Varianta 2

1) Fie $A = \{a_i = (l_i, c_i), i = \overline{1,n}\}$, $l_i \neq l_j\ \forall i \neq j$

unde $a_i \to$ cub
$l_i \to$ latura cubului
$c_i \to$ culoarea cubului

Definim mulțimea $S_1 = \{a_i \mid \forall j > i\ \ l_i > l_j$ și $c_{i-1} \neq c_i$ și $c_{i+1} \neq c_i\}$

Orice mulțime de forma S este o soluție a problemei.

Definim următorul algoritm pentru determinarea mulțimii cu înălțime maximă:

Definim ordinea „>" : $a_i > a_j \iff l_i > l_j$

Algoritm: ordonăm mulțimea A descrescător conform definiției „>".

```
solutie.push_back(primul element din A după sortare).
pentru i = 1, n-1
|  dacă elementul curent din A are culoare diferită de
|  | ultimul element din soluție
|  |     soluție.push_back(elementul curent din A)
|  |___
|___
scrie soluția
```

Deoarece elementele sunt sortate după l_i și adăugăm un cub nou a_i doar dacă are o culoare diferită de a ultimului element adăugat, este clar că secvența dată de algoritm este o soluție.

Demonstrăm că soluția este optimă.

P(1): Pentru un cub, soluția este a_1.

P(2): Pentru 2 cuburi $a_1 = (l_1, c_1)$
$a_2 = (l_2, c_2)$

I $l_1 > l_2$
dacă $c_1 = c_2$ soluția este a_1 (cea ce alege și algoritmul nostru)
dacă $c_1 \neq c_2$ soluția dată de algoritm este $a_1 a_2$ (care este și soluția optimă

II $l_2 > l_1$
analog cazul I

Presupunem P(n): soluția dată de algoritm coincide cu soluția optimă până la ~~un~~ pasul n.

~~Fie pasu~~ Demonstrăm P(n+1):
Pp. R. A. că soluția aleasă de noi diferă la pasul n+1 de soluția optimă.

Soluția noastră: $a_{k_1}, a_{k_2}, \dots, a_{k_n}, a_{k_{1,n+1}} = (l_{k_{1,n+1}}, c_{k_{1,n+1}})$ $\quad c_{k_{1,n+1}} \neq c_{k_n}$

soluția optimă: $a_{k_1}, a_{k_2}, \dots, a_{k_n}, a_{k_{2,n+1}} = (l_{k_{2,n+1}}, c_{k_{2,n+1}})$ $\quad c_{k_{2,n+1}} \neq c_{k_n}$

Dacă soluția optimă este într-adevăr optimă avem inegalitatea

$$\cancel{l_{k_1}} + \cancel{l_{k_2}} + \cancel{\dots} + l_{k_{1,n+1}} \leq \cancel{l_{k_1}} + \cancel{l_{k_2}} + \cancel{\dots} + l_{k_{2,n+1}}$$

$$\Rightarrow l_{k_{1,n+1}} \leq l_{k_{2,n+1}}$$

Dar $l_{k_{1,n+1}} \neq l_{k_{2,n+1}}$ din ipoteza problemei, deci:

$$l_{k_{1,n+1}} < l_{k_{2,n+1}} \quad (1)$$

Dar algoritmul alege $l_{k_{1,n+1}}$ maxim astfel încât $c_{k_{1,n+1}} \neq c_{k_n}$ din setul de cuburi rămase. Prin urmare, deoarece nu am ales $l_{k_{2,n+1}}$:

$$l_{k_{1,n+1}} > l_{k_{2,n+1}} \quad (2)$$

Din (1) și (2) $\Rightarrow$ ↯ $\Rightarrow$ Presupunerea este falsă $\Rightarrow a_{k_{1,n+1}} = a_{k_{2,n+1}}$
$\Rightarrow$ soluția dată de alg. este optimă.

Complexitate spațiu: $O(n)$

Complexitate timp: Sortarea $O(n \log n)$

selecția $O(n)$

b) Dacă aveam $l_i \geq l_j$ $\forall j > i$ soluția nu mai era valabilă.

Contraexemplu: $(4, 2)$ $(4, 3)$ $(3, 2)$ $(2, 3)$

algoritmul nostru alege:

$(4, 2)$ $(2, 3)$ $\Rightarrow l_{max} = 6$

dar soluția optimă este

$(4, 3), (3, 2)$ $\Rightarrow l_{max} = 7$