Μέγιστη Αύξουσα Υπακολουθία:

Είσοδος: Υπακολουθία φυσικών αριθμών $a_1, \dots, a_n$

Έξοδος: Το πλήθος στοιχείων στην Μ.Α.Υ.

Χρόνος $O(n^2)$

for $j=1$ to n: $Prev[j] = NIL$

for $j=1$ to n

$$L(j) = \max\{L(i) \mid i \text{ τ.ω. } a_i < a_j\} + 1$$

$$Prev[j] = \arg\max\{L(i) \mid i \text{ τ.ω. } a_i < a_j\}$$

return $\max\{L(j) \mid j=1,\dots,n\}$

$x = \arg\max\{L(j) \mid j=1,\dots,n\}$

return $(x, Prev)$

a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8
5	2	8	6	3	6	9	7

$L(1) = 1$

$L(2) = 1$

$L(3) = 2$

$L(4) = 2$

$L(5) = 2$

$L(6) = 3$

$Prev(6) = 5$

$L(7) = 4$

$Prev[7] = 6$

$L(8) = 4$

Δυναμικός Προγραμματισμός:

Βασική ιδέα:

1. Υποπροβλήματα
2. Διάταξη στα υποπροβλήματα
3. Σύνδεση στα υποπροβλήματα

Η λύση σε προηγούμενα υποπροβλήματα βοηθά την λύση σε επόμενα.

α_1 α_2 α_3 α_4 α_5 α_6 α_7 α_8

5 2 8 6 3 6 9 7

Ακολουθία: $\alpha_1, \dots, \alpha_n$

Υπακολουθία: $\alpha_{i_1}, \dots, \alpha_{i_k}$ και ισχύει ότι $1 \le i_1 < i_2 < \dots < i_k \le n$

Αύξουσα αν: $\alpha_{i_1} < \alpha_{i_2} < \dots < \alpha_{i_k}$

Υποπροβλήματα: Μ.Α.Υ. που τελειώνει με το στοιχείο α_j. (η υπακολουθία αυτή θα περιέχει στοιχεία μεταξύ των $\alpha_1, \dots, \alpha_j$).

Έστω $L(j) =$ # στοιχείων της Μ.Α.Υ. που τελειώνει με το α_j

$\max \emptyset = 0$

Αναδρομική σχέση:

$$\begin{cases} L(j) = \max\{L(i) \mid \alpha_i < \alpha_j\} + 1 \\ L(1) = 1 \end{cases}$$

miro

Διορθωτική Απόσταση

Είσοδος: Συμβολοσειρές $X[1...m]$ και $Y[1...n]$

Έξοδος: Η διορθωτική απόσταση των X και Y

for $i=0$ to m

  $E(i,0) = i$

for $j=1$ to n

  $E(0,j)=j$

for $i=1$ to m

  for $j=1$ to n

$$E(i,j) = \min\left\{ E(i-1,j)+1,\ E(i,j-1)+1,\ E(i-1,j-1) + \text{diff}(i,j) \right\}$$

return $E(m,n)$.

χρόνος $O(mn)$

X[1 ... m]

Y[1 .... n]

SNOWY ⇝ SUNNY

N ← U
O ← N
W ← N

S N O W Y
S U N N Y
✓ X X X ✓

3 διορθώσεις

S _ N O W Y
S U N N _ Y
✓ X ✓ X X ✓

_ S N O W _ Y
S U N _ _ N Y

S N O _ Y
S N O W Y

S N O W W Y
S N O W _ Y

Υποπροβλήματα:

E(i, j) = Διορθωτική απόσταση των λέξεων X[1...i]
και Y[1...j]

Εμείς θέλουμε την τιμή το E(m, n).

$E(i, j)$

1η περ.		2η περ.		3η περ.
$X[i]$		—	,	$X[i]$
—	ή	$Y[j]$	ή	$Y[j]$

1η περίπτωση:

$$E(i,j) = 1 + E(i-1, j)$$

2η περίπτωση:

$$E(i,j) = 1 + E(i, j-1)$$

3η πρόταση:

$$F(i,j) = \begin{cases} 1 + F(i-1, j-1), & X[i] \neq Y[j] \\ F(i-1, j-1), & X[i] = Y[j]. \end{cases}$$

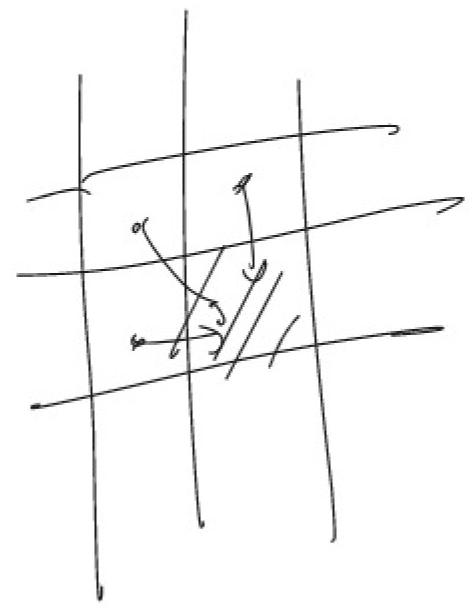
Έστω

$$diff(i,j) = \begin{cases} 1, & \text{αν } X[i] \neq Y[j] \\ 0, & \text{αλλιώς.} \end{cases}$$

$$
\begin{cases}
F(i,j) = \min \left\{ 1 + F(i-1,j),\ 1 + F(i,j-1),\ \mathrm{diff} + F(i-1,j-1) \right\} \\[2ex]
F(0,j) = j, \quad j = 1, \dots, n \\[1ex]
F(i,0) = i, \quad i = 1, \dots, m
\end{cases}
$$

i

j



Σακίδιο!

Είσοδος: n είδη με βάρος $w_1, \dots, w_n$ και αξία $v_1, \dots, v_n$ και μέγεθος σακιδίου W.

Έξοδος: Επιλογή αντικειμένων με την μεγαλύτερη αξία που να χωράν στο σακίδιο

με επανάληψη:

$\underline{\log W}$

χρόνος $O(W \cdot n)$

```
K(0) = 0
for w=1 to W
    K(w) = max{ K(w - wi) + vi | wi ≤ w }
return K(W)
```

Χωρίς επαναλήψεις!

```
for j = 1 to n
    K(0, j) = 0
for w = 1 to W
    K(w, 0) = 0
for j = 1 to n
    for w = 1 to W
        if w_j > w :  K(w, j) = K(w, j-1)
        else :  K(w, j) = max{ K(w, j-1), K(w-w_j, j-1) + v_j }
return K(W, n).
```

χρόνος $O(W \cdot n)$

$W = 10$

Είδος	Βάρος	Τιμή
1	6	30
2	3	14
3	4	16
4	2	9

• Με επανάληψη:

$K(w)$ = η μέγιστη αξία με χωρητικότητα w

$$\begin{cases} K(w) = \max\limits_{\substack{i:\\ w_i \le w}} \{ v_i + K(w - w_i) \} \\ K(0) = 0 \end{cases}$$

Χωρίς επανάληψη:

υποπροβλήματα!

$K(w, j) =$ μέγιστη αξία σακιδίου μεγέθους w με αντικείμενα $1, \dots, j$

θέλουμε $K(W, n)$.

$$K(w, j) = \max \left\{ K(w, j-1),\ v_j + K(w - w_j, j-1) \right\}$$