CV.1 Korytnačia grafika

Turtle príkazy

Dokumentacia: https://docs.python.org/2/library/turtle.html

- Pohyb a kreslenie
  - pohyb dopredu/dozadu - forward()/backward()
  - otocenie dolava/doprava - left()/right()
  - nastavenie pozície pera - setposition()
  - nastavenie smeru peru - setheading()
  - undo() - návrat do stavu pred posledný príkaz
  - speed() - rýchlost kreslenia
- Zistenie stavu pera
  - smer - towards()
  - vzdialenost - distance()
- Stav kreslenia
  - kreslit/nekreslit - pendown()/penup()

1. Pomocou príkazu "import turtle" si naimportujte kniznicu turtle.
2. V shelli nakreslite vodorovnú čiaru dĺžky 100 pixelov.
3. V shelli nakreslite rovnostranný trojuholník.

Úlohy

1. Vytvorte nový súbor (New File), skopírujte doň "Základný program" (bez číslovania riadkov!) a spustite ho (Run -> Run module).
2. Diskutujte o tom čo robí medzi sebou/s cvičiacim. Hovoríme, že program definuje funkciu "mocnina" so vstupom i a že program obsahuje for-cyklus.

Korytnačia grafika

Úlohy

1. Vykreslite rovnostranný 3, 4, 5-uholník.
2. Vykreslite 2 štvorce vedľa seba.
3. Vykreslite Obrázok 1 zo súboru [attachment:obrazky.pdf obrazky].

Ťažšie Úlohy

1. Pomocou for-cyklu vykreslite obrázok 2 zo súboru [attachment:obrazky.pdf obrazky].
2. Vykreslite n-uholník. To jest, definujte funkciu so vstupom n, ktorá vykreslí n-uholník.
3. Pomocou for-cyklu vykreslite obrázok 3 zo súboru [attachment:obrazky.pdf obrazky].
4. Pomocou for-cyklu vykreslite obrázok 4 zo súboru [attachment:obrazky.pdf obrazky].

Hardcore Úloha

1. Vykreslite obrázok 5 zo súboru [attachment:obrazky.pdf obrazky].

BRUTAL Úloha

1. Pomocou korytnačky spočítajte odmoninu z n. (3 body - priniesť na prednášku)

CV.2 Základné konštrukcia

if-else

1. Načítajte číslo **a** a vypíšte a=načítané číslo a a*a = načítané číslo na druhú.
2. Načítajte dve premenné a,b z klávesnice.
3. Vytlačte ich maximum (vždy sa vytlačí len jedno číslo). Vyriešte to s použitím iba if (žiadne else). Následne to vyriešte použitím jedného if a jedného else.
4. Vymeňte ich tak aby v **a** bolo menšie číslo.
5. Načítajte tri premenné a,b,c z klávesnice.
6. Vymeňte (zrotujte) obsah troch premenných a,b,c.
7. Vytlačte maximum a minimum.
8. (ťažký) Vymeňte ich tak, aby a < b < c.

for-while

1. Vypíšte za sebou do riadku n hviezdičiek. n dostaneme ako vstup z klávesnice.
2. Pre vstup z klávesnice N, vypočítajte $1^3+2^3+3^3+...N^3$.
3. Pre vstup z klávesnice N, vypíšte všetky štvorce (t.j. čísla ako 1 ($1=1^2$), 4 ($4=2^2$), 9 ($9=3^2$)...) menšie ako N.
4. Spočítajte sumu kladných čísel zadávaných z klávesnice. Ukončite keď je zadané záporné číslo a vypíšte sumu.
5. Vypočítajte mocninu a^b pre a,b načítané z klávesnice.
6. Spočítajte sumu sum = 1 + 1/2 + 1/4 + 1/8 + ... + 1/(2^100).
    1. Vypíšte sumu.
    2. Vypíšte všetky medzisúčty.
    3. Programovo zistite, v ktorej iterácii sa už hodnota nezmení.
7. Vypočítajte približnú hodnotu integrálu [0,Pi] pre funkciu sin(x),cos(x). Diskutujte ako.
8. Načítajte n z klávesnice a zistite, či je to prvočíslo.
9. Vypíšte binárny rozvoj celého čísla n.
10. Spočítajte ciferný súčet celého čísla n.
11. Zistite, pre ktoré x platí $x^2 + 2 = x^3 + 2*x + 1$.

