Kolokwium poprawkowe 1 i 2 ze Wstępu do Informatyki

23.01.2016

Czas trwania: 180 min. Każde zadanie oddajemy na osobnej kartce, podpisanej imieniem, nazwiskiem, kierunkiem studiów i numerem indeksu. Oddanie zadania z kol. I (1,2,3) lub II (4,5,6) skutkuje skasowaniem dotychczasowej punktacji z tego kolokwium.

Kolokwium 1

1. (3p.) Napisz funkcję `najdluzszy(L1, L2)`, która dla zadanych dwóch list `L1` i `L2`, zwróci najdłuższy wspólny spójny fragment. Jeśli listy nie mają wspólnego fragmentu to zwróci pustą listę.

   Przykład:
   `L1 = [5, 2, 3, 8, 5, 7, 6]`
   `L2 = [3, 8, 4, 3, 8, 5, 7, 3]`
   Wynik: `[3, 8, 5, 7]`

2. (3p.) Napisz funkcję `macierz_par(lista_par)`, która dla danej listy `lista_par` liczb naturalnych (i, j), takich że $i, j \leq n$, zwróci listę n list długości n (reprezentującą macierz) M, taką, że $M[i][j] == 1$ jeśli para (i, j) lub (j, i) znajduje się w liście `lista_par`, a $M[i][j] == 0$ w przeciwnym wypadku.

   Np. dla listy par: `krawedzie = [(0,1),(1,2),(1,3),(2,3)]` odpowiadająca jej macierz par (w postaci listy list):

```
>>> macierz_par(krawedzie) =
[[0,1,0,0],
[1,0,1,1],
[0,1,0,1],
[0,1,1,0]]
```

3. (4p.) Fragment kosmosu wyznaczany przez pewien obiekt kosmiczny możemy przedstawić jako listę dwu-elementową, w której na pozycji pierwszej (zerowej) jest nazwa obiektu, a na pozycji drugiej lista obiektów krążących wokół obiektu (przedstawionych w ten sam sposób). Np. dla Układu Słonecznego być może byłaby to taka lista:

```
uklad_sloneczny =['Slonce' , [
    [ 'Merkury' , [ ] ] ,
    [ 'Wenus' , [ ] ] ,
    [ 'Ziemia' , [
        [ 'Ksiezyc' , [
            ['Lunar Reconnaissance Orbiter' , [ ]] ,
        ],
        [ 'Miedzynarodowa Stacja Kosmiczna' , [ ] ]
        ] ... ] ,
    ], ...
    [ 'Saturn' , [
                [ 'Tytan' , [ ] ... ] ,
                ]
    ], ...
     [ 'Dziewiata planeta ' , [ ] ]
    ]
]
```

   Napisz funkcję adres(uklad, obiekt), która zwróci listę obiektów, wokół których krąży (pośrednio) dany obiekt, lub listę pustą, jeśli takiego obiektu nie ma. Np.

```
adres(uklad_sloneczny, 'Tytan') = ['Slonce', 'Saturn']
adres(uklad_sloneczny, 'Lunar Reconnaissance Orbiter') = ['Slonce', 'Ziemia', 'Ksiezyc']
```

Kolokwium 2

4. (3p.) Załóżmy, że mamy w plikach tekstowych zapisaną informację o położeniu genów na chromosomach w następującym formacie: "ID_genu chromosom początek koniec", przy czym dla każdego gatunku mamy osobny plik. Napisz dwie funkcje: `wczytaj(nazwapliku)` i `wspolne(gatunek1,gatunek2)`, które odpowiednio pozwolą wczytać informację z pliku do słownika a następnie wyszukać geny, które posiadają taki sam identyfikator oraz znajdują się na tym samym chromosomie i ich położenia na chromosomie mają niepuste przecięcie. Liczby całkowite "początek" i "koniec" w pliku wyznaczają numery nukleotydów kodujące dany gen (włącznie). Powiemy, że przecięcie jest niepuste, jeśli choć jeden nukleotyd w obu genach ma ten sam numer.

5. Co zostanie wypisane po wykonaniu następujących programów:
   (2p.) **a.py**

```
l1 = [1,2,3,3,4]
l2 = [3,4,4,4,5,6]
def foo(a,b):
        s1 = set(l1)
        s2 = set(l2)
        return s1 & s2, s1, s2
print foo(l1,l2)
```

   (2p.) **b.py**

```
d = {'a' : [1,2,3]}
def foo(a,b):
        if len(a[b]) > 0:
                c = {b : [a[b][-1]], a[b][-1] : a[b][:-1]}
                e = foo(c,a[b][-1])
                for k,v in e.items():
                        c[k] = v
                return c
        else:
                return {b : ['y']}
print foo(d,'a'), d
```

6. (3p.) W pliku `geny.txt` znajduje się tekst, w którym pojawiają się również nazwy genów u różnych gatunków organizmów. W zależności od gatunku, nazwa genu może mieć różny format

   - dla muszki owocowej nazwa zaczyna się na FBgn, a następnie jest numer genu (np FBgn0032283)
   - dla człowieka: ENSG, a mastępnie 11-cyfrowy identyfikator genu: np ENSG00000210049
   - dla pozostałych gatunków, np. szczekuszki amerykańskiej (łac. Ochotona princeps): ENS, potem unikalny, 3-literowy skrót łacińskiej nazwy, następnie G (jak gen) oraz maksymalnie 12-cyfrowy numer genu (np. ENSOPRG00000018879).

   Napisz funkcję `geny_gatunkow()`, która używając wyrażeń regularnych przeszuka plik geny.txt i zwróci listę z listami nazw genów, pogrupowanymi wg. gatunku, z którego pochodzą. Kolejność genów w listach oraz kolejność list w wynikowej liście nie ma znaczenia.

   Np. dla pliku `geny.txt` o treści:

```
Przykładami genów u człowieka są geny: ENSG00000070444 oraz ENSG00000099977, natomiast u szczekuszki są :ENSOPRG00000000637 i ENSOPRG00000000040. Alpaka (lac. vicugna pacos) i muszka owocowa mają natomiast geny odpowiednio: ENSVPAG00000018879, FBgn0032283. Alpaka ma dodatkowo gen ENSVPAG00000002500.
```

```
>> print geny_gatunkow()
[['ENSG00000070444', 'ENSG00000099977'],
['ENSOPRG00000000637', 'ENSOPRG00000000040'],
['ENSVPAG00000018879', 'ENSVPAG00000002500'],
['FBgn0032283']]
```

Powodzenia!