Αλγόριθμοι και Πολυπλοκότητα (2020-2021)
1η Σειρά Γραπτών Ασκήσεων
Ναυσικά Αμπατζή 03117198

Άσκηση 1α

Ισχύει ότι: $\boxed{n \cdot 2^{2^{100}} = \Theta(n)}$

$$\sum_{k=1}^{n} k \cdot 2^{-k} = \sum_{k=1}^{n} \frac{k}{2^k} = S_n$$

$$S_n = \frac{1}{2} + \frac{2}{4} + \frac{3}{8} + \dots + \frac{n-1}{2^{n-1}} + \frac{n}{2^n} \quad (1)$$

και $$2S_n = 1 + \frac{2}{2} + \frac{3}{4} + \frac{4}{8} + \dots + \frac{n}{2^{n-1}} \quad (2)$$

$$(2)-(1) \Longrightarrow 2S_n - S_n = \underbrace{\left(1 + \frac{1}{2} + \frac{1}{4} + \dots \frac{1}{2^{n-1}}\right)}_{\text{άπειρη γεωμετρική σειρά με κοινό παράγοντα το } 1/2} - \frac{n}{2^n}$$

άρα $$S_n = \left(2 - \frac{1}{2^{n-1}}\right) - \frac{n}{2^n} = \left(\frac{2^{n+1}}{2^n} - \frac{2}{2^n}\right) - \frac{n}{2^n} \Rightarrow$$

$$\Rightarrow \sum_{k=1}^{n} \frac{k}{2^k} = \frac{2^{n+1} - n - 2}{2^n} = 2^1 - n \cdot 2^{-n} - 2 \cdot 2^{-n}$$

Η πολυπλοκότητα του αθροίσματος αυτού καθορίζεται μόνο από τον σταθερό όρο 2 καθώς οι υπόλοιποι είναι αρνητικοί

άρα $\boxed{\sum_{k=1}^{n} k/2^k = \Theta(1)}$

Άρα η σειρά μέχρι τώρα είναι η εξής:

$\sum_{k=1}^{n} k/2^k$, $n \cdot 2^{2^{100}}$

• Για το $\sum_{k=1}^{n} k\cdot 2^k$: Γνωρίζουμε ότι $\sum_{k=0}^{n} x^k = \frac{x^{n+1}-1}{x-1}$ και

$$\frac{d}{dx}\sum_{k=0}^{n} x^k = \sum_{k=0}^{n} k x^{k-1}$$

Έτσι : $\sum_{k=1}^{n} kx^k = x\sum_{k=0}^{n} kx^{k-1} = \sum_{k=0}^{n} kx^k = \sum_{k=1}^{n} kx^k$

Συνεπώς:

$$\sum_{k=1}^{n} kx^k = x\frac{d}{dx}\left(\frac{x^{n+1}-1}{x-1}\right) = \frac{x+(n(x-1)-1)x^{1+n}}{(x-1)^2}$$

Ορίζοντας ως x το 2 έχουμε ότι:

$$\sum_{k=1}^{n} k\cdot 2^k = (n-1)2^{n+1}+2 = n\cdot 2^{n+1} - 2^{n+1} + 2$$

Άρα $\boxed{\sum_{k=1}^{n} k\cdot 2^k = \Theta(n\cdot 2^n)}$

• Για το $n\sum_{k=0}^{n}\binom{n}{k}$

Γνωρίζουμε ότι $(a+x)^n = \sum_{k=0}^{n}\binom{n}{k}a^{n-k}x^k$ (binomial theorem)

$$\underset{x=1}{\overset{a=1}{\Rightarrow}} (1+1)^n = \sum_{k=1}^{n}\binom{n}{k} \Rightarrow \sum_{k=1}^{n}\binom{n}{k} = 2^n \Rightarrow n\sum_{k=1}^{n}\binom{n}{k} = n\cdot 2^n$$

άρα $\boxed{n\sum_{k=1}^{n}\binom{n}{k} = \Theta(n\cdot 2^n)}$

· Για τα $2^{2^{2100}} n$ και $2^{(\log_2 \log_2 n)^4}$ μεγαλύτερης τάξης είναι το $2^{2^{210}} n$.

Αυτό ισχύει διότι εάν θέσουμε $2^{2^{210}} = c$, έχουμε ότι $c \cdot n = c \cdot 2^{\log_2 n}$ και προφανώς το $\log_2 n$ είναι μεγαλύτερης τάξης από το $(\log_2(\log_2 n))^4$.

άρα $\boxed{2^{(\log_2 \log_2 n)^4} < 2^{2^{2100}} \cdot n}$

· Για τα $\log(n!) / (\log\log n)^5$ και $n 2^{2^{2100}}$

$$\lim_{n \to \infty} \frac{2^{2^{2100}} \cdot n \cdot (\log\log n)^5}{\log(n!)}$$ ① θέτω $2^{2^{2100}} = c$

Ισχύει ότι $\log(n!) \leq n\log n \Rightarrow \frac{n}{\log(n!)} \geqslant \frac{1}{\log n}$

άρα $\frac{c \cdot n (\log\log n)^5}{\log(n!)} \geqslant \frac{c \cdot (\log\log n)^5}{\log n}$.

$$\cdot \lim_{n \to \infty} \frac{c \cdot (\log\log n)^5}{\log n} \overset{\frac{\infty}{\infty}}{=} \lim_{n \to \infty} c \frac{5(\log\log n)^4 \cdot \frac{1}{\cancel{n}\log n}}{\frac{1}{\cancel{n}}} =$$

$$= \lim_{n \to \infty} c \cdot \frac{5(\log\log n)^4}{\log n} = \cdots = \lim_{n \to \infty} \frac{c 120 \log\log n}{\log n} \overset{\frac{\infty}{\infty}}{=}$$

$$= \lim_{n \to \infty} c 120 \cdot \frac{\frac{1}{n\log n}}{\frac{1}{n}} = \lim_{n \to \infty} c \cdot 120 \cdot \frac{1}{\log n} = 0$$

άρα και το αρχικό όριο τείνει στο 0, άρα η $\log(n!) / (\log\log n)^5$ είναι μεγαλύτερη από την $2^{2^{2100}} n$ για $n \to \infty$

Άρα $n \cdot 2^{22100} = O\left(\frac{\log n!}{(\log\log n)^5}\right)$

Άρα η μέχρι τώρα σειρά είναι η εξής:

$$\sum_{k=1}^{n} k2^{-k}, \quad 2^{(\log_2 \log_2(n))^4}, \quad n \cdot 2^{22100}, \quad \frac{\log n!}{(\log\log n)^5}$$

• Για το $\log\left(\binom{2n}{n}\right)$, το $\binom{2n}{n} \xrightarrow{n \to \infty} \frac{4^n}{\sqrt{n\pi}}$

Άρα $\log\left(\binom{2n}{n}\right) \xrightarrow{\text{ασυμπτωτικά}} n\log 4 - \frac{1}{2}\log(n\pi) = \Theta(n)$

άρα $\log\left(\binom{2n}{n}\right) = \Theta(n)$

• Για τις $\sqrt{n}^{\log_2 \log_2(n!)}$ και $\log n!/(\log\log n)^5$ ισχύει ότι

$\log n!/(\log\log n)^5 = O\left(\sqrt{n}^{\log_2 \log_2(n!)}\right)$, καθώς η δεύτερη είναι εκθετική συνάρτηση ενώ η πρώτη ασυμπτωτική.

• Οι $n\sum_{k=0}^{n}\binom{n}{k}$ και $\sum_{k=1}^{n} k2^k$ έχουν ίδια πολυπλοκότητα $\Theta(n \cdot 2^n)$ η οποία είναι η μεγαλύτερη απ' όλες.