CV.3 Podmienené príkazy

Zoznam

1. Vytvorte zoznam **list** 10-tich "náhodných" prvkov tvaru (2^i mod 113). (Môžete to urobiť for-cyklom s premennou i od 1 do 10.)
2. Vytlačte celý zoznam z úlohy 1. Načítajte číslo **k** a vytlačte posledných **k** položiek zoznamu.
3. Načítajte dve hodnoty **i,j** a vytlačte zoznam od **i**-tej po **j**-tu položku.
4. Nájdite a vypíšte maximum hodnôt zoznamu.
5. Zrotujte zoznam o jeden prvok doľava. O **X** prvkov doprava.
6. Nájdite najmenší prvok a pozíciu (index), na ktorej sa nachádza.
7. Pre zadané číslo nájdite jeho pozíciu v zozname, vráťte -1, ak sa v zozname nenachádza.

Polynóm

- Vypočítajte hodnotu polynómu zadaného zoznamom koeficientov ([1,0,0,2] predstavuje $2.x^3+1$) v bode a (načítané z klávesnice) .

Delenie intervalu

1. Nájdite odmocninu z reálneho čísla.
2. Nájdite koreň polynómu z predchádzajúcej úlohy.

Dalsie ulohy

1. Napíšte skript (program), ktorý v cykle číta čísla z klávesnice, až kým nezadá užívateľ 0. Po ukončení cyklu vypíše:

- počet načítaných císel
- počet načítaných záporných čísel
- súčet načítaných císel
- priemer načítaných čísel
- najmenšie načítané číslo
- druhé najmenšie načítané číslo
- či bolo načítané nejaké záporné číslo (true/false)

Pozn. Implementujte ho bez toho aby ste si ukladali všetky načítané čísla v zozname.

2. Napíšte skript (program), ktorý v cykle číta čísla z klávesnice, až kým nezadá užívateľ 0. Zadané čísla ukladá do zoznamu (list) pomocou funkcie append.

- Po ukončení cyklu:
- zistite a vypíšte počet prvkov v zozname
- vypíše celý zoznam
- zistite či zoznam obsahuje duplicity

3. Modifikujte predchádzajúcu úlohu tak, aby zoznam neobsahoval duplicity, teda zadané číslo vložíte do zoznamu, len ak sa v ňom ešte nie je.

CV.4 Zoznamy a vnorene cykly

Úlohy - Vnorené cykly a zoznam

1. Naimportujte modul random. Funkcia random.randint(a,b) vygeneruje nahodne cislo z intervalu <a,b>. Pomocou funkcie random.randint() vytvorte zoznam **list**, ktory bude obsahovať 10 náhodných prvkov z intervalu <0,9>.
2. Nacitajte z klavesnice cislo n. Pomocou for-cyklu zistite, kolkokrat sa n nachadza v zozname **list**.
3. Vytvorte zoznam **unique**, ktorý bude obsahovať každý prvok zo zoznamu **list** práve raz. Vyuzite pri tom program z predchadzajucej ulohy (prvok **a** zo zoznamu **list** pridame do zoznamu **unique** iba, ak sa **a** nachadza v zozname **unique** 0-krat).
4. Pre každý prvok zoznamu **unique** zistite (uložte si do ďalšieho zoznamu **frequency**), koľkokrát sa daný prvok vyskytuje v zozname **list**.
5. Vykreslite histogram vodorovne.

```
pre list = [0,0,1,2,2,1,0,1,1] sa vypise
  0 ***
  1 ****
  2 **
  (na poradi hodnot nezalezi)
```

Úlohy - Ladenie a Eratostenovo sito

1. Pozrite si tutorial k ladeniu v prostredi IDLE: https://www.cs.uky.edu/~keen/help/debug-tutorial/debug.html
2. Odladťe Eratostenovo sito zo slajdu na strane 22 - prednáška 4.
3. Pozrite si prednášku 4 (potom odložiť a pozrieť len keď budete v koncoch) a pokúste sa porozumieť základným myšlienkam a naprogramujte Erastotenovo sito od zaciatku.

