


Pénzösszegek feladatmegoldás

Függvény: lehetséges_összegek

```
C: > Zsuzsa > SZTE digit > algoritmusok > github >  penzosszegek.py > ...  
1  def lehetséges_összegek ( n , érmék ) :  
2      # Halmaz az összes lehetséges összeg követésére  
3      összegeket = {0}  
4  
5      érmékben lévő érmék esetében :  
6      új_összegek = beállítva ()  
7      s - re összegben :  
8      új_összegek.add ( s + érme )  
9      összegeket.frissítés ( new_sums )  
10  
11     # Távolítsuk el a 0-t és rendezzük  
12     összegeket.eldobni ( 0 )  
13     rendezett_összegek = rendezve ( összegek )  
14  
15     return len ( sorted_sums ), rendezett_sums
```

Paraméterek: n: Az érmefajták száma; érmék: Az érmék értékeinek listája.
Változók inicializálása: összegeket = {0}: Egy halmazt használunk az összes elérhető összeg tárolására. Kezdetben csak a 0 szerepel, mert bármilyen értéket hozzáadva kezdőként a nulláról indulunk. Érmék iterációja: A külső ciklus minden érme értéket (érme) végigvesz:

```
5      érmékben lévő érmék esetében :  
6      új_összegek = beállítva ()
```

Minden érme értékhez létrehozunk egy új halmazt (új_összegek), amelyben az adott érme hozzáadásával keletkező új összegek szerepelnek.

Összegek frissítése: Az aktuális halmaz (összegeket) minden lehetséges összegéhez hozzáadjuk az aktuális érme értékét:

```
7      s - re összegben :  
8      új_összegek.add ( s + érme )
```

Halmaz frissítése: Az újonnan kiszámított összegeket hozzáadjuk az eredeti halmazhoz:

```
9      összeget. frissítés ( new_sums )
```

Rendezés és visszatérés: A 0-t eltávolítjuk az összegekből:

```
12     összeget. eldobni ( 0 )
```

Az összegeket növekvő sorrendbe rendezzük:

```
13     rendezett_összegek= rendezve ( összegek )
```

A függvény visszaadja az elérhető összegek számát és a rendezett összeget:

```
15     return len ( sorted_sums ), rendezett_sums
```

Bemenetkezelés:

```
18     import sys
19     bemenet= sys . stdin . olvas
20     adat= bemenet (). osztott vonalak ()
21
22     n= int ( adat [ 0 ] )
23     értéket= lista ( térkép ( int , adat [ 1 ]. felosztás ()))
```

Bemenet olvasása: A `sys.stdin.olvas` a bemenetet egyetlen szöveggént olvassa be. A `osztott vonalak()` sorokra bontja a bemenetet. Értékek feldolgozása: Az első sor (`adat[0]`) az érték számát (`n`) tartalmazza. A második sor (`adat[1]`) az érték értékeit (értéket) tartalmazza, amelyeket listává alakítunk az `int` típus használatával.

Eredmény kiszámítása:

```
26     k, összegek = lehetséges_összegek ( n , érték )
27
28     # Kimenet kiírása
29     nyomtatás( k )
30     nyomtatás( " " . join ( map ( str , summas )
```

Függvény hívása: A `lehetséges_összegek` függvényt meghívjuk az érték listájával és az érték számával. Eredmény feldolgozása: `k`: Az elérhető

különböző összegek száma; összegek: Az összes lehetséges összeg növekvő sorrendben. Eredmény kiírása/nyomtatás: Kiírjuk az k értéket, amely az elérhető összegek számát adja meg. Kiírjuk/nyomtatás a summas értékeit szóközzel elválasztva. Ez a kód hatékonyan kiszámítja az k értékkel elérhető összes különböző összeget, és visszaadja a számukat és a konkrét értékeket. A halmazok használata biztosítja, hogy az összegek mindig egyediek legyenek, és a rendezés növekvő sorrendben adja vissza az eredményeket.