vytvorit: volanie `gcd(10,15)` **vrati** 5, volanie `gcd(30,40)` **vrati** 10, volanie `gcd(20,27)` **vrati** 1.

11. Vyrieste cvicenie 5.6 na strane 49 v knihe.