ΠΕΤΣΑ ΓΕΩΡΓΙΑ
ΑΜ: 3200155

Άσκηση 1

$$\log^9 n < \log n^{\log n} < \sqrt[9]{n} < n\log(\log n)^9 < n\log_9 n < n^{\log n} < 2^{\log^2 n} <$$

$$< n^{\sqrt{n}} < n! < \sqrt{n}^{\,n} < 2^{n\log n} < n^n$$

Άσκηση 2

- $T(n) = T(n/2) + O(n)$ $\quad a = 1, \quad b = 2, \quad d = 1$

  Συνεπώς $b^d > a$ (αφού $2^1 > 1$)

  Άρα με βάση το master theorem $T(n) = O(n^d) = O(n^1) = \boxed{= O(n)}$

- $T(n) = T(n/2) + O(1)$ $\quad a = 1, \quad b = 2, \quad d = 0 \quad (n^d = n^0 = 1)$

  Συνεπώς $b^d = a$ (αφού $2^0 = 1$)

  Άρα $T(n) = O(n^d \log n) = O(n^0 \log n) = \boxed{O(\log n)}$

- $T(n) = 2T(n/2) + O(1)$ $\quad a = 2, \quad b = 2, \quad d = 0$

  Συνεπώς $b^d < a$ (αφού $2^0 < 2$)

  Άρα $T(n) = O(n^{\log_b a}) = O(n^{\log_2 2}) = O(n^1) = \boxed{O(n)}$

- $T(n) = 4T(n/2) + O(n^2)$ $\quad a = 4, \quad b = 2, \quad d = 2$

  Συνεπώς $b^d = a$ (αφού $2^2 = 4$)

  Άρα $T(n) = O(n^d \log n) = \boxed{O(n^2 \log n)}$

- $T(n) = 4T(n/2) + O(n^3)$ $\quad a = 4, \quad b = 2, \quad d = 3$

  Συνεπώς $b^d > a$ (αφού $2^3 > 4$)

  Άρα $\boxed{T(n) = O(n^3)}$

• $T(n) = 4T(n/2) + O(n)$ $a=4$, $b=2$, $d=1$

Συνεπώς $b^d < a$ (αφού $2^1 < 4$)

Άρα $T(n) = O(n^{\log_b a}) \Rightarrow O(n^{\log_2 4}) = O(n^{2\log_2 2}) = \boxed{O(n^2)}$

Λύση 3

Ο αλγόριθμος υπολογίζει το γινόμενο 2 ακεραίων. Η λογική του αλγορίθμου είναι η εξής: "Μεταφράζοντας" τους a, b σε δυαδικούς αριθμούς και με τη λογική του πολλαπλασιασμού με το χέρι έχουμε:

$$\begin{array}{r} b_n \ldots b_1 b_0 \\ \times\ \underline{a_n \ldots a_1 a_0} \\ * \ldots * \\ \ldots * 0 \end{array}$$

Στην αρχή, το p (που είναι το γινόμενο) περιέχει το 0. Μετά την πρώτη επανάληψη όμως, θα έχει τον b αν το a_0 είναι 1, αλλιώς, θα εξακολουθήσει να έχει το 0. Ουσιαστικά δηλαδή, πολλαπλασιάζει τον $b_n \ldots b_0$ με κάθε ψηφίο του $a_n \ldots a_0$ σαρώνοντας από δεξιά προς τα αριστερά τον a (δηλαδή υποδιπλασιάζοντας τον a, έτσι ώστε να γίνει κάθε φορά shift με το δεξιά). (όπως και με την λογική του πολλαπλασιασμού με το χέρι). Στο τέλος, ο p θα περιέχει το γινόμενο a×b όπου το b κάθε φορά, μετά από κάθε επανάληψη γίνεται shift αριστερά έτσι ώστε σε κάθε επανάληψη να βρίσκεται στη σωστή θέση δηλαδή να προστίθεται στο p το κατάλληλο "βάρος" του b.

Αναλλοίωτη βρόχου

$$\begin{array}{r} b_n \ldots\ldots b_0 \\ \times\ \underline{a_n \ldots a_k \ldots a_0} \\ * \ldots * \ldots * \end{array}$$

Αρχικά, θα έχουμε το πολύ $\log a$ επαναλήψεις.

Οπότε, στην k-οστή επανάληψη έχουμε:

$p = a_0 \cdot b + a_1 \cdot 2b + \ldots + a_k \cdot k \cdot 2b$

$p = b(a_0 + 2a_1 + \ldots + 2k a_k)$

$p = b(a_0 + 2\sum_{i=1}^{k} i a_i)$

Στην $k+1$ επανάληψη:

$$p = a_0 \cdot b + a_1 \cdot 2 b + \dots + 2k a_k b + 2(k+1) a_{k+1}$$

$$p = b\left(a_0 + \sum_{i=1}^{k+1} 2 \cdot i \cdot a_i\right)$$

$$p = b\left(a_0 + 2 \sum_{i=1}^{k+1} i a_i\right)$$

κατά τον ~~[illegible]~~ τερματισμό θα έχουμε ότι

$$p = b\left(a_0 + 2 \sum_{i=1}^{\log a} i a_i\right),$$ δηλαδή ~~[illegible]~~ $p = b \times a$.

Άσκηση 5

```
Askisi5( A[x1, x2 ... x2n])
    MergeSort (A)
    for(i=0 to n-1)
        print (A[i+1], ' ', A[n-i])
```

Έστω προς άτοπο ότι υπάρχει καλύτερος αλγόριθμος. Υποθέτουμε ότι οι 2 αλγόριθμοι κάνουν τις ίδιες επιλογές μέχρι ενός σημείου, και ύστερα ο καλύτερος αλγόριθμος κάνει μια καλύτερη επιλογή. Δηλαδή ισχύει ότι $i_1 = j_1 \dots i_r = j_r$, $i_{r+1} \neq j_{r+1}$, όπου i_i ~~[illegible]~~ η επιλογή του αλγόριθμου Askisis και j_i η επιλογή του καλύτερου αλγορίθμου. Τότε το ζεύγος j_{r+1} θα αποτελούνταν σίγουρα από έναν ελάχιστο αριθμό που απομένει και οι προηγούμενοι ελάχιστοι (αφού ο πίνακας είναι ταξινομημένος και οι μέγιστοι έχουν ήδη μπει σε ζεύγη) και έναν που δεν είναι ο μέγιστος που απομένει, αλλά ένας μικρότερός του. Αυτό το άθροισμα θα είναι μικρότερο μεν από το i_{r+1}, όμως το ζεύγος j_{r+2} θα είναι μεγαλύτερο από το i_{r+2} ~~[illegible]~~ εφόσον ο επόμενος ελάχιστος αριθμός στην $r+2$ επιλογή είναι μεγαλύτερος από τον ελάχιστο στην $r+1$, και το ~~[illegible]~~ ζεύγος του $r+2$ θα είναι επίσης μεγαλύτερο από το ~~[illegible]~~ ζεύγος του $r+1$. Οπότε το συνολικό άθροισμα του j_{r+2} είναι μεγαλύτερο από το i_{r+2}, συνεπώς η επιλογή που κάνει ο καλύτερος αλγόριθμος δεν είναι βέλτιστη.