Bomboane – descrierea soluției

a) Împărțirea frățească

fie a[i] - numarul de bomboane din cutia i

Metoda 1 – 10% din punctajul cerinței

Se generează toate valorile pentru numărul de colegi și se verifică dacă numărul de bomboane din fiecare cutie împărțit la numărul de colegi dă același rest.

Complexitate O(N*max(a[i])

Metoda 2 – 30 % din punctajul cerinței

Se observă că numărul de bomoane care pot rămâne în fiecare cutie este cuprins între 0 și min(a[i]).

Complexitate O(N*min(a[i])

Metoda 3 – 100% din punctaj

fie a[i] - numarul de bomboane din cutia i

Trebuie determinat un numar k astfel incat fiecare a[i] sa dea acelasi rest la impartirea la k.

```
a[1] = x1 * k + r
a[2] = x2 * k + r
......
a[n] = x3 * k + r
```

Daca facem diferenta intre oricare doua elemente ale vectorului a[i] si a[j] obtinem:

`a[i] - a[j] = xi * k + r - (xj * k + r) = (xi - xj) * k`

Prin urmare diferenta intre doua elemente ale vectorului este multiplu de k.

Pentru k maxim acesta va fi `cmmdc(a[1] - a[2], a[2] - a[3], a[3] - a[4] ... a[n-1] - a[n]).`

Un algoritmul de rezolvare:

- pentru a evita valori negative se determina `min = min(a[1], a[2] ... a[n])`
- se scade min din elementele vectorului `a[1] = a[1] - min, a[2] = a[2] - min ... a[n] = a[n] - min`
- se determina `k = cmmdc(a[1], a[2] ... a[n]).`

b) Împărțirea diferențiată

Fie `ci` numarul maxim de colegi care pot primi bomboane din cutia `i`

Pentru a determina `ci` se determina cea mai mare valoare pentru care

`ci * (ci + 1) / 2 <= a[i]`

Metoda 1 – 30% din punctaj

Se calculează numărul maxim de elevi care pot primi bomboane dintr-o cutie. Se observă că pentru ca acest număr să fie maxim trebuie găsiți cei mai mici termeni, respectiv 1, 2, 3 … Se încearcă formarea unui vector cu numărul de bomboane distribuit. Se procedează asemănător sortării prin inserție. Se adaugă ultimul termen la sfârșitul vectorului. Dacă se obține o diferență deja creată, acesta se deplasează spre începutul vectorului până când diferențele sunt distincte.

Metoda 2

Cea mai simplă modalitate de a afișa o împărtire este

`1 ci 2 ci-1 3 ci-2 .... (de exemplu pt 10 bomboane 1, 4, 2, 3)`

La afisarea primei perechi trebuie avut in vedere faptul ca daca ci * (ci + 1) < a[i] atunci se afiseaza 1 si ci + diferenta (a[i] - ci * (ci+1) / 2

De exemplu pt a[i] = 10 se determina ci = 4.

Impartirea diferentiata 1 4 2 3

Pt a[i] = 12 impartirea diferentiata 1 6 2 3 (prima pereche 1 si 4 + 12 - 10)

Algoritmul

```
l <- 1 //pornim de la marginea stanga
r <- ci //pornim de la marginea dreapta
cat timp l <= r executa
    daca suntem la prima pereche afiseaza l si k + a[i] - ci *
(ci + 1) / 2
    altfel afiseaz l si r
    l <- l + 1//trecem la urmatoarele valori
    r <- r - 1
sf cat timp
```

Datorita faptului ca la fiecare pas diferenta intre 2 elemente consecutive scade, se respecta cerinta ca elementele consecutive sa aiba diferenta distincta.