CV.5 Zoznamy a vnorene cykly

Úlohy - Kreslenie

1. Pre vstup z klavesnice **n** vykreslite stvorec šírky **n**.

```
n=3
***
***
***
```

2. Pre vstup z klavesnice **n** vykreslite trojuholnik šírky **n**.

```
n=3
*
**
***
```

3. Pre vstup z klavesnice **n** vykreslite vlajku šírky **n**.

```
n=3
*
**
***
**
*
```

4. Pre **n** vykreslite pyramídu výšky **n**.

```
n=3
  *
 ***
*****
```

5. Pre **n, m** vykreslite pyramídu výšky **n** s meniacou sa hodnotou vzoru. Pre hodnoty > 10 vypíšte len poslednú číslicu napr 23456789012....

```
n=3, m=2
  2
 234
23456
```

6. Pre **n, m** vykreslite kríž veľkosti **n** (n je vzdy neparne) s meniacou sa hodnotou vzoru. Pre hodnoty > 10 vypíšte len poslednú číslicu napr 23456789012....

```
n=5, m=3
  5
  4
54345
  4
  5
```

7. Pre neparne **n** vykreslite písmeno X veľkosti(šírky a výšky) **n**.

```
n=5
*   *
 * *
  *
 * *
*   *
```

Úlohy - Vnorene cykly a zoznamy

Simulácia opilca: opilec sa potáca medzi domovom a krčmou. Jeho krok je dĺžky 1 ale v náhodnom smere. Úlohou je zistiť aká je pravdepodobnosť, že dostane domov, do krčmy či zaspí na ceste po m krokoch.

1. Naprogramujte si "mesto" s dĺžkou cesty **n** (predpokladajte, ze n je neparne) medzi domovom a krčmou.

```
n = 11 - cesta dĺžky 11. (=11x .)
HOME..........PUB
```

2. Umiestnite opilca **O** do stredu mesta a vykreslite mesto.

```
HOME.....O.....PUB
```

3. Urobte náhodný pohyb opilca **O** o 1 a vykreslite. (Pouzite modul random.)

```
HOME.....O.....PUB
HOME....O......PUB
```

4. Opakujte kým nepríde domov, do krčmy alebo po 20 krokoch nezaspí.

```
HOME.....O.....PUB
HOME....O......PUB
HOME.....O.....PUB
HOME......O....PUB
```

5. Zopakujte 2 krát a zistite koľko krát prišiel domov.
6. Zopakujte 1000 krát a zistite koľko krát prišiel domov, do krčmy a zaspal.

Úlohy - Histogram

Cieľom je pre náhodnú vzorku dát si spočítať štatistiky výskytu jednotlivých prvkov.

1. Vygenerujte náhodný zoznam **list** s 100 číslami z intervalu <0,20> - opakovane generujte náhodné hodnoty.
2. Vytvorte zoznam **unique**, ktorý bude obsahovať každý prvok zo zoznamu **list** práve raz. Ako zabezpečíte, že prvok zoznamu sa pridá do unique len raz?
3. Zistite koľkokrát sa nachádza v zozname **list** prvý prvok z **unique** a vytlačte .
4. Vytvorte zoznam **frequency** ktorý bude obsahovať početnosť výskytu jednotlivých prvkov zozname **'list**. **Frequency[0]** zodpovedá početnosti výskytu
    - prvku **unique[0]** v zozname **list** atď.
5. Vykreslite histogram vodorovne.

```
pre list = [0,0,1,2,2,1,0,1,1] sa vypise
  0 ***
  1 ****
  2 **
  (na poradi hodnot nezalezi)
```

CV.6 Funkcie

Sekcia 1: Mocniny

1. Definujte funkciu druha_mocnina, ktora pre vstup a vrati hodnotu a*a.
2. Pomocou funkcie druha_mocnina vypiste druhe mocniny cisel 1,2,3,4,5.
3. Definujte funkciu nta_mocnina, ktora pre vstupy a a n vrati hodnotu an . (aj a aj n budu vstupmi funkcie).
4. Pomocou funkcie nta_mocnina vypiste cisla 21 , 22 , 23 , ... , 210 . Pomocou funkcie nta_mocnina definujte funkciu male_mocniny, ktora pre vstupy a a hranica vypise vsetky mocniny cisla a mensie ako cislo hranica.

Sekcia 2: Prvocisla

1. Definujte funkciu delitelnost, ktora pre vstupy a a d vrati hodnotu True, ak a%d==0, a hodnotu False, ak a%d>0. Pomocou funkcie delitelnost vypiste vsetky delitele cisla 12. (Delitele cisla 12 su 1, 2, 3, 4, 6, 12.)

