PYTHON

Osnove 5

Python urejevalnik se nahaja na strani:
`https://www.w3resource.com/python-exercises/python-basic-exercises.php#EDITOR`

1. **Vsota prvič** Z uporabo števil iz danega seznama in dane vsote določite na koliko načinov lahko zapišemo vsoto, če se števila iz seznama lahko poljubno ponovijo in vrstni red je pomemben.

   Nato nalogo prilagodite tako, da bo rezultat seznam vseh možnih načinov.

   primer

   ```
   seznam = {1, 2} vsota = 5
   res = 8

   1+1+1+1+1
     1+1+1+2   1+1+2+1   1+2+1+1   2+1+1+1
       1+2+2     2+1+2     2+2+1
   ```

2. **Vsota drugič** Z uporabo števil iz danega seznama in dane vsote določite na koliko načinov lahko zapišemo vsoto, če se števila iz seznama lahko poljubno ponovijo in vrstni red ni pomemben.

   Nato nalogo prilagodite tako, da bo rezultat seznam vseh možnih načinov. **primer**

   ```
   seznam = {1, 2} vsota = 5
   res = 3

   1+1+1+1+1  1+1+1+2  1+2+2
   ```

3. **Virahanka** Virahanka je bil indijski matematik prozodist. Živel je v 6. stoletju našega štetja.

   Kaj je prozodija? V Wikipediji preberemo:
   Verzologija (stihoslovje, metrika, prozodija) je teorija verza, del literarne teorije.

   Glej `https://www.youtube.com/watch?v=nQR1NY03zIA`.

   Virahanka je 500 let pred Fibonaccijem omenil zaporedje, ki danes nosi Fibonaccijevo ime. Nanj je naletel pri stihoslovju. Zanimali so ga različni vzorci, sestavljeni iz kratkih in dolgih zlogov. Dolgi zlogi *guru* so bili dvakrat daljši od kratkih *lahu*. Zanimalo ga je, koliko je različnih vzorcev dane dolžine, sestavljenih iz dolgih in kratkih zlogov. Odgovor na to vprašanje ga je pripeljal do Fibonaccijevega zaporedja. Označimo kratki zlog z `L` in dolgi z `G`. Obstaja en sam vzorec dolžine kratkega zloga, dva dolžine dolgega zloga itd. Poglejmo:

```
1: L
2: LL, G
3: LG, LLL, GL
4: LLG, GG, LGL, LLLL, GLL
5: LGG, LLLG, GLG, LLGL, GGL, LLLLL, GLLL
```

Nize dolžine $n+1$ dobimo tako, da na konec vsakega niza predzadnje dolžine $n-1$ dodamo dolg zlog, tem pa še dodamo nize, ki jih dobimo, če na konec vsakega niza zadnje dolžine n, dodamo kratek zlog. Število nizov dolžine $n+1$ dobimo kot vsoto števila nizov dolžine n in nizov dolžine $n-1$. To je pa ravno definicija Fibonaccijevega zaporedja. Vseh vzorcev dolžine $n+1$ je enako $F_{n+1} = F_n + F_{n-1}$.

Napiši program, ki sprejme dolžino in izpiše vse vzorce dane dolžine.

4. **Produkt ciklov v permutaciji** Permutacije so spreminjanje vrstnega reda elementov. Elemente označimo s številkami 1 do n. Permutacijo zapišemo v obliki dvovrstične matrike. V prvi vrstici so zapisani elementi. V naravnem vrstnem redu v drugi pa je zapisan nov vrstni red, ki določa permutacijo.

$$\begin{pmatrix} 1 & 2 & 3 & \dots & n \\ x_1 & x_2 & x_3 & \dots & x_n \end{pmatrix}$$

Permutacijo se da zapisat v obliki zaporedja (produkta) ciklov.
Poglejmo primer:

$$a = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 3 & 5 & 4 & 7 & 8 & 6 & 1 & 2 \end{pmatrix}, \quad a = (1,3,4,7)(2,5,8)(6)$$

Poglejmo katere zamenjave pripeljejo do zgornje permutacije. Začnemo z 1 v zgornji vrstici, ta se zamenja s 3. Zato je pod 1 zapisana 3 v drugi vrstici. Število 1, ki se sedaj nahaja v na mestu 3 se zamenja s 4. Od tod 4 na tretjem mestu itd. Tako sta 3 in 4 že na pravih mestih. Nadaljujemo z 2 in tako naprej.

```
1. Cikel 1 -> 3 -> 4 -> 7 -> 1 cikel se je zaključil
2. Cikel 2 -> 5 -> 8 -> 2
3. Cikel 6 -> 6
```

Prvi cikel dobimo z lihim številom zamenjav, je liha permutacija medtem, ko je drugi soda permutacija. Število 6 ostane ne svojem mestu, torej nič zamenjav, zato je ta cikel soda permutacija.
`Naloga:` Razstavimo dano permutacijo na produkt ciklov.

5. **Sto zapornikov**

`https://www.youtube.com/watch?v=iSNsgj1OCLA`

Direktor zapora ponudi 100 obsojenim priložnost pomilostitve. Vsak zapornik je označen s številko med 1 in 100. V sobi s 100 predali, ki so ravno tako označeni s številkami od 1 do 100, direktor naključno vstavi po eno zapornikovo številko v vsak predal. Zaporniki drug za drugim vstopajo v sobo. Vsak zapornik lahko odpre in pogleda 50 predalov v poljubnem vrstnem redu. Po tem se predali ponovno zaprejo. Če med iskanjem vsak zapornik v enem od predalov najde svojo številko,

so vsi zaporniki pomiloščeni. Če samo en zapornik ne najde svoje številke ni pomiloščen nobeden. Preden prvi zapornik vstopi v sobo, se lahko zaporniki pogovarjajo o strategiji, po tem pa ne morejo več komunicirati. Kakšna je najboljša strategija zapornikov?