Τελική κατάταξη σε αύξουσα σειρά:

$$\sum_{k=1}^{n} k2^{-k},\quad 2^{(\log_2\log_2 n)^4},\quad n^{2^{2^{100}}},\quad \log\left(\binom{2n}{n}\right),\quad \log(n!)/(\log\log n)^5,$$

$$(\sqrt{n})^{\log_2\log_2(n!)},\quad \sum_{k=1}^{n} k2^{k},\quad n\sum_{k=0}^{n}\binom{n}{k}$$

με $n^{2^{2^{100}}} = \Theta(n) = \log\left(\binom{2n}{n}\right)$ και

$$n\sum_{k=0}^{n}\binom{n}{k} = \Theta(n2^n) = \sum_{k=1}^{n} k\cdot 2^k.$$

Άσκηση 1Β

① $T(n) = 6T(n/3) + n^2 \log n$

Εφαρμογή του Master Theorem:
$a = 6$, $b = 3$, $f(n) = n^2 \log n$
$n^{\log_b a} = n^{\log_3 6} = n^{1,63}$
Ισχύει ότι $f(n) = \Omega(n^{\log_3 6 + \epsilon})$ με $\epsilon \approx 0,2$

Επίσης για επαρκώς μεγάλα n έχουμε ότι:
$af(n/b) = 6f(n/3) = 6\left(\frac{n}{3}\right)^2 \cdot \log\frac{n}{3} = \frac{2n^2}{3} \cdot \log\frac{n}{3} \leq$
$\leq \frac{2}{3} n^2 \log n = \frac{2}{3} f(n) = cf(n)$, $c = 2/3$

Άρα η λύση της αναδρομικής σχέσης είναι:
$\boxed{T(n) = (n^2 \log n)}$

② $T(n) = 9T(n/3) + n^2 \log n$
$a = 9$, $b = 3$ $f(n) = n^2 \log n$
$n^{\log_b a} = n^{\log_3 9} = n^2 = \Theta(n^2)$
Δεν μπορούμε να εφαρμόσουμε το Master Theorem, διότι η $n^2 \log n$ είναι ασυμπτωτικά μικρότερη της $n^{\log_b a} = n^2$ αλλά όχι πολυωνυμικά.
Έχουμε ότι $T(n) = 9T(n/3) + n^2 \log n =$
$= 9\left(9 \cdot T\left(\frac{n}{9}\right) + \frac{n^2}{9} \cdot \log\frac{n}{9}\right) + n^2 \log n =$

$$= 9\left(9 \cdot T\left(\frac{n}{9}\right) + \frac{n^2}{9}\log n - \frac{n^2 \log 9}{9}\right) + n^2 \log n$$

$$= 81 \cdot T\left(\frac{n}{27}\right) + n^2 \cdot \log n - n^2 \log 9 + n^2 \log n =$$

$$= 81\, T(n/27) + 2n^2 \log n - n^2 \log 9$$

Γενικά έχουμε για κάθε κ: $T(n) = 3^{2k}\, T\left(\frac{n}{3^k}\right) + kn^2 \log n + \Theta(n^2)$

για $k = \log_3 n$ έχουμε:

$$T(n) = 3^{2\log_3 n} \cdot T\left(\frac{n}{3^{\log_3 n}}\right) + \log_3 n \cdot n^2 \log n + \Theta(n^2) =$$

$$= 3^{\log_3 n^2} \cdot T\left(\frac{n}{n}\right) + n^2 \log^2 n + \Theta(n^2) \overset{T(1)=1}{=}$$

$$= n^2 + n^2 \log^2 n + \Theta(n^2) = \Theta(n^2 \log^2 n)$$

$$\boxed{T(n) = \Theta(n^2 \log^2 n)}$$

③ $T(n) = 11\, T(n/3) + n^2 \log n$

Εφαρμογή του Master Theorem:

$a = 11$, $b = 3$, $f(n) = n^2 \log n$

- $n^{\log_b a} = n^{\log_3 11} \simeq n^{2,18}$
- $f(n) = n^2 \log n$ ασυμπτωτικά μικρότερη από το $n^{2,18}$ άρα $f(n) = O(n^{\log_b a})$

Από την 1η περίπτωση του Master Theorem έχουμε ότι $\boxed{T(n) = \Theta(n^{\log_3 11})}$

④ $T(n) = T(n/4) + T(n/2) + n$ ①

$T(n/4) < T(n/2)$

άρα $T(n) > 2T(n/4) + n$

Θεωρούμε $T_1 = 2T(n/4) + n$ Master Theorem Εφαρμογή

$a = 2$, $b = 4$ άρα $n^{\log_b a} = n^{\log_4 2} = n^{0,5}$

$f(n) = n = \Omega(n^{\log_4 2 + \epsilon})$, όπου $\epsilon \approx 0,2$

Επίσης για επαρκώς μεγάλα n:

$$a f(n/b) = 2 f(n/4) = 2 \cdot \frac{n}{4} \leq 2n = c f(n), \ c = 2$$

άρα από την $3^{η}$ περίπτωση έχουμε ότι:

$T_1(n) = \Theta(n)$

άρα $\underline{T(n) = \Omega(n)}$

Για $n = n/2$ ①:

$T(n/2) = T(n/8) + T(n/4) + n/2$ ②

① - ② $\Rightarrow T(n) - T(n/2) = T(n/2) - T(n/8) + n/2 \Rightarrow$

$\Rightarrow T(n) = 2T(n/2) - T(n/8) + n/2 < 2T(n/2) + n/2$

Έστω $T_2(n) = 2T(n/2) + n/2$, $a = 2$, $b = 2$, $f(n) = n/2$

Εφαρμογή του Master Theorem:

$n^{\log_b a} = n^{\log_2 2} = n^1 = n = \Theta(n)$

$f(n) = O(n^{\log_2 2 - \epsilon})$, όπου $\epsilon = 0,5$, άρα Εφαρμόζοντας την $1^{η}$ περίπτωση του Master Theorem προκύπτει ότι $T_2(n) = \Theta(n)$

άρα $\underline{T(n) = O(n)}$

Τελικά προκύπτει ότι $\boxed{T(n) = \Theta(n)}$

④ $T(n) = T(n/3) + \sqrt{n}$

Εφαρμογή του Master Theorem: $a=1$, $b=3$, $f(n)=\sqrt{n}$

$n^{\log_b a} = n^{\log_3 1} = n^0 = 1$

$f(n) = \Omega(n^{\log_b a+\varepsilon})$

Για επαρκώς μεγάλα n έχουμε ότι: $a f(n/b) =$

$= 1 \cdot \sqrt{\frac{n}{3}} = \frac{1}{\sqrt{3}}\sqrt{n} \le c \cdot f(n)$ όπου $c=1$

άρα από την περίπτωση 3 προκύπτει ότι: $T(n) = \Theta(\sqrt{n})$

⑤ $T(n) = 2T(n/4) + T(n/2) + n =$

$= T(n/4) + T(n/4) + T(n/2) + n$

Αυτή η περίπτωση λύνεται εάν αξιοποιήσουμε τις ειδικές μορφές του Master Theorem που υπάρχουν στις διαφάνειες.

Συγκεκριμένα εάν θεωρήσουμε ότι:

$\gamma_1 = 1/4$, $\gamma_2 = 1/4$, $\gamma_3 = 1/2$, τότε:

$\gamma_1 + \gamma_2 + \gamma_3 = \frac{1}{4} + \frac{1}{4} + \frac{1}{2} = \frac{1}{2} + \frac{1}{2} = \frac{2}{2} = 1$

Τότε $T(n) = \Theta(n \log n)$

⑥ $T(n) = T(n^{2/3}) + \Theta(\log n)$

Ισχύει ότι για κάποιο $c_1 > 0$ $\Theta(\log n) = c_1 \log n$
Έτσι για την αναδρομική σχέση έχουμε:

$$T(n) = T(n^{2/3}) + \Theta(\log n) = T(n^{4/9}) + c_1 \log n + \frac{2}{3} c_1 \log n = \ldots =$$

$$= T(n^{(2/3)^k}) + c \log n \sum_{i=0}^{k} (2/3)^i \quad \forall\, k > 0$$

Ασυμπτωτικά για $k \to \infty$:

$$T(n) = \lim_{k \to \infty} \left[T(n^{(2/3)^k}) + c \log n \sum_{i=0}^{k} (2/3)^i \right] =$$

$$= T(1) + c \log n \sum_{i=0}^{\infty} (2/3)^i = c \log n \cdot \frac{1}{1 - \frac{2}{3}} = 3c \log n = \Theta(\log n)$$

$$\boxed{T(n) = \Theta(\log n)}$$

Άσκηση 2

(α) Αλγόριθμος ταξινόμησης:

Έστω ο πίνακας A[n] που θέλουμε να ταξινομήσουμε.