2. Cislo a vacsie ako 2 je prvocislo, ak pre kazde cislo d z mnoziny {2, 3, … a-1} plati, ze a%d>0. Pomocou funkcie delitelnost definujte funkciu test_prvociselnosti, ktora pre vstup a vrati hodnotu True, ak a je prvocislo, a hodnotu False, ak a nie je prvocislo. Mozete predpokladat, ze a>2. Pomocou funkcie test_prvociselnosti vypiste vsetky prvocisla vacsie ako 2 a mensie ako 50. Upravte funkciu test_prvociselnosti tak, aby funkcia vratila hodnotu True aj v pripade, ze a=2. Pomocou upravenej funkcie test_prvociselnosti vypiste vsetky prvocisla mensie ako 50. (cislo 2 je najmensie prvocislo)

Sekcia 3: Zakladna enumeracia

Vyrieste ulohy zo sekcie Zakladna enumeracia na stranke:

http://www.uim.elf.stuba.sk/kaivt/Predmety/B-ALPRI/C4

Sekcia 4: Dalsie ulohy

1. Napíšte funkciu s 3 parametrami: from, to, step, a funkcia vypíše čísla začínajúc číslom from, s krokom step, tak aby nebola prekročená hranica to. Predpokladajte, ze step je kladne cislo.
2. Upravte funkciu z prikladu 1 tak, aby step mohlo byť aj záporné číslo.
3. Napiste funkciu, ktora vypise prvych N členov aritmetickej postupnosti a_{i+1} = a_{i} + dna základe parametrov a_{0}, d a N.
4. Napiste funkciu, ktora vypise prvych N členov geometrickej postupnosti a_{i+1} = a_{i} *r na základe parametrov a_{0}, r a N.
5. Tazsia uloha: Napiste funkciu, ktora vypise prvych N členov geometrickeho radu, kde i-ty člen radu predstavuje súčet prvých i členov geometrickej postupnosti.
6. Tazsia uloha: Vytvorte program, ktorý názorne demonštruje, že geometricky rad konverguje pre r v absolútnej hodnote menšej ako 1 ku a_0/(1-r).

CV.9 List, Set, Dictionary

Úlohy:

1. Vezmite ľubovoľný text z novín (minimálne 10 riadkov) priraďte ho do premennej **text**.
   - (napr. text = "jaj kde bolo")
2. Spočítajte koľko je rôznych znakov v texte. **Návod**: pridávajte postupne znaky do nejakej množiny.
3. Spočítajte koľko je rôznych dvojíc znakov.
4. Zistite maximálnu veľkosť slova max_size v texte a priemernú dĺžku slova (použite **dictionary**).
5. Zistite koľko dvojíc písmen typu "AB", "de" (t.j. dvojice, kde znaky za sebou nasledujú v ASCII tabuľke) sa v texte nachádza a vypíšte ich.

Substitučná šifra

1. Text z reťazca **text** náhodne spermutujte (písmená) a uložte do premennej **perm_text**. (napr. perm_text. = alejasdh...)
2. Vytvorte kľúč substitučnej šifry tak, že text sa bude šifrovať tak, že dané písmeno sa bude vždy šifrovať podľa prvého "neproblematického" výskytu.

```
kľúč: text      = jej kke
      perm_text = alejasd
     j   --> a
     e   --> l
     j   --> e XXX - problém (j-->a už dané tj preskakujem a idem ďalej)
     " " --> j
     k   --> a (vzor pre a už existuje - ďalej)
     k   --> s
     ...
```

1. Zašifrujte (nahraďte každé "j" za "a", "e" za "l") text podľa skonštruovaného kľúča.
2. Dešifrujte podľa kľúča -- môžete si vyrobiť opačný kľúč.
3. Zašifrujte kľúčom iný text z novín - niektoré písmená ktoré v kľúči chýbajú doplňte tak aby pôvodné mapovanie zostalo a doplnené písmená sa náhodne spermutovali.

CV.10 Práca so súborom

Úlohy:

1. Vygenerujte textový súbor **data.txt** s náhodnými celými číslami.
2. Zistite dĺžku súboru.
3. Načítajte textový súbor **data.txt** a vypočítajte priemer jeho hodnôt.
4. Vygenerujte textový súbor **data2.txt** s nasimulovanými výsledkami študentov zo skúšky.
   - Každý riadok súboru bude tvaru **meno;** (náhodný reťazec dĺžky 2) **počet bodov** (náhodný počet bodov < 101). Vytvorte vlastnú funkciu na vygenerovanie náhodného reťazca danej dĺžky (size - parameter). Použite **chr** a **ord("A")**.
5. Utrieďte riadky podľa počtu bodov a zapíšte výsledok do súboru.
6. Utrieďte riadky podľa mena a v prípade rovnosti podľa počtu bodov. Zapíšte výsledok do súboru.
7. Utrieď riadky z data2.txt podľa počtu bodov a v prípade rovnosti podľa "otočeného" mena (meno odzadu).
   - Upravte buble sort s tým, že budete mať vlastnú funkciu na porovnanie riadkov (tzv. comparator).