Če vsak zapornik odpre 50 predalov naključno je verjetnost, da najde svojo število enaka 1/2. Verjetnost, da bi vsi zaporniki našli svojo številko je

$$(1/2)^{100} \approx 0.00000000000000000000000000008.$$

Strategija Vsak izbere najprej predal s svojo številko, pogleda vanj in se odpravi k predalu s številko, ki jo je našel v tem predalu in tako naprej, slej ko prej bi našel v predalu svojo številko in cikel bi bil zaključen. Verjetnost, da bodo zaporniki pomiloščeni je enaka verjetnosti, da pri dani permutaciji ni noben cikel daljši od 50, ta pa je približno 0.31.

Naloga Naredite simulacijo tako, da pri zaporedju naključnih permutacij iščete delež tistih, pri katerih so vsi cikli krajši ali enaki 50.

6. **Rojstni dan** Vprašamo se, kolika je verjetnost, da imata vsaj dva dijaka v razredu s 23 dijaki rojstni dan na isti dan.

   Definicija problema Nalogo rešujemo s simulacijo. Med dijake naključno razdelimo datume rojstva. Nalogo poenostavimo tako, da jim razdelimo namesto datumov kar naravna števila med 1 in 365. Simulacijo poženemo 100 krat in štejemo porazdelitve, kjer se razdeljena števila ponovijo.

Rešitve 5

Vsota prvič

```python
#!/usr/bin/env python3

def n_nac(lst, n):    # lst je seznam sumandov n je največja vsota pomemben je
↪  vrstni red
    res = []
    j = 1;
    while j <= n:     # j <= n  gledamo vsote j
        val = 0
        for i in lst: # pregledamo sumande i
            c = j - i # c = j - i
            if c > 0:
                val += res[c - 1] # povečamo število vsot j za res[c - 1]
        if lst.__contains__(j):   # c - 1 zato, ker je na res[0] število vsot 1
            val += 1              # če pa je j v lst pa dodamo še tega ^^I
        res.insert(j,val)         # novo iztračunano število vsot j dodamo v res.
        j+=1
    return res                    # res[i] je enako število vsot j s sumandi lst

def l_nac(lst, n):
    res = []
    j = 1
    while j <= n:
        vals = []
        for i in lst:
            c = j - i
            if c > 0:
                val = res[c - 1]
                val = [x + [i] for x in val]
                vals = vals + val
        if lst.__contains__(j):
            vals +=  [[j]]
        res.insert(j, vals)
        j += 1
    return res

if __name__ == '__main__':
    lst = [2, 3]
    n = 5
    print(n_nac(lst, n))
    print(l_nac(lst, n))
```

Vsota drugič

```python
#!/usr/bin/env python3
def n_nac(lst, n):    # lst je seznam sumandov n je največja vsota vrstni red ni
↪  pomemben
    nac = [1] + [0 for i in range(n)]
    for i in lst:
        for j in range(i, n + 1):
            c = j - i
            nac[j] += nac[c]
    return(nac[1:])

def list_nac(lst, n):
    nacini = []
    j = 1
    while j <= n:
        vals = []
        for i in lst:
            c = j - i
            if c > 0:
                val = nacini[c - 1]
                val = [x + [i] for x in val if x[-1] <= i]
                vals = vals + val
        if lst.__contains__(j):
            vals += [[j]]
        nacini.insert(j, vals)
        j += 1
    return(nacini)

if __name__ == '__main__':
    #n = int(input('n -> '))
    lst = [1,2]
    n = 5
    print (n_nac(lst, n))
    print (list_nac(lst, n))
```

Virahanka

```python
#!/usr/bin/env python3

v = [["l"], ["ll", "g"]]

n = int(input('Globina -> '))

for _ in range(n):
    less = [x + "l" for x in v[-1]]
    more = [x + "g" for x in v[-2]]
    v.append(less + more)
print(v)
```

perm_cycles.py

```python
#!/usr/bin/env python3
def to_cycles(s):
    ds = dict(enumerate(s))
    cycles = []
    while ds:
        si = iter(ds)
        cycle = []
        it = next(si)
        while it in ds:
            cycle.append(ds[it])
            jt = ds[it]
            del ds[it]
            it = jt
        cycles.append(cycle)
    return cycles
```

100 zapornikov

```python
#!/usr/bin/env python3
import random
from perm_cycles import to_cycles

random.seed()
def statistics(n):
    p = 0
    for _ in range(n):
        s = list(range(100))
        random.shuffle(s)
        c = to_cycles(s)
        if max([len(x) for x in c]) <= 50:
            p += 1
    return p

if __name__ == '__main__':
    n = int(input('n -> '))
    print(statistics(n))
```

Rojstni dan

```python
#!/usr/bin/env python3
import random

def deli(n):
    ucenci = [random.randint(1,356) for x in range(n)]
    if len(ucenci) != len(set(ucenci)):
        return True
    else:
        return False

if __name__ == '__main__':
    res = 0
    for _ in range(1000):
        if deli(23):
            res += 1
    print(res)
```