① Ξεκινάμε από το στοιχείο A[n] και με διαδοχικές συγκρίσεις του A[n] με το A[n-1], του A[n-1] με το A[n-2] ... μέχρι να φτάσουμε στο πρώτο στοιχείο,

② Βρίσκουμε το μεγαλύτερο στοιχείο του πίνακα, έστω A[k].

③ Πραγματοποιούμε μία προθεματική περιστροφή για τα στοιχεία A[1]... A[k]. Προφανώς εάν το A[k] είναι το πρώτο στοιχείο του πίνακα η περιστροφή αυτή πραγματοποιείται γύρω από ένα στοιχείο

④ Πραγματοποιούμε μία ακόμα προθεματική περιστροφή για k=n και έτσι στη θέση A[n] βρίσκεται τώρα το μεγαλύτερο στοιχείο του πίνακα.

⑤ Αναζητούμε ξανά το μεγαλύτερο στοιχείο του πίνακα ανάμεσα στα στοιχεία A[1] ... A[n-1]. Επαναλαμβάνουμε τα βήματα ②, ③ και ④ με n=n-1.

⑥ Τα βήματα ⑤ → ③ → ④ επαναλαμβάνονται μειώνοντας το n κατά 1 κάθε φορά, μέχρι να προκύψει n=1. Τότε ο πίνακας A[n] είναι ταξινομημένος κατά αύξουσα σειρά.

Για n=1, δε θα πραγματοποιηθεί καμία περιστροφή.
Για n=2, θα πραγματοποιηθεί το πολύ μία περιστροφή.
Για n≥3 θα πραγματοποιούνται 2 περιστροφές και το μέγεθος του τμήματος αναζήτησης θα μειώνεται κατά 1.

Έστω $T(n)$ ο αριθμός των περιστροφών. και P_n ο ελάχιστος αριθμός των περιστροφών που μπορεί να πραγματοποιηθούν με αυτόν τον αλγόριθμό.

Ισχύει ότι $P_n \leq T(n)$

Η αναδρομική σχέση που ικανοποιεί το $T(n)$ δίνει:

$T(1) = 0$

$T(2) \leq 1$

$T(n) \leq 2 + T(n-1) \quad , n \geq 3$

Έτσι προκύπτει ότι $\boxed{T(n) \leq 2n-3 \quad , n \geq 2}$

(β) Για την προσημασμένη προθεματική περιστροφή ο αλγόριθμος είναι ο εξής:

Έστω A[n] ο πίνακας που θέλουμε να ταξινομήσουμε.

① Βρίσκουμε το μεγαλύτερο στοιχείο του κατ' απόλυτη τιμή. Έστω A[k].

② Πραγματοποιούμε προθεματική περιστροφή για τα στοιχεία A[1] ... A[k]. Τα πρόσημα των στοιχείων αυτών τώρα έχουν αντιστραφεί.

③
- Εάν το A[k] έχει αρνητικό πρόσημο πραγματοποιούμε προθεματική περιστροφή για $k=n$, ώστε να φτάσει στην μεγαλύτερη θέση του πίνακα.
- Εάν το A[k] έχει θετικό πρόσημο πραγματοποιούμε μία προθεματική περιστροφή για $k=1$ ώστε να αλλάξει μόνο το δικό του πρόσημο. Ομοίως πραγματοποιούμε προθεματική περιστροφή για $k=n$ ώστε να βρεθεί το A[k] στην μεγαλύτερη θέση και με θετικό πρόσημο.

④ Αναζητούμε ξανά το μεγαλύτερο στοιχείο του πίνακα ανάμεσα στα A[1] .... A[n-1] και επαναλαμβάνουμε τα βήματα ① ② ③ για $n=n-1$.

Αυτή η διαδικασία συνεχίζεται μέχρι το $n=n-1$ να ισούται με μηδέν. Τότε ο πίνακας έχει ταξινομηθεί κατά αύξουσα σειρά και με θετικά πρόσημα.

Συνολικά θα έχουμε το πολύ:

- n προθεματικές περιστροφές για να φτάσουν όλα τα στοιχεία στην πρώτη θέση του πίνακα
- n προθεματικές περιστροφές για να βρεθούν στο τέλος του πίνακα (σε αύξουσα σειρά)
- n προθεματικές περιστροφές για $k=1$ όταν τα στοιχεία βρίσκονται στην πρώτη θέση του πίνακα, εάν υποθέσουμε ότι όλα έχουν θετικό πρόσημο.

Άρα συνολικά $3n$ περιστροφές. και επειδή το μικρότερο (κατ' απόλυτη τιμή) στοιχείο του πίνακα θα χρειαστεί το πολύ μία προθεματική περιστροφή γύρω από τον εαυτό του εάν χρειάζεται να αλλάξουμε το πρόσημό του από αρνητικό σε θετικό, τελικά θα έχουμε $\boxed{\text{το πολύ } 3n-1 \text{ προθεματικές περιστροφές.}}$

(γ)

1. • Έστω ότι ο πίνακας A_t έχει τουλάχιστον ένα θετικό αριθμό και k ο μεγαλύτερος. Τότε ο αριθμός $k+1$ θα πρέπει να είναι αρνητικός $-(k+1)$, εκτός εάν $k=n$.

1η περίπτωση

$k=n$: [. . . . . n . . .] ΑΡΧΙΚΟΣ ΠΙΝΑΚΑΣ

↓ 1η περιστροφή

[−n . . . . . .]

↓ 2η περιστροφή

[. . . . n]

2η περίπτωση

Το k βρίσκεται πριν το $-(k+1)$ στον πίνακα:

[. . . k . . . −(k+1) . . .] ΑΡΧΙΚΟΣ ΠΙΝΑΚΑΣ

↓ 1η περιστροφή

[(k+1) . . . k . . .]

↓ 2η περιστροφή

[. . . . −(k+1), k . . .] $-(k+1), k$: συμβατό ζεύγος

3η περίπτωση

Το k βρίσκεται μετά το $-(k+1)$ στον πίνακα:

[. . . −(k+1) . . k . . .]

↓ 1η περιστροφή

[−k . . . (k+1) . . .]

↓ 2η περιστροφή

[. . . k, (k+1) . . .] $k, k+1$: συμβατό ζεύγος

• Έστω ότι ο πίνακας A_t έχει μόνο αρνητικούς αριθμούς και είναι διαφορετικός από τον $[-1, -2 \dots -n]$. Τότε θα υπάρχουν αριθμοί $k, -(k+1)$, όπου το $-k$ θα βρίσκεται σε θέση μετά το $-(k+1)$. Έτσι θα έχουμε:

$[\dots \ -(k+1) \ \dots \ -k \ \dots]$

$\downarrow$ 1η περιστροφή για τα στοιχεία $[\dots \ -(k+1)]$

$[(k+1) \ \dots \ -k \ \dots]$

$\downarrow$ 2η περιστροφή για τα στοιχεία που βρίσκονται πριν το $-k$

$[\dots \ -(k+1) \ -k \ \dots]$ προκύπτει συμβατό ζεύγος $-(k+1), -k$

Άρα αποδείξαμε ότι για κάθε ενδιάμεσο πίνακα A_t (διαφορετικό από τον $[-1 \ -2, \dots, -n]$) μπορούμε να δημιουργήσουμε ένα νέο συμβατό ζεύγος με το πολύ 2 περιστροφές προσημασμένες προθεματικές.