Ďalšie príklady

- [https://www.w3resource.com/python-exercises/file/index.php]

Cvičenie 11

Reťazce

Príklad 1:
Bez použitia metódy count() naprogramujte funkciu s nasledovnými vlastnosťami. Funkcia má jeden parameter: reťazec ret. Funkcia vráti, koľkokrát sa znak 'a' nachádza v reťazci ret.

Príklad 2:
Bez použitia metódy find() naprogramujte funkciu s nasledovnými vlastnosťami. Funkcia má dva parametre: znak zn a reťazec ret. Ak sa znak zn vyskytuje v retazci ret, potom funkcia vráti index pozície, na ktorej sa vyskytuje prvý krát. Inak vráti hodnotu -1.

Príklad 3:
Definujte funkciu F s nasledujúcimi vlastnosťami. F má jeden parameter: zoznam reťazcov. F vráti ten reťazec zo zoznamu, v ktorom sa znak 'a' vyskytuje najčastejšie. Nepoužite metódu count(). Namiesto toho použite funkciu z Prikladu 1.

Priklad 4:
Definujte funkciu F s nasledujúcimi vlastnosťami. F má jeden parameter: zoznam reťazcov. Funkcia vráti prvý reťazec, ktorý obsahuje znak 'a'. Ak žiaden reťazec neobsahuje znak 'a', funkcia vráti hodnotu 0.

Množiny

Príklad 1:
Definujte funkciu F1 s nasledujúcimi vlastnosťami. F1 má dva parametre: premennú prvok a množinu s1. F1 vráti hodnotu True, ak sa obsah premennej prvok nachádza v množine s1, inak vráti hodnotu False. V definícii funkcie F1 nesmiete použiť výraz „prvok in s1"! Namiesto toho použite for-cyklus.

Príklad 2:
Definujte funkciu F2 s nasledujúcimi vlastnosťami. F2 má dva parametre: množinu s1 a množinu s2. F2 vráti prienik týchto množín. V definícii funkcie F2 nesmiete použiť výraz „s1&s2"! Namiesto toho použite for-cyklus. Takisto nesmiete použiť výraz typu „prvok in s1". Môžete ale použiť funkciu F1 z Príkladu 1.

Príklad 3:
Definujte funkciu F3 s nasledujúcimi vlastnosťami. F3 má dva parametre: množinu s1 a množinu s2. F3 vráti rozdiel množín s1-s2. V definícii funkcie F3 nesmiete použiť výraz „s1-s2"! Namiesto toho použite for-cyklus. Takisto nesmiete použiť výraz typu „prvok in s1". Môžete ale použiť funkciu F1 z Príkladu 1.

Príklad 4:
Definujte funkciu F4 s nasledujúcimi vlastnosťami. F4 má dva

parametre: množinu s1 a množinu s2. F4 vráti zjednotenie množín s1 a s2. V definícii funkcie F3 nesmiete použiť výraz „s1|s2“! Namiesto toho použite for-cyklus.

Reťazce a množiny

Príklad 1:
Definujte funkciu F s nasledujúcimi vlastnosťami. F má dva parametre: reťazec ret1 a reťazec ret2. F vráti hodnotu 1, ak sa niektorý znak z reťazca ret1 nachádza v reťazci ret2. V opačnom prípade vráti F hodnotu 0.

Príklad 2:
Definujte funkciu F s nasledujúcimi vlastnosťami. F má dva parametre: reťazec ret1 a reťazec ret2. F vráti hodnotu 1, ak sa každý znak z reťazca ret1 nachádza v reťazci ret2. V opačnom prípade vráti F hodnotu 0.

Príklad 3:
Definujte funkciu F s nasledujúcimi vlastnosťami. F má dva parametre: reťazec ret1 a reťazec ret2. F vráti počet rôznych znakov, ktoré sa súčasne vyskytujú aj v reťazci ret1 aj v reťazci ret2.

Príklad 4:
Definujte funkciu F s nasledujúcimi vlastnosťami. F má jeden parameter: reťazec ret. F vráti počet znakov, ktoré sa v reťazci ret vyskytujú viac ako raz. Otestujte funkciu na reťazci “mama”. Funkcia by mala vrátiť hodnotu 2.

Príklad 5: (z cvičenia 9)
Definujte funkciu F s nasledujúcimi vlastnosťami. F má jeden parameter: reťazec ret. F vráti počet rôznych dvojíc znakov v reťazci ret.

Cvičenie 12

Slovníky

Ak si nepamätáte, ako sa pracuje so slovníkmi pozrite si poslednú prednášku alebo stránku:

https://snakify.org/lessons/dictionaries_dicts/

Úloha 1:
Uvažujme zoznam, ktorého prvky sú reťazce 'A' až 'FX' v ľubovoľnom poradí a s ľubovoľným počtom opakovaní (aj 0-krát). Napríklad:

```
L=['FX', 'FX', 'E', 'FX', 'B', 'FX', 'E', 'E']
```

Časť A:
Definujte funkciu F1 s nasledujúcimi vlastnosťami. F1 má jeden parameter: zoznam tvaru, ako je popísané vyššie. F1 vráti slovník, ktorého kľúče budú známky 'A' až 'FX', a ktorého hodnoty budú počty, koľkokrát sa daný kľúč nachádza v zozname.

Časť B:
Definujte funkciu F2 s nasledujúcimi vlastnosťami. F2 má jeden parameter: slovník d (predpokladáme, že hodnoty v slovníku sú čísla). F2 vykreslí histogram prislúchajúci k slovníku. Napríklad pre slovník d=F1(L), ktorý dostaneme po aplikovaní funkcie F1 z časti A na zoznam L, nám funkcia F2 vypíše:

```
A:
B:*
C:
D:
E:***
FX:****
```

Úloha 2:
Máme výsledky minulej sezóny anglickej futbalovej ligy uložené ako slovník. Kľúče v slovníku sú mená tímov a hodnoty sú počty bodov získané za sezónu. Napríklad:

```
d={'Chelsea':93, 'Liverpool':76, 'Arsenal':75, 'Hull City':34,
'Sunderland':24}
```

Definujte funkciu nova_sezona(d,postupujuci), ktorá má nasledujúce vlastnosti. Funkcia má dva parametre: slovník d, v ktorom sú výsledky z minulej sezóny a zoznam postupujuci, ktorý obsahuje mená dvoch tímov, ktoré postúpili do ligy z nižšej súťaže (napríklad postupujuci=['Watford','Southampton']). Funkcia vymaže zo slovníka d dva tímy s najnižším bodovým ziskom, doplní do slovníka obidva tímy zo zoznamu postupujúci a nastaví body každého tímu v slovníku na 0.

Triedenie a binary search

Úloha 1 (insertion sort):
Definujte funkciu F1 s nasledujúcimi vlastnosťami. F1 má jeden parameter: zoznam čísel. Funkcia utriedi zoznam od najmenšieho čísla po najväčšie pomocou insertion sortu. To znamená: Naprogramujte insertion sort! Ak si už nepamätáte, ako insertion sort funguje, pozrite si toto video:

https://www.youtube.com/watch?v=uMqVuEEWJv4

Úloha 2 (bubble sort):
Definujte funkciu F1 s nasledujúcimi vlastnosťami. F1 má jeden parameter: zoznam čísel. Funkcia utriedi zoznam od najmenšieho čísla po najväčšie pomocou bubble sortu. To znamená: Naprogramujte bubble sort! Ak si už nepamätáte, ako bubble sort funguje, pozrite si toto video:

https://www.youtube.com/watch?v=18OO361--1E&t=153s

Úloha 3 (selection sort):
Definujte funkciu F1 s nasledujúcimi vlastnosťami. F1 má jeden parameter: zoznam čísel. Funkcia utriedi zoznam od najmenšieho čísla po najväčšie pomocou selection sortu. To znamená: Naprogramujte selection sort! Ak si už nepamätáte, ako selection sort funguje, pozrite si toto video:

https://www.youtube.com/watch?v=R_f3PJtRqUQ

Úloha 4 (binary search):
Definujte funkciu F1 s nasledujúcimi vlastnosťami. F1 má dva parametre: zoznam čísel L utriedený od najmenšieho čísla po najväčšie a číslo N. Funkcia zistí, či sa číslo N nachádza v zozname L pomocou metódy binary search. To znamená: Naprogramujte binary search! Ak si už nepamätáte, ako binary search funguje, pozrite si toto video:

https://www.youtube.com/watch?v=tLMuuy_xoyM