2. Ζητούμενος Αλγόριθμος:

Η επίλυση του για τις 4 από τις 5 πιθανές περιπτώσεις προκύπτει από το προηγούμενο ερώτημα, καθώς για κάθε δυνατό ζεύγος ενός πίνακα $A[n]$ χρειαζόμαστε το πολύ 2 προσυμασμένες προθεματικές περιστροφές. Άρα για να βρεθούν όλα τα δυνατά ζεύγη θα χρειαστούμε το πολύ $2n$ περιστροφές καθώς σε κάθε εύρεση θα περιορίζουμε την αναζήτηση σε $n-1$.. στοιχεία

- Για την $5^{η}$ περίπτωση όπου προκύπτει πίνακας της μορφής $[-1, -2, -3 \ldots -n]$

Αλγόριθμος:

Επανέλαβε τα 2 εξής βήματα n φορές:

1. Προθεματική περιστροφή των n αριθμών
2. Προθεματική περιστροφή των $n-1$ αριθμών.

Απόδειξη με επαγωγή: ότι χρειάζονται $2n$ περιστροφές

Αφού τρέξουμε τον αλγόριθμο $2k$ φορές $(k \leq n)$ ο πίνακας θα έχει την εξής μορφή:

$$[-(k+1), -(k+2) \ldots -(n-1) -n, 1 \ldots (k-1), k] \quad (1)$$

- Για $k=0$ ο ισχυρισμός προφανώς ισχύει.
- Θεωρώ ότι ισχύει και για k και εκτελώ τον παραπάνω αλγόριθμο στον πίνακα (1):

$$\xrightarrow{1^{η}\ \text{περιστροφή}} [-k, -(k-1) \ldots -1, n, (n-1), \ldots (k+2), k+1]$$

$$\xrightarrow{2^{η}\ \text{περιστροφή}} [-(k+2), \ldots, -(n-1) -n, 1, \ldots (k-1), k, k+1]$$

Άρα προκύπτει ότι ο ισχυρισμός ισχύει για $k+1$, άρα θα ισχύει για κάθε k με: $0 \leq k \leq n$.
Έτσι για $k=n$ ο πίνακας θα ταξινομηθεί.

Εφόσον έχουμε εξετάσει όλες τις περιπτώσεις προκύπτει ο εξής Τελικός αλγόριθμος:

Εφαρμόζουμε τα βήματα των περιπτώσεων 1-4 k φορές είτε μέχρι ο $A[n]$ να ταξινομηθεί σωστά είτε μέχρι να προκύψει η μορφή της $5^{ης}$ περίπτωσης.

Εάν έχει ταξινομηθεί σωστά το πρόβλημα τελείωσε. Αλλιώς εφαρμόζουμε τον αλγόριθμο της $5^{ης}$ περίπτωσης.

Συνολικά θα έχουμε $2k+2(n-k)=2n$ περιστροφές.

Άσκηση 3

①

Θεωρούμε μία τράπουλα n καρτών. Επιλέγοντας την πρώτη κάρτα προφανώς δημιουργείται η πρώτη στοίβα. Έστω ότι η επιλογή κάποιας επόμενης κάρτας οδηγεί σε δημιουργία δεύτερης στοίβας. Για να επιλέξουμε που θα τοποθετηθεί μία κάρτα (c) η οποία έχει τιμή μικρότερη από τις κάρτες στην κορυφή των δύο ή και περισσότερων στοιβών θα ψάξουμε τη στοίβα με τη μικρότερη τιμή, ώστε πιθανές κάρτες με μεγαλύτερη τιμή από αυτή της c να έχουν περισσότερες πιθανότητες να τοποθετηθούν σε ήδη υπάρχουσα στοίβα.

Στη χειρότερη περίπτωση η διάταξη της τράπουλας θα είναι: 1, 2, 3, 4 ... n, άρα θα δημιουργηθούν n στοίβες.

Απόδειξη ορθότητας του αλγορίθμου

(Οι κάρτες συμβολίζονται με c)

Έστω S η παραπάνω υλοποίηση και P μία γενική επίλυση. Ο στόχος είναι να συσχετίσουμε τις S και P επαγωγικά.

Έστω $x^i = (x_1^i, x_2^i \ldots x_{k_i}^i)$ οι τιμές των στοιχείων στις κορυφές των στοιβών που προκύπτουν μετά την S υλοποίηση οι οποίες θα βρίσκονται σε αύξουσα σειρά ①

(Απόδειξη του Θεωρήματος ① στη συνέχεια)

Ομοίως ορίζω $y^i = (y_1^i, y_2^i \ldots x_{k_i}^i)$ για την P.

Θα αποδειχθεί ότι (με επαγωγή) $\forall i = 1, \dots, n$ έχουμε ότι
a) $h_i \leq k_i$ και
b) $x_j^i \geq y_j^i \quad \forall j = 1, \dots, h_i$

Επαγωγικά έχουμε ότι για $i=1$, $h_1 = k_1 = 1$ και $x^1 = y^1 = c_1$

Για $i>1$ έχουμε τις εξής δύο περιπτώσεις:

(1) Σύμφωνα με τη λύση S το c_i δημιουργεί νέα στοίβα. Υποθέτουμε ότι $c_i > x_j^{i-1} \geq y_j^{i-1} \ \forall j = 1 \dots h_{i-1}$
Τότε έχουμε ότι $k_i \geq h_{i-1} + 1 = h_i$
Για $j = 1 \dots h_{i-1}$ έχουμε ότι $c_i > x_j = x_j^{i-1} \geq y_j^{i-1} = y_j^i$
από επαγωγική υπόθεση. και $y_{h_i}^i \leq c_i = x_{h_i}^i$

(2) Σύμφωνα με τη λύση S το c_i τοποθετείται σε ήδη υπάρχουσα στοίβα.
Έστω m ο δείκτης ώστε $x_m^i = c_i$. Τότε
$k_i \geq k_{i-1} \geq h_{i-1} = h_i$
Επίσης για κάθε $j = 1 \dots m-1$ έχουμε ότι
$c_i > x_j^{i-1} \geq y^{i-1}$ και έτσι $x_j^i \geq y_j^i$
Συνεπώς:
$x_m^i = c_i = \min\{x_m^{i-1}, c_j\} \geq \min\{y_m^{i-1}, c_j\} = y_m^i$

Για $j = m+1, \dots, h_i$ έχουμε ότι:

$$x_j^i = x_j^{i-1} \geq \max\{c_i, x_j^{i-1}\} \geq \max\{c_i, t_j^{i-1}\} \geq t_j^i$$

$(h_n < k_n)$
Εάν θέσουμε n στη θέση του i έχουμε ότι η λύση S ελαχιστοποιεί τον αριθμό των στοιβών.

Απόδειξη του Θεωρήματος ①

· Προφανώς για $i=1$ το ① ισχύει.
Υποθέτουμε ότι ισχύει για n κάρτες

Έστω ότι έχουμε την στοίβα k όπου τοποθετούμε την $(n+1)$οστή κάρτα πάνω από μία ήδη υπάρχουσα έστω m κάρτα. Ονομάζω l την $(n+1)$οστή κάρτα $q < m < r$ από την αρχική υπόθεση.

	q	l m	r
αρχή στοιβών	$k-1$	k	$k+1$

Υποθέσαμε ότι $r > m$ άρα θα πρέπει να ισχύει και ότι $r > l$.
Επίσης αρχικά ισχύει ότι $q < m$, αλλά η κάρτα l μπήκε μετά από το m. Έτσι προκύπτει ότι $a<d$.

Άρα ισχύει ότι $q < l < r$ και από επαγωγή για $(n+1)$ κάρτες οι κορυφές των στοιβών του αλγορίθμου θα βρίσκονται κατ' αύξουσα σειρά ταξινομημένες από αριστερά προς τα δεξιά.

Εύρεση υπολογιστικής πολυπλοκότητας του αλγορίθμου

Η χειρότερη περίπτωση (κάρτες σε αύξουσα σειρά αρχικά ταξινομημένες) απαιτεί οι κάρτες να συγκριθούν όλες μεταξύ τους. Δηλαδή για n κάρτες γίνονται:

$$0+1+2+\dots+n-1 = (1+2+\dots+n-1+n)-n = \frac{n(n-1)}{2} = \frac{n^2}{2} - \frac{n}{2} \text{ συγκρίσεις}$$

άρα η πολυπλοκότητα είναι $\Theta(n^2)$

Βελτιστοποίηση

Αποδείχθηκε ότι οι κορυφές των στοιβών του αλγορίθμου βρίσκονται σε αύξουσα σειρά. Επομένως η επιλογή της στοίβας γίνεται με binary search ώστε να βρούμε ανάμεσα σε ποιες κορυφές στοιβών βρίσκεται η τιμή της κάρτας και από αυτές την τοποθετούμε στη μεγαλύτερη κορυφή. Εάν βρεθούμε στην αριστερότερη στοίβα και η τιμή της κάρτας είναι μικρότερη από την τιμή της κορυφής τότε η κάρτα μπαίνει ως νέα κορυφή. Εάν βρεθούμε στην δεξιότερη στοίβα και η τιμή της κάρτας είναι μεγαλύτερη από την κορυφή τότε δημιουργούμε νέα στοίβα, αλλιώς ορίζουμε την κάρτα ως νέα κορυφή της δεξιότερης στοίβας.

Έστω m το πλήθος των στοιβών στη χειρότερη περίπτωση. Τότε στη χειρότερη περίπτωση θα έχουμε n κάρτες με $\log m$ συγκρίσεις για κάθε κάρτα. Όμως θα ισχύει ότι $n \geq m \Rightarrow$
$\Rightarrow \log n \geq \log m \Rightarrow$ Πολυπλοκότητα $\Rightarrow n \cdot \log m \leq n \log n$
$\Rightarrow \boxed{O(n \log n)}$

(2) Θα χρησιμοποιηθεί η βελτιστοποιημένη λύση του αλγορίθμου από το προηγούμενο ερώτημα. Έχουμε k στοίβες που θέλουμε να συγχωνεύσουμε ανά δύο. Χρησιμοποιούμε τον αλγόριθμο merge

Κόστος: $\sum_{i=1}^{k/2} \{$πλήθους καρτών i-οστου ζεύγους $-1\} = n - \frac{k}{2}$

για $k \leq n$

άρα το κόστος για να χωριστούν οι στοίβες σε δύο μέρη είναι $\Theta(n)$ και εφόσον κάθε φορά το πλήθος θα είναι το μισό, για να φτάσουμε σε μια στοίβα θα χρειαστούμε $\log_2(k)$ διαιρέσεις

Εφόσον $\log_2 k \leq \log_2 n$ το κόστος του merge προκύπτει $O(n\log_2 n)$

Τελικά όλος ο αλγόριθμος έχει χρόνο εκτέλεσης στη χειρότερη περίπτωση

$$\underbrace{O(n\log n)}_{\text{από το προηγούμενο ερώτημα}} + O(n\log n) = O(n\log n)$$

③ Είσοδος : 3, 2, 4, 7, 8, 1, 5, 6

Αλγόριθμος 1ου ερωτήματος :

1. 3 2 4 7 8 1 5 6 | 3 | 1η στοίβα

2. 3 2 4 7 8 1 5 6 , $2 < 3$

2
3

3. 3 2 4 7 8 1 5 6 , $4 > 2$ → νέα στοίβα :

2	
3	4

4. 3 2 4 7 8 1 5 6 , $7 > 2$ και $7 > 4$ → νέα στοίβα

2		
3	4	7

5. 3 2 4 7 8 1 5 6 , $8 > 2$ και $8 > 4$, $8 > 7$ → νέα στοίβα

2			
3	4	7	8

6. 3 2 4 7 8 1 5 6

1			
2			
3	4	7	8

7. 3 2 4 7 8 1 5 6

1			
2		5	
3	4	7	8

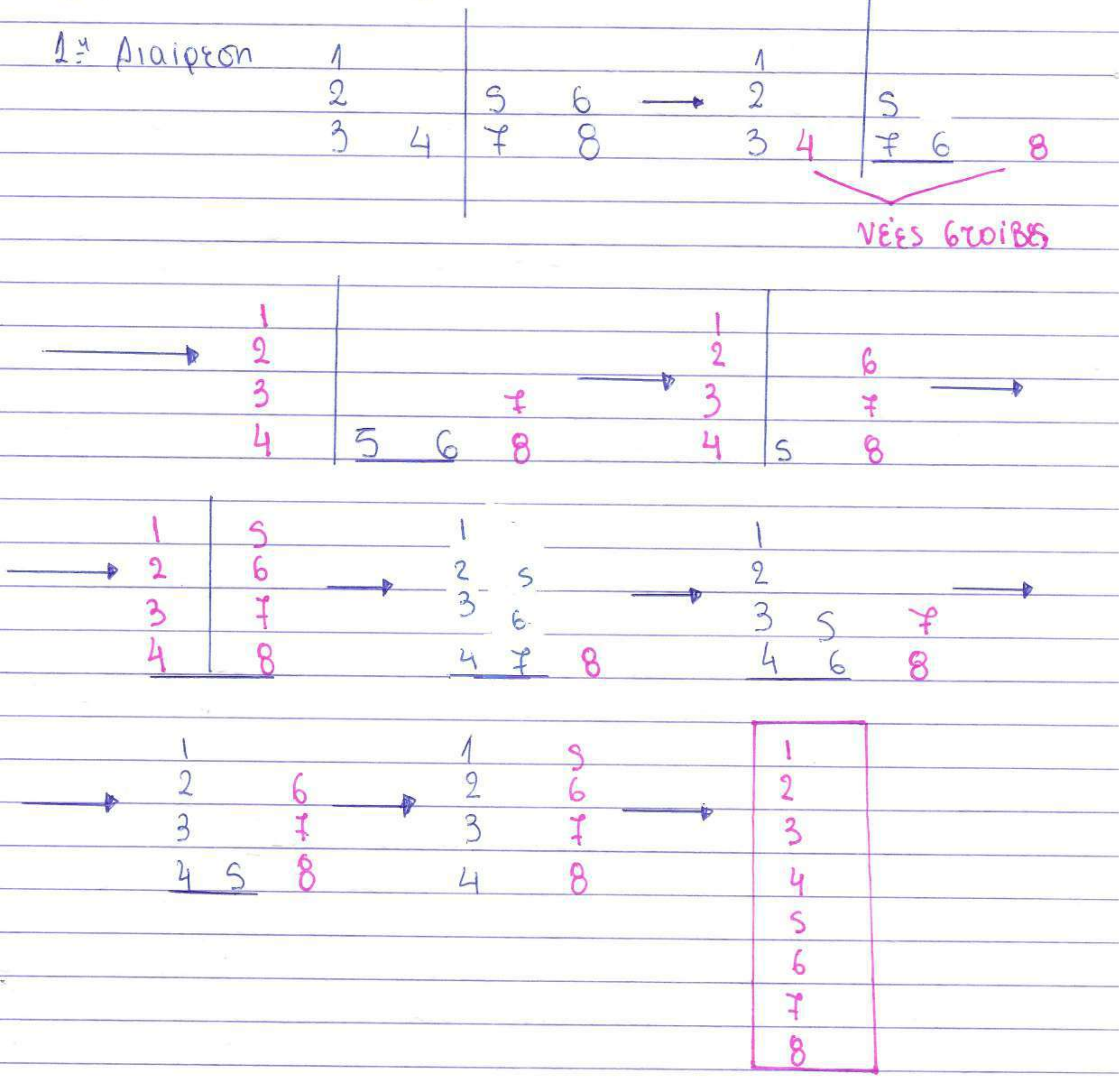
8. 32478156

1			
2		5	6
3	4	7	8

Προκύπτουν 4 στοίβες

Αλγόριθμος 2ου ερωτήματος

1η Διαίρεση



Το μήκος της μέγιστης αύξουσας υπακολουθίας είναι:
4 (3 4 7 8)

(4) Αρκεί να δειχθεί ότι ο αλγόριθμος του 1ου ερωτήματος δημιουργεί στοίβες πλήθους μεγαλύτερου ή ίσου του μήκους της μέγιστης αύξουσας υπακολουθίας, καθώς ο αλγόριθμος αυτός αποδείξαμε ότι δημιουργεί το ελάχιστο πλήθος στοιβών.

Θα χρησιμοποιηθεί η επαγωγή για την απόδειξη του ισχυρισμού αυτού.

- Η υπόθεση ισχύει αν σε κάθε στοίβα έχουμε ή κανένα ή 1 στοιχεία από κάθε αύξουσα υπακολουθία και αν οι μεγαλύτεροι αριθμοί κάθε υπακολουθίας είναι τοποθετημένοι στις τέρμα δεξιά στοίβες.

- Για είσοδο μία κάρτα ($i=1$) οι παραπάνω προϋποθέσεις ισχύουν.
- Υποθέτουμε ότι ισχύουν για i κάρτες και θα αποδείξουμε ότι ισχύουν για $(i+1)$ κάρτες.

Έστω ότι η $(i+1)$ στη κάρτα μπορεί να τοποθετηθεί στο τέλος κάποιας αύξουσας υπακολουθίας $[x_1, x_2 \dots x_k]$
Ισχύει ότι $(i+1)$ κάρτα $> x_k$ και είναι μεγαλύτερη από τις κορυφές όλων των στοιβών πριν από αυτή (απόδειξη στο 1ο ερώτημα), άρα μεγαλύτερα και από τις κορυφές πριν το x_k.
Άρα η $(i+1)$ κάρτα πρέπει να τοποθετηθεί σε μία στοίβα σίγουρα δεξιότερα από το x_k και η στοίβα αυτή μέχρι την κάρτα i δεν περιέχει κανένα στοιχείο της $[x_1, x_2 \dots x_k]$

Συνεπώς εφόσον αυτό θα συμβαίνει για κάθε αύξουσα υπακολουθία, θα ισχύει και για τη μέγιστη.

⑤ Παρατήρηση

Το κορυφαίο στοιχείο της k-οστής στοίβας είναι το τελευταίο στοιχείο κάποιας αύξουσας υπακολουθίας με k στοιχεία.

Σύμφωνα με αυτή την παρατήρηση για να αποδείξουμε ότι ο αλγόριθμος του (1) λύνει το πρόβλημα της μέγιστης αύξουσας υπακολουθίας, αρκεί να αποδείξουμε ότι:

Ισχυρισμός

Στο τέλος του παιχνιδιού εάν επιλέξουμε οποιαδήποτε κάρτα από την k-οστή στοίβα μπορούμε να βρούμε μία αύξουσα υπακολουθία στον αρχικό πίνακα με τους αριθμούς, της οποίας το μήκος είναι k.

• Έχουμε αποδείξει ότι οι κορυφαίες κάρτες σε κάθε (X) στοίβα βρίσκονται από αριστερά προς τα δεξιά σε αύξουσα σειρά.

• Θα αποδείξουμε τώρα ότι οποιαδήποτε στιγμή κάθε κάρτα κάποιας (k) στοίβας αποτελεί στοιχείο κάποιας αύξουσας υπακολουθίας της οποίας το μέγεθος δίνεται από το k.

Εφαρμόζουμε την επαγωγή:

• Η πρώτη κάρτα της πρώτης στοίβας (k=1) αποτελεί αύξουσα υπακολουθία με μήκος 1.

• Θεωρούμε ότι ο ισχυρισμός ισχύει για m κάρτες. Θα αποδειχθεί ότι ισχύει και για (m+1) κάρτες.

Θεωρούμε ότι βρισκόμαστε στην k στοίβα. Για $k=1$ προφανώς ο ισχυρισμός ισχύει και για την $(m+1)$ κάρτα. Για $k>1$ κοιτάμε το κορυφαίο στοιχείο της $(k-1)$ στοίβας. Γνωρίζουμε από το θεώρημα ⊛ ότι το στοιχείο αυτό θα είναι μικρότερο από το στοιχείο $(m+1)$ που μόλις τοποθετήσαμε.
Επίσης ισχύει ότι η κάρτα στην κορυφή της $(k-1)$ στοίβας είναι η m (από την αρχική υπόθεση), δηλαδή είναι μέρος μίας αύξουσας υπακολουθίας με μήκος $(k-1)$ έτσι ώστε η υπακολουθία με την προσθήκη της $(m+1)$ κάρτας σχηματίζει μία αύξουσα υπακολουθία με μήκος k.
Άρα αποδείξαμε το ζητούμενο.

• Για το μήκος της μέγιστης φθίνουσας υπακολουθίας:

$\forall y \in [1, 2, \dots, n] \longrightarrow (n+1) - y$ και εφαρμόζουμε τον αλγόριθμο του (1)
Οι k' στοίβες που θα δημιουργηθούν αποτελούν το μήκος της max φθίνουσας υπακολουθίας της αρχικής τράπουλας

(6) Έστω το ανακάτεμα της τράπουλας με $(nm+1)$ φύλλα
Έστω ότι ο αλγόριθμος του (1) δημιούργησε k στοίβες και η i-οστή έχει x_i στοιχεία
Υποθέτω ότι $x_i < m+1$ και $k < n+1$, δηλαδή
$x_i \le m$ και $k \le n$
άρα $x \cdot y_{max} \le m \cdot n$ και $n \cdot m + 1 = \sum_{i=1}^{k} x_i \le \sum_{i=1}^{k} y_{max} \le x \cdot y_{max} \le$

$\le n \cdot m$ άτοπο αφού δε γίνεται για $n, m \ge 0$: $nm+1 \le n \cdot m$

άρα θα δημιουργηθούν είτε $(n+1)$ στοίβες είτε μια στοίβα μεγέθους $m+1$.
Η παρατήρηση αυτή συνδέεται την Αρχή του Περιστερώνα των Διακριτών Μαθηματικών.

Παράδειγμα εισόδου:

21, 22, 23, 24, 25, 16, 17, 18, 19, 20, 11, 12, 13, 14, 15, →
6, 7, 8, 9, 10, 1, 2, 3, 4, 5

Εφαρμογή του Αλγορίθμου (1):

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

Άσκηση 4

Μέρος 1

Όσον αφορά τη χρονική στιγμή της σύγκρουσης:
Το νωρίτερο που μπορεί να συμβεί είναι εάν και τα δύο σωματίδια κινούνται με τη μέγιστη ταχύτητα V_{max} και το αργότερο εάν κινούνται με την ελάχιστη V_{min}. Και στις δύο αυτές περιπτώσεις εφόσον η ταχύτητά τους είναι κοινή θα συγκρουστούν στο μέσο της μεταξύ τους απόστασης, δηλαδή στο $\frac{L}{2}$.
Έστω t η στιγμή της σύγκρουσης.
άρα $t_{min} = \frac{L}{2 \cdot V_{max}}$ και $t_{max} = \frac{L}{2 \cdot V_{min}}$

Έτσι το σύνολο των πιθανών χρονικών στιγμών των συγκρούσεων είναι της μορφής $\Theta\left(\frac{L}{V_{min}} - \frac{L}{V_{max}}\right)$
* απόδειξη στην επόμενη σελίδα

Για να βρούμε την χρονική στιγμή της πρώτης σύγκρουσης και την ταυτότητα των σωματιδίων που συμμετείχαν θα χρησιμοποιήσουμε τον αλγόριθμο της δυαδικής αναζήτησης.
Χωρίζουμε το παραπάνω διάστημα σε δύο ίσα διαστήματα. Ελέγχουμε εάν οι θέσεις όλων των σωματιδίων τύπου a είναι μικρότερες από των σωματιδίων τύπου b τη χρονική στιγμή t' που βρίσκεται στη μέση του διαστήματος αυτού. Εάν οι θέσεις των a είναι μικρότερες τότε δεν έχει συμβεί καμία σύγκρουση μέχρι την t'. Τότε συνεχίζουμε τον διαχωρισμό στο δεξί διάστημα (χρονικές στιγμές μετά την t'). Επαναλαμβάνουμε αυτή τη διαδικασία μέχρι να βρεθεί κάποιο σωματίδιο τύπου a δεξιά από κάποιο τύπου b.

Τότε το σωματίδιο αυτό είναι εξαϋλωμένο, ωστόσο ο υπερυπολογιστής "συμπεριφέρεται" σαν αυτό να ήταν το μοναδικό σωματίδιο στο πεδίο της μάχης. Τότε κοιτάμε το αριστερό κομμάτι του τμήματος που έχουμε διαχωρίσει. Συνεχίζουμε την παραπάνω διαδικασία μέχρι να βρούμε την ίδια θέση σε μία δεδομένη χρονική στιγμή για δύο διαφορετικά $i \begin{cases} a \\ b \end{cases}$

Σε κάθε επανάληψη χρειαζόμαστε χρόνο $O(n)$ για να βρούμε τις θέσεις όλων των στοιχείων (η κλήσεις του υπερυπολογιστή).

Δηλαδή θα έχουμε $O(n)$ κλήσεις για $\Theta\left(\frac{L}{V_{min}} - \frac{L}{V_{max}}\right)$ χρονικές στιγμές.

Άρα στη χειρότερη περίπτωση θα χρειαστούμε χρόνο:

$$\boxed{O\left(n \log\left(\frac{L}{V_{min}} - \frac{L}{V_{max}}\right)\right)}$$

Απόδειξη του $\Theta(L/V_{min} - L/V_{max})$

Γνωρίζουμε ότι ένα χρονικό διάστημα είναι ένα άπειρο σύνολο. Έτσι για να βρούμε μία χρονική στιγμή αυτή θα πρέπει να είναι μέρος ενός πεπερασμένου συνόλου. Θεωρώ ότι η χρονική αυτή στιγμή είναι πολλαπλάσιο του $x$sec. και τότε οι χρονικές στιγμές διακριτοποιούνται. Έτσι με αυτή την ακρίβεια το σύνολο θα έχει $\frac{1}{x}\left(\frac{L}{2V_{min}} - \frac{L}{2V_{max}}\right) + 1 = \Theta\left(\frac{L}{V_{min}} - \frac{L}{V_{max}}\right)$.

Μέρος 2

- Επιλέγουμε τυχαία ένα σωματίδιο τύπου b και αναζητούμε τις χρονικές στιγμές που συγκρούεται με τα σωματίδια τύπου a, εάν δεν εξαϋλωνόταν και κρατάμε το ζευγάρι με τον μικρότερο χρόνο σύγκρουσης, έστω t'. Φυσικά δεν υπάρχουν σωματίδια τύπου a που να βρίσκονται δεξιότερα του σωματιδίου τύπου b που εξετάζουμε εκείνη την χρονική στιγμή, αλλιώς το b θα είχε ήδη συγκρουστεί με κάποιο από αυτά για $t < t'$.
  Έτσι απορρίπτονται όλα τα σωματίδια τύπου b που βρίσκονται δεξιότερα του επιλεγμένου σωματιδίου τύπου b από πιθανά για συμμετοχή στην πρώτη σύγκρουση.

- Επιλέγουμε ξανά ένα τυχαίο σωματίδιο τύπου b και επαναλαμβάνουμε την ίδια διαδικασία μέχρι να φτάσουμε και στο μοναδικό υποψήφιο σωματίδιο τύπου b. Συγκρίνοντας για τελευταία φορά τους χρόνους σύγκρουσης με σωματίδια τύπου a βρίσκουμε το ζευγάρι (a, b) και την χρονική στιγμή της πρώτης σύγκρουσης

Σύμφωνα με την πιθανοτική Quickselect σε κάθε βήμα τουλάχιστον $n/4$ σωματίδια τύπου b απαλείφονται με αρκετά μεγάλη πιθανότητα. Συνεπώς οι επαναλήψεις θα είναι της τάξης του $O(\log n)$

;Βελτιστοποίηση του παραπάνω αλγορίθμου.

Θα βελτιώσουμε την γραμμική αναζήτηση στα σωματίδια τύπου a σε κάθε επανάληψη που δεν μειώνονται ποτέ. Αντί να εξετάζεται κάθε σωματίδιο τύπου b με όλα τα a κάθε φορά, τη μία φορά θα εξετάζεται το τύπου b με όλα τα a και την επόμενη ένα τυχαίο τύπου a με όλα τα τύπου b. Τότε σε κάθε επανάληψη οι υπολογισμοί θα μειώνονται.

- Χρησιμοποιώντας την πιθανοτική Quickselect τώρα η γραμμική αναζήτηση έχει κ πράξεις, οι οποίες πολλαπλασιάζονται με τα 3/4 σε κάθε δύο επαναλήψεις
- Θα έχουμε $2 \cdot \log_{4/3} n$ κλήσεις και χρόνο $\Theta(1)$ για κάθε εκτίμηση.

Το τελικό κόστος δίνεται από την αναδρομική σχέση:

$$T(2n) = T\left(\frac{3n}{4} + n\right) + \Theta(n) \Rightarrow T(n) = T\left(\frac{7n}{8}\right) + \Theta(n/2)$$

από Master Theorem $\Rightarrow \boxed{T(n) = \Theta(n)}$

Άσκηση 5

(α) Αρχικά θεωρούμε έναν πίνακα A με m θέσεις αρχικοποιημένες στο '0' και μία συνάρτηση hash, η οποία αντιστοιχεί δεδομένα στις m θέσεις του πίνακα.

- Για τη λειτουργία "πρόσθεσε το x στο S" τροφοδοτείται το x στη συνάρτηση hash και απαιτά έτσι μία θέση στον πίνακα A. Το bit της θέσης που αντιστοιχίζεται γίνεται '1'.
- Για τη λειτουργία "έλεγξε εάν $x \in S$" τροφοδοτείται το x στη συνάρτηση hash και ελέγχεται το bit του πίνακα A, στο οποίο το αντιστοιχεί η συνάρτηση hash. Εάν αυτό είναι '0' το στοιχείο σίγουρα δεν βρίσκεται στο σύνολο S. Εάν το bit είναι '1' τότε είτε το $x \in S$, είτε έγινε τυχαία '1' από προηγούμενη εισαγωγή στοιχείου. (false positive περίπτωση)

Για να εμφανιστεί η false positive απάντηση με μικρή πιθανότητα θα πρέπει το πίνακας A να έχει πολλές θέσεις και να περιέχει μακρόχρονα μηδενικά, ώστε το περιεχόμενο των πληροφοριών του πίνακα να είναι αρκετά μικρό σε σχέση με το μέγεθός του.

Η διαγραφή στοιχείου από τον A δεν επιτρέπεται καθώς υπάρχει ενδεχόμενο το bit του στοιχείου που από '1' τεθεί σε '0' να αντιστοιχεί και σε κάποιο άλλο στοιχείο ταυτόχρονα. Πρόκειται δηλαδή για false negative περίπτωση, η οποία δεν επιτρέπεται.

Γνωρίζουμε ότι η συνάρτηση hash αντιστοιχεί ένα στοιχείο x σε μία από τις m θέσεις του πίνακα A με πιθανότητα $\frac{1}{m}$.

Επομένως η πιθανότητα κάποιο bit να μην τεθεί ίσο με '1' σε κάποια εισαγωγή είναι ίση με $\left(1-\frac{1}{m}\right)$

Γνωρίζουμε ότι για μεγάλο m ισχύει ότι:

$$\lim_{m\to\infty}\left(1-\frac{1}{m}\right)^m = \frac{1}{e} = e^{-1} \quad ①$$

Εάν εισάγουμε n θετικούς φυσικούς αριθμούς τότε η πιθανότητα κάποιο bit να παραμείνει ίσο με 0 είναι:

$$P_1 = \left(1-\frac{1}{m}\right)^n = \left[\left(1-\frac{1}{m}\right)^m\right]^{\frac{n}{m}} \overset{①}{\simeq} e^{-n/m}$$

Άρα η πιθανότητα κάποιο bit να είναι ίσο με 1 είναι:

$$P_2 = 1 - P_1 = 1 - e^{-n/m}$$

Εφόσον έχουμε μία συνάρτηση hash, δηλαδή σε κάθε εισαγωγή γίνεται αντιστοίχιση σε μία θέση του πίνακα, η πιθανότητα για false positive απάντηση ισούται με

$$\boxed{P = 1 - e^{-n/m}}$$

Αν $m = 8n$, η πιθανότητα για false positive απάντηση ισούται με:

$$P' = 1 - e^{-n/8n} = 1 - e^{-1/8} = 0,1175$$

(b) Για $k \geq 1$ hash συναρτήσεις:

- Εάν $k = 1$ πρόκειται για την περίπτωση του (a) ερωτήματος
- Εάν $k > 1$:

Τότε θεωρούμε τον πίνακα A με τις m του θέσεις αρχικοποιημένες στο 0.

- Για τη λειτουργία "πρόσθεσε το x στο S": τροφοδοτούμε το x στις k συναρτήσεις hash ώστε να το αντιστοιχίσουν σε k θέσεις του πίνακα A. Τα bits των k αυτών θέσεων τίθενται ίσα με '1'.

- Για τη λειτουργία "επέλεξε εάν $x \in S$": τροφοδότησε το x στις k συναρτήσεις hash ώστε να το αντιστοιχίσουν σε k θέσεις του πίνακα A. Εάν έστω και ένα (bit από τα k) είναι 0 τότε θεωρούμε ότι το στοιχείο ΔΕΝ βρίσκεται στον πίνακα A. Εάν είναι όλα τα bits '1' τότε θεωρούμε ότι βρίσκεται στον πίνακα A. Στην πραγματικότητα εάν και τα k bits είναι '1' τότε είτε το στοιχείο όντως βρίσκεται στον A, είτε τα k bits αυτά έχουν τεθεί στο '1' "καταλάθως" ύστερα από εισαγωγή άλλων στοιχείων. (false positive περίπτωση)

Ομοίως με το (α) η διαγραφή στοιχείου δεν επιτρέπεται καθώς από τα k bits που θα αλλάξουν από '1' σε '0' είναι αρκετά πιθανό να αλλάξουν και άλλες ομάδες bit που αντιστοιχούν σε άλλα στοιχεία (false negative περίπτωση)

Πιθανότητα για false positive απάντηση

Ομοίως με πριν η πιθανότητα κάποιο bit να μην τεθεί ίσο με '1' ύστερα από εισαγωγή στοιχείου είναι ίση με $P_1 = \left(1 - \frac{1}{m}\right)$ για 1 hash συνάρτηση.

Για τις k συναρτήσεις: $P_1 = \left(1 - \frac{1}{m}\right)^k$

Έτσι για μεγάλα m $\lim\limits_{m \to \infty} \left(1 - \frac{1}{m}\right)^m = e^{-1}$

άρα η P_1 γίνεται: $\left(1 - \frac{1}{m}\right)^k = \left[\left(1 - \frac{1}{m}\right)^m\right]^{\frac{k}{m}} \approx e^{-k/m}$

Εάν έχουμε εισάγει n στοιχεία η πιθανότητα κάποιο bit να παραμένει στο 0 είναι:

$$\left(1 - \frac{1}{m}\right)^{k \cdot n} = e^{-kn/m}$$

και να είναι '1': $1 - e^{-kn/m}$

Έστω τώρα ότι ένα στοιχείο (από τα n) $\nexists S$. Κάθε μία από τις k θέσεις του πίνακα A που αντιστοιχίζονται από τις k hash συναρτήσεις είναι '1' με πιθανότητα:
$1 - e^{-kn/m}$

Η πιθανότητα και οι k θέσεις να είναι ίσες με '1' που θα οδηγήσει σε false positive απάντηση ισούται με:

$$\boxed{\left[1-\left(1-\frac{1}{m}\right)^{kn}\right]^k \approx \left(1-e^{-kn/m}\right)^k} \quad (\text{Ρτελικό})$$

Η βέλτιστη τιμή του k είναι αυτή που θα ελαχιστοποιήσει την πιθανότητα για false positive απάντηση, δηλαδή την Ρτελικό.

Έστω $f(k) = \left(1-e^{-kn/m}\right)^k$

$$f'(k) = \frac{\overbrace{\left(1-e^{-kn/m}\right)^k}^{A(k)} \cdot \overbrace{\left[m\left(e^{kn/m}-1\right)\cdot \ln\left(1-e^{-kn/m}\right)+nk\right]}^{B(k)}}{m\left(e^{kn/m}-1\right)}$$

$f'(k) = 0 \Rightarrow A(k) = 0$ ή $B(k) = 0$.

• Για $A(k) = 0 \Rightarrow \left(1-e^{-kn/m}\right)^k = 0 \Rightarrow 1-e^{-kn/m} = 0 \Rightarrow$
$\Rightarrow e^{-kn/m} = 1 \Rightarrow e^{-kn/m} = e^0 \overset{\ln}{\Rightarrow} \ln e^{-kn/m} = \ln e^0 \Rightarrow$
$\Rightarrow -kn/m = 0 \Rightarrow \boxed{k=0}$ άτοπο αφού $k>1$.

• Για $B(k) = 0 \Rightarrow \left(e^{kn/m}-1\right)\cdot \ln\left(1-e^{-kn/m}\right) + \frac{kn}{m} = 0$

Αποδεικνύεται γραφικά ότι η ρίζα του $B(k)$ είναι η $k = \frac{m}{n} \ln 2$, αφού

$$B\left(\frac{m}{n}\ln 2\right) = m \cdot \left(e^{\frac{m}{n}\cdot\frac{n}{m}\ln 2} - 1\right) \cdot \ln\left(1 - e^{-\frac{m}{n}\cdot\frac{n}{m}\ln 2}\right) + n \cdot \frac{m}{n}\ln 2 =$$

$$= m(2-1) \cdot \ln(1 - 0{,}5) + m\ln 2 =$$

$$= m \cdot \ln\frac{1}{2} + m\ln 2 = m(\cancel{\ln 1}^{0} - \ln 2) + m\ln 2 =$$

$$= -m\ln 2 + m\ln 2 = 0$$

Επομένως η βέλτιστη τιμή του k είναι η $k = \frac{m}{n} \cdot \ln 2$

Αν $m = 8n$, η πιθανότητα για false positive είναι ίση με

$$P = \left(1 - e^{-kn/m}\right)^k = \left(1 - e^{-k/8}\right)^k$$

Για $m = 8n$ $\quad k_{βέλτιστο} = \frac{8n}{n} \cdot \ln 2 = 6$

άρα $\quad P = \left(1 - e^{-6/8}\right)^6 = 0{,}0216$

Άσκηση 6

• Για την αποθήκευση του S θα χρησιμοποιηθεί η δενδρική δομή Trie. Για να διασχίσουμε όλο το S (και τα N γράμματα) θα χρειαστούμε χρόνο O(N), καθώς σε κάθε εξέταση συμβολοσειράς οι ενέργειες που γίνονται για κάθε γράμμα είναι σταθερού χρόνου. Όσον αφορά τη χωρική πολυπλοκότητα στη χειρότερη περίπτωση θα χρειαστούμε τόσα φύλλα όσα και τα γράμματα της S, άρα O(N) η πολυπλοκότητα.

• Ερώτηση: Ποιο είναι το μέγιστο κοινό πρόθεμα;
Ανάγουμε το LCA σε RMQ. Αρκεί μία φορά.
Χρησιμοποιούμε k-1 RMQ queries για να βρούμε το LCA των k συμβολοσειρών. BFS για να βρω το μέγιστο κοινό πρόθεμα.
Η χρονική πολυπλοκότητα για την αναγωγή του LCA σε RMQ είναι O(N), για την δημιουργία του Sparse Table είναι O(nlogn), για την απάντηση σε κάθε query του RMQ είναι O(1) και για την απάντηση στο αρχικό ερώτημα είναι O